НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

На правах рукопису

ЯЦИШИН АНДРІЙ ЮРІЙОВИЧ

УДК [004.65:004.415.2](043.3)

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ПОБУДОВИ РОЗПОДІЛЕНИХ СХОВИЩ ДАНИХ ГІБРИДНОГО ТИПУ

05.13.06 – Інформаційні технології

ДИСЕРТАЦІЯ

на здобуття наукового ступеня
кандидата технічних наук

Науковий керівник:
доктор технічних наук, професор
Томашевський Валентин Миколайович

Київ – 2016

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ACID (Atomicity, Consistency, Isolation, and Durability) – атомарність, сталість, ізоляція і довготриваліть даних;

BASE (Basically Available, Soft state, Eventual consistency) – базова доступність даних у «м'якому» стані з поступовою сталістю ;

CASE – комп'ютеризоване проектування та створення програм;

CRM (Customer Relationship Management) – система управління стосунками з клієнтами;

DPA (Data Publishing Area) – вітрини даних;

DSA (Data Staging Area) – перехідна область;

EDL (Extract Description Language) – мова опису витянення даних;

EDW (Enterprise Data Warehouse) – корпоративне сховище даних

ERP (Enterprise Resource Planning) – корпоративне планування ресурсів

E/R (Entity-Relationship) – модель «сутність-зв'язок»;

ETL (Extract-Transform-Load) – методологія інтеграції даних «Вивантаження-Перетворення-Завантаження»

EVER (Event-Entity-Relationship) – модель, що підтримує моделювання інформаційних структур з точки зору пов'язаних осіб і подій;

HOLAP (Hybrid On-Line Analytical Processing) – гібридна модель представлення даних;

JSON (JavaScript Object Notation) – формат запису об'єктів JavaScript;

MAC (Multidimensional Aggregation Cube) – модель багатовимірного куба;

ME/R (Multidimensional Entity/Relationship) – багатовимірна модель «сутність-зв'язок»;

MDL (Map Description Language) – мова опису прив'язування даних;

MOLAP (Multidimensional On-Line Analytical Processing) – багатовимірна модель представлення даних із використанням універсальної багатовимірної моделі;

NoSQL (Not Only SQL) – системи баз даних не на основі відношень;

OODB (Object-oriented Database) – об’єктно-орієнтована база даних;

OLAP (On-Line Analytical Processing) – аналітична обробка у режимі реального часу;

OLTP (On-line Transaction Processing) – транзакційна обробка у режимі реального часу;

OSS (Operating Source Systems) – операційні бази даних;

ROLAP (Relational On-Line Analytical Processing) – реляційна модель представлення даних;

SDL (Source Description Language) – мова опису джерел даних;

SQL (Structured Query Language) – мова структурованих запитів;

UML (Unified Modelling Language)– уніфікована мова моделювання;

XML (eXtensible Markup Language) – мова розширюваної розмітки;

БД (Database) – база даних;

БВБД (Multidimensional Database) – багатовимірна база даних;

БМФ (Multidimensional Fact Model) – багатовимірна модель фактів;

БД XML (XML DB) – база даних XML;

ВБД (Virtual Database) – віртуальна база даних;

ВД (Data Mart) – вітрина даних;

ІАС ПБ – інформаційно-аналітична система «Прозорий бюджет»;

ІС ВУ – інформаційна система «Віртуальний університет» Міністерства фінансів України;

ІСУДФ – інтегрована система управління державними фінансами;

ІС (IS) – інформаційна система;

ІТ (IT) – інформаційна технологія;

ІУС (IMS) – інформаційно управляюча система;

МБ (Megabyte) – мегабайт;

МБСД (Multibase Data Warehouse) – мультибазове сховище даних;

НД (Data Store) – носій даних;

ОС (OS) – операційна система;

ПЗ – програмне забезпечення;

ПВУ – портал «Віртуального університету» Міністерства фінансів України;

ПЖ ФУ – портал журналу «Фінанси України» Міністерства фінансів України;

ПІПО АФУ – портал Інституту післядипломної освіти «Академії фінансового управління» Міністерства фінансів України;

РБД (Relational Database) – реляційна база даних;

СДГТ (Hybrid Data Warehouse) – сховище даних гібридного типу;

СКБД (Database Management System) – система керування базою даних;

СКМБСД (Multibase Data Warehouse Management System) – система керування мультибазовим сховищем даних;

СД (Data Warehouse) – сховище даних;

СУДФ –система управління державними фінансами;

УБМ – узагальнена багатовимірна модель;

ТСУДФ – типова система управління державними фінансами;

ЦФ – цільова функція;

1/мс – елементів даних за мілісекунду;

1/мс*грн – елементів даних за мілісекунду на гривню;

1/мс^2*грн – елементів даних за мілісекунду виконання запиту на гривню, за мілісекунду оптимізації сховища.

ВСТУП

Актуальність теми. Умови розвитку сучасного інформаційного простору вказують на необхідність опрацювання великих обсягів даних структурованого, слабко-структурованого і неструктурованого характеру, які знаходяться у різнорідних джерелах: реляційних і багатовимірних базах даних, базах даних XML і NoSQL, структурованих і неструктурованих текстових файлах, геоданих, медіафайлах тощо.

Проблема ефективної роботи з даними є багатоаспектною і завжди привертала увагу вчених і фахівців-практиків та знайшла своє відображення в працях учених, зокрема N. Debbarma, G. Nath, H. Das [1], J.F. Kimpel [2], J.O.Chan [3], M. Jarke,M. Jeusfeld, C. Quix, P. Vassiliadis [4], Y. Yuan, R.L.X.Zhang [5], M. Jarke, M. Lenzerini, Y. Vassiliou, P. Vassiliadis [6], N.Rahman [7], L. Song, Y. Zhan, Y. Li [8], A. Gosain, S. Sabharwal, R. Gupta [9], C.Blanco, I.de Guzmán, E. Fernández-Medina [10].

Щоправда, вчені досліджували проблему побудови сховищ даних не окремо, а сукупно із питаннями фрагментації даних, проблемою «великих даних» та розподіленого зберігання даних: A.Cuzzocrea, L.Bellatreche, Y.Song [11], A. Boulmakoul, L. Karim, M.H. Laarabi, R. Sacile, E. Garbolino [12], A.Gupta, F. Yang, J. Govig, A. Kirsch, K. Chan [13], R. Sharma, G. Sharma [14], Y.Tian, T. Zou, F. Özcan, R. Goncalves, H. Pirahesh [15].

Інформаційним технологіям опрацювання даних присвячена велика кількість наукових досліджень: щодо обробки геоданих в сховищах даних - G.Viswanathan, M. Schneider [16], A. Ablimit, F. Wang, H. Vo, R. Lee, Q. Liu, X.Zhang, J. Saltz [17], A. Aji, X. Sun, H. Vo, Q. Liu, R. Lee, X. Zhang, J. Saltz , F.Wang [18]; обробки генетичних даних в сховищах даних - T.Triplet, G.Butler [19], S.G. Neogi, P. Vasilis, M. Krestyaninova, M. Kapushesky, I. Emam, A.Brazma, U. Sarkans [20]; інтеграції даних - A. Quiroz, E. Huang, L. Ceriani [21], A. Cal, D.Calvanese, G. Giacomo, M. Lenzerini [22], S.R. Jeffery, L. Sun, M. DeLand, N.Pendar, R. Barber, A. Gald [23].

Ефективне використання різних типів даних, розташованих на просторово розосереджених обчислювальних засобах, залежить від застосування технологій роботи з гетерогенними даними, консолідації однотипних даних, гібридних баз даних, паралельного опрацювання даних та ін. Зокрема, питанням консолідації даних присвячено праці A. Gupta, D. Tan, J.Kulesza, R. Pathak, S. Stefani, V. Srinivasan [24], L. Bellatreche, A. Cuzzocrea, Y.Song [25], M. Golfarelli, J. Aligona, E. Gallinuccib, P. Marcela, S. Rizzib, [26], H.J. Watson [27], W. Eckerson, D.Power [28]. Для фізичного розподілу даних переважно використовуються архітектура «Virtual Database», технології BigData – великих даних (платформа Hortonworks на основі Hadoop і рішень Apache Foundation). Паралелізм у сховищах даних розглядається у працях T. Stöhr, H.Martens, E. Rahm [29], T. Lauer, A.Datta, Z.Khadikov, C. Anselm [30], J.P.Costa, P. Furtado [31], результати досліджень яких використовуються для логічного розподілу на основі бібліотеки типів та фізичного розподілу на основі фрагментації даних. Методи структурування баз і сховищ даних досліджуються в працях X. Wu, D. Theodoratos, W.H. Wang, T. Sellis [32], C. Zhang, X. Yao, J.Yang [33], A. Gupta [34], Y.J. Chang, J.T. Horng, B.J. Liu [35], J.T. Horng, C.W.Chen [36] і переважно зводяться до побудови оптимальних планів виконання запитів з використанням методик організації даних.

Питанням управління контентом сховищ даних присвячені праці W.Inmon [37..38], R. Kimball [39..40], D. Hackney [41] та багатьох інших вчених. Задачі визначення архітектури розподілених сховищ із складною ієрархічною структурою, насамперед гомогенних з великою кількістю файлових даних, розглядаються у працях вітчизняних вчених А.А. Пашнєва, О.В. Курка, О.В.Мігури [42], формалізації предметної області багатовимірних баз даних та методиці оцінки якості управління даними присвячені робота Г.А. Кучука, А.В.Петрова, Р. В. Корольова [43], [44], [45], структурування сховищ даних, побудованих на основі поширених платформ, наприклад Hadoop досліджуються у роботах Д.В. Приймака та О.І. Акимишина [46], а питанням

формалізації процесів структурування, опрацювання даних сховищ та побудови просторів даних присвячені роботи Н.Б. Шаховської [47..48].

Існуючі підходи однак не враховують особливості побудови розподілених сховищ даних гібридного типу (РСД ГТ), які зберігають дані з різнорідних джерел на різних просторово розосереджених обчислювальних засобах і забезпечують ефективність експлуатації сховища за економічними критеріями. Зокрема, у роботах відсутні комплексні дослідження наявності різних типів джерел даних, різних рівнів абстракції даних, міжрівневих переходів між ними та не розглядається можливість підвищення ефективності роботи сховища в процесі його експлуатації. У існуючих моделях побудови сховищ даних не досліджується ефективність логічного і фізичного розподілу даних.

Таким чином, попри наукові результати дослідження різних аспектів побудови та експлуатації сховищ даних, залишається невирішеною проблема побудови ефективних РСД ГТ, що зумовило актуальність науково-прикладного завдання створення інформаційної технології (ІТ) побудови сховищ зазначеного типу та вимагає концептуально-методологічних інновацій у цій сфері на основі проведення ґрунтовних досліджень.

Зв'язок роботи з науковими програмами, планами, темами.

Дисертаційна робота виконувалась у відповідності з науково-технічними планами кафедри автоматизованих систем обробки інформації та управління Національного технічного університету України «Київський політехнічний інститут» і науково-дослідними роботами Міністерства фінансів України, в яких автор при розробці та впровадженні сучасних інтернет-технологій в роботу органів виконавчої влади брав безпосередню участь:

– «Модернізація управління державними фінансами в контексті адміністративної реформи» (державний реєстраційний 0111U007012) – розробка інформаційно-аналітичної системи «Прозорий бюджет» Міністерства фінансів України, 2011 рік.;

– «Моделювання та прогнозування розвитку фінансової системи України» (державний реєстраційний № 0111U007014) – розробка

інформаційної системи «Віртуальний університет» Міністерства фінансів України, 2011 рік.

Мета і задачі дослідження.

Метою дисертаційної роботи є зниження сукупної вартості збереження та обробки даних у розподілених сховищах даних гібридного типу за рахунок нової інформаційної технології, створеної на основі комплексу математичних моделей і методів аналізу та оцінювання процесів опрацювання даних у сховищах зазначеного типу і оптимізації їх структури.

Для досягнення поставленої мети в дисертаційній роботі вирішуються такі завдання:

1) аналіз процесу та методів побудови РСД ГТ і проблем, що виникають при цьому;

2) розроблення методу побудови РСД ГТ з мінімальною сукупною вартістю збереження та обробки даних при обмеженнях на час виконання запитів і кількість реплікованих копій даних;

3) розроблення математичних моделей багаторівневого описання РСД ГТ та їх відображень і взаємодії;

4) розроблення моделей та методів логічного і фізичного розподілу даних у РСД ГТ і дослідження ефективності та збіжності розроблених методів;

5) створення ІТ побудови РСД ГТ на основі розроблених моделей, методів та засобів;

6) експериментальне дослідження створеної ІТ побудови РСД ГТ.

Об'єктом дослідження є процес побудови розподілених сховищ даних гібридного типу.

Предметом дослідження є моделі, методи та засоби побудови розподілених сховищ даних гібридного типу.

Методи дослідження. У процесі досліджень для побудови РСД ГТ використовувались: методи системного аналізу – для декомпозиції процесу побудови; теорія реляційних баз даних і реляційна алгебра – для формалізації описання РСД ГТ і оптимізації запитів; методи обчислювального інтелекту,

еволюційного програмування – для знаходження конфігурацій розміщення та реплікації даних.

Наукова новизна одержаних результатів:

1) вперше запропоновано формальну алгебричну систему сховищ даних гібридного типу, яка відрізняється від існуючих набором основ (сутності, елементи даних РБД, БВБД, БД XML і NoSQL) та операцій (новими є операції перетворення основ між собою, виділення структурованої і слабко-структурованої частини відношення, з'єднання сутностей, побудови зв'язної множини), дослідження властивостей яких дозволило оперувати даними, представленими різними моделями, та будувати для них РСД ГТ;

2) удосконалено метод спрощення формул алгебричної системи РСД ГТ за рахунок визначення перспективних множин інваріантів з наступним деталізованим вибором кращого у цих множинах, який відрізняється від існуючих наявністю правил оцінювання інваріантів формули запиту на основі критерію вартості, що дозволяє отримати інваріант виразу запиту до сховища з мінімальною сукупною вартістю збереження та обробки даних;

3) вперше розроблено метод побудови РСД ГТ з мінімальною сукупною вартістю збереження та обробки даних, що здійснює логічний та фізичний розподіл даних, використовує запропоновану алгебраїчну систему та відрізняється від існуючих тим, що враховує інформацію про дані та запити;

4) розроблено ІТ побудови РСД ГТ на базі комплексу математичних моделей і методів аналізу та оцінювання процесів опрацювання даних у сховищах зазначеного типу та оптимізації структури сховищ, що дозволяє створювати інформаційні системи, у яких ефективно обробляються гетерогенні дані, що зберігаються розподілено.

Особистий внесок здобувача.

Усі наукові результати, отримані у дисертації, одержані здобувачем особисто. У друкованих працях, опублікованих у співавторстві, здобувачу належать такі результати: у праці [49] здобувачем проведено аналіз існуючих джерел з проектування та створення архітектур розподілених сховищ даних

гібридного типу, у праці [50] – дано формальне визначення мультибазового сховища даних, сформульована математична модель задачі проектування та оптимізації РСД ГТ і запропоновано метод її розв'язання.

Практичне значення отриманих результатів.

Одержані в дисертаційній роботі результати можуть бути використані при проектуванні систем обробки даних гетерогенного та розподіленого типу за допомогою інформаційної технології побудови РСД ГТ. Розроблені моделі, методи та засоби дозволяють здійснювати побудову таких сховищ даних з мінімальною сукупною вартістю збереження та обробки даних.

Практичне значення роботи полягає у зниженні сукупної вартості збереження та обробки даних у інформаційних системах, розроблених на основі створеної інформаційної технології.

Результати дисертаційної роботи використані при розробленні, впровадженні і експлуатації таких інформаційних та інформаційно-аналітичних систем (див. Додаток Л):

1) сховища даних для системи управління державними фінансами (СУДФ) Міністерства фінансів України;

2) інформаційно-аналітичної системи «Прозорий Бюджет» Міністерства фінансів України (свідоцтво про реєстрацію авторського права на твір № 43465);

3) інформаційної системи «Віртуальний Університет» Міністерства фінансів України (свідоцтво про реєстрацію авторського права на твір № 43464);

4) порталу Інституту післядипломної освіти ДННУ «Академія фінансового управління»;

5) порталу періодичного наукового видання (свідоцтво про реєстрацію авторського права на твір № 55790).

Використання результатів дисертаційного дослідження підтверджено відповідними актами впровадження (див. Додаток М).

Результати дослідження впроваджені в навчальний процес кафедри автоматизованих систем обробки інформації та управління Національного

технічного університету України "Київський політехнічний інститут" для викладання дисципліни "Організація баз даних і знань".

Основні положення роботи можна використовувати для:

– розробки та впровадження систем збереження та обробки даних розподіленого і гетерогенного характеру;

– зниження сукупної вартості збереження та обробки даних у інформаційних системах і підвищення швидкості їх опрацювання;

– розробки засобів логічного та фізичного розподілу даних на основі аналізу даних і статистики запитів.

Апробація результатів дисертації.

Основні положення та результати дисертаційної роботи обговорювались на 13 наукових конференціях, в тому числі: Всеукраїнській школі-семінарі молодих вчених та студентів "Сучасні комп'ютерні інформаційні технології" ("ACIT-2011", "ACIT-2012", Тернопіль, 2011, 2012 рр.); Міжнародній науковій конференції "Інтелектуальні системи прийняття рішень і проблеми обчислювального інтелекту" ("ISDMCI’2011", "ISDMCI’2012", "ISDMCI’2014"); І Міжнародній науково-практичній конференції "Сучасні інформаційні системи та технології" ("ASIT-2012", Суми, 2012 р.).

Публікації. Основні положення дисертаційного дослідження опубліковано в 2-х монографіях та 20 друкованих наукових працях (18 опубліковано одноосібно), у тому числі: 7 статей у фахових наукових виданнях, що входять у міжнародні наукометричні бази та 13 публікацій у матеріалах міжнародних конференцій.

Структура та обсяг дисертації.

Дисертаційна робота складається із вступу, чотирьох розділів, висновків, списку використаних джерел та додатків. Повний обсяг дисертації становить 192 сторінок, з них 151 сторінок основного тексту, 60 рисунків, 2 таблиці, список використаних джерел з 116 найменувань на 14 сторінках, 7 додатків на 27 сторінках.

РОЗДІЛ 1. ПРОБЛЕМА ПОБУДОВИ РОЗПОДІЛЕНИХ СХОВИЩ ДАНИХ ГІБРИДНОГО ТИПУ

У даному розділі представлений аналіз робіт, в яких описано підходи, моделі, методи та засоби побудови сховищ даних, сформульовано мету і завдання дослідження, а також концепцію побудови розподілених сховищ даних гібридного типу.

1.1. Загальна характеристика баз і сховищ даних

Будь-яка інформаційна система у процесі свого функціонування дозволяє користувачеві взаємодіяти з даними. Для опису колекцій даних, що використовуються з метою їх зберігання, керування та обробки, використовують два основних поняття: бази та сховища даних.

Наведемо деякі визначення, які будемо використовувати в подальшому.

База даних (БД) – це сукупність екземплярів різних типів записів і відношень між записами та елементами, як визначено в роботі [51]. Основними видами БД, що використовуються при побудові сховищ даних є реляційні, об’єктні та об’єктно-орієнтовані, а також багатовимірні.

Реляційні БД представляють дані у вигляді залежностей між атрибутами. Таблиці реляційної бази даних містять записи (рядки), кожен з яких складається з полів (стовпців). У звичайній реляційній базі даних, кількість полів кожного запису (ключа) можуть однозначно ідентифікувати цей запис.

Багатовимірні БД дозволяють зберігати дані як значення міри відносно набору атрибутів вимірів. Багатовимірна модель даних є n-мірним масивом (так званий гіперкуб або куб). Кожен вимір має зв'язану з ним ієрархію рівнів консолідованих даних.

Міри в багатовимірному масиві відповідають стовпцям у таблиці реляційної бази даних, значення яких функціонально залежить від значень інших стовпців. Виміри виступають як індекси для визначення значення в багатовимірному масиві даних. Значення в одній комірці може представляти

агреговані дані, розраховані з більш детальних даних нижчого рівня тієї ж розмірності.

Об’єктно-орієнтовані бази даних визначають всі дані як об’єкти деяких класів, відомих базі даних. При цьому підтримуються додаткові операції над об’єктами, які відповідають методам їх класів.

Об’єктно-реляційні бази даних є гібридом реляційних та об’єктно-орієнтованих баз даних і підтримують реляційну модель даних, у якій елементами відношень є об’єкти. Дана модель дозволяє використовувати методи класів цих об’єктів.

Для забезпечення взаємодії між базами даних та іншими застосуваннями або користувачем використовуються системи керування базами даних (СКБД), які надають загальний репозитарій для зберігання й опрацювання структурованих даних, керують даними в зовнішній пам'яті (на дисках) та в оперативній пам'яті з використанням дискового кеша; журналізують зміни та здійснюють резервне копіювання й відновлення бази даних після збоїв. Крім того, СКБД забезпечує підтримку мов БД (мов визначення даних та маніпулювання даними).

У процесі розвитку інформаційних систем виникла потреба не лише зберігати дані та керувати ними, але й відповідати вимогам певних застосувань, наприклад, систем підтримки прийняття рішень, що базуються на використанні складних запитів до баз даних.

Сховище даних – предметно-орієнтована інформаційна база даних, спеціально розроблена та призначена для підготовки звітів і бізнес-аналізу з метою підтримки ухвалення рішень в організації. На рис. 1.1 показано схему використання сховища даних в типовій інформаційній системі (ІС).

Зазвичай у типовій ІС застосування - ядро, що обробляє дані за закладеним в нього алгоритмом, запитує інформацію у сховища даних, яке у свою чергу виконує відповідні запити до баз даних, і передає інформацію іншим прикладним застосуванням або кінцевим користувачам. Таким чином, сховище даних не

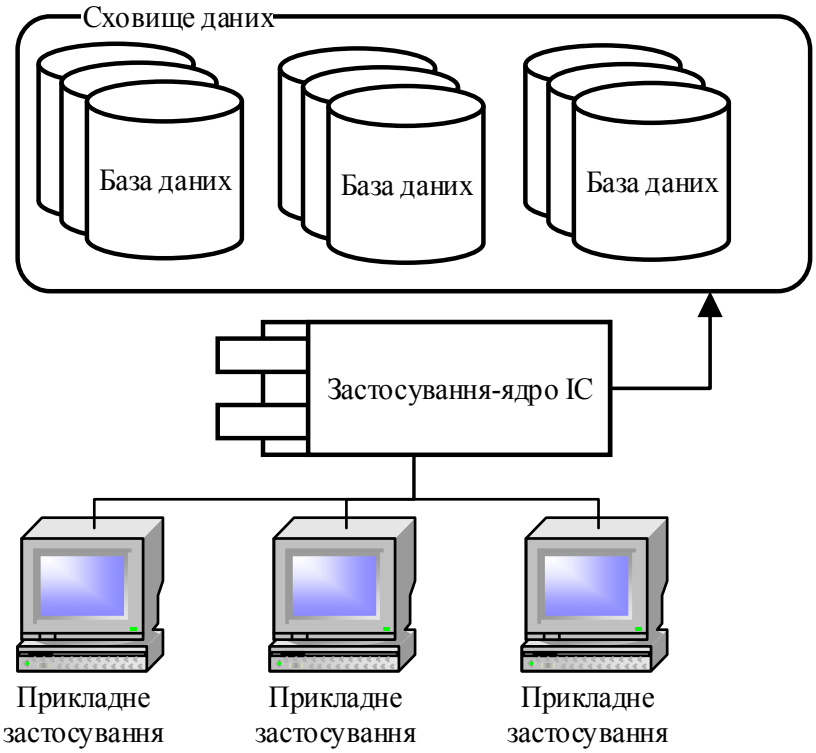


Рис. 1.1. Схема типової ІС [Розроблено автором]

лише виконує функцію зберігання даних, але і надає можливості ефективного доступу до цих даних, однак для забезпечення такого доступу потрібно виконання додаткових умов. У роботі «The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouse» Ральф Кімбол сформулював основні вимоги до СД [39]:

- підтримка високої швидкості отримання даних зі сховища;
- підтримка внутрішньої несуперечності даних;
- можливість отримання і порівняння так званих зрізів даних (slice and dice);
- наявність зручних утиліт перегляду даних сховища;
- повнота і достовірність даних, що зберігаються;
- підтримка якісного процесу поповнення даних.

Сховища даних переважно є інтеграцією реляційної та багатовимірної моделей даних і складаються з реляційних та багатовимірних баз даних. Такі сховища будемо називати сховищами даних гібридного типу (за визначенням у роботі [47] є зрізом сховища даних, масивом тематичної, вузьконапрямленої

інформації, що орієнтована, наприклад, на користувачів однієї робочої групи або департаменту.

1.2. Загальні підходи до побудови сховищ даних

Існують різні види сховищ даних в залежності від їх структури або способу побудови. Сховища даних можуть будуватися за двохрівневою архітектурою, що включає джерела даних і вітрини даних, а також трьохрівневою архітектурою, яка включає додатково так зване «центральне сховище» даних.

Крім того, можна виділити операційне сховище даних (ОСД), що є предметно-орієнтованим, інтеґрованим, змінюваним набором консолідованих даних, який містить поточну (не історичну) деталізовану інформацію [51].

Одними із основних підходів до побудови сховища даних є корпоративна інформаційна фабрика Білла Інмона [37..38] та шина Ральфа Кімбола [40].

У книзі «The Data Warehouse Toolkit. Second Edition» Р. Кімболл (R. Kimball) запропонував підхід до побудови сховищ даних [40], який також відомий як підхід «знизу вгору», оскільки проектування сховища починається з нижнього рівня сховища (вітрин даних) і закінчується його верхнім рівнем (операційними базами даних). Сховище, побудоване за цим підходом, називається сховищем даних з архітектурою шини, або просторовим сховищем і включає операційні бази даних, перехідну область, вітрини даних, засоби доступу до даних.

У цьому сховищі операційні бази даних зберігають первісну інформацію. Перехідна область слугує для перенесення даних з операційних баз даних до вітрин даних, які містять як детальні, так і агреговані дані.

Запропонована архітектура багатовимірного сховища даних дозволяє ефективно виконувати запити як до агрегованих даних, так і зберігати детальні дані. Використання архітектури шини та понять узгодженості дозволяє отримати незалежні між собою вітрини, які використовують один набір вимірів,

що забезпечує можливість паралельної розробки вітрин різними підрозділами, а також цілісність усієї системи в цілому.

Певною модифікацією підходу Р. Кімболла є підхід незалежних вітрин даних, описаний Wayne Eckerson у роботі [52]. Основною відмінністю цього підходу є незалежність і неузгодженість між собою вітрин даних. Даний підхід не деталізує первинні бази даних, перехідну область і застосування користувачів.

Другим основним підходом до побудови сховищ даних є підхід Інмона або спіральний підхід. Цей підхід відомий як підхід «зверху вниз», оскільки проектування системи починається з верхнього рівня (власне сховища даних) і закінчується нижнім рівнем (вітринами даних) [37..38]. Сховище, побудоване за таким підходом, було названо «корпоративною інформаційною фабрикою». Крім того, наявність вітрин даних дозволяє отримувати агреговані дані та використовувати переваги систем класу OLAP.

Таке сховище, за Інмоном, містить корпоративні застосування, перехідну область, операційне сховище даних, сховище даних підприємства, вітрини даних відділів, застосування систем прийняття рішень, сховища для дослідження або видобування даних, альтернативні системи зберігання даних.

Узагальненим варіантом підходу Інмона є підхід до побудови централізованого сховища даних, у якому багатовимірні представлення забезпечуються реляційною базою даних [52]. Відмінністю від підходу Інмона є те, що в сховищі немає вітрин даних. Цю архітектуру слід вибирати тоді, коли сховище даних є інтегральною частиною стратегічного рішення.

Об’єднане сховище даних [52] передбачає наявність уже розгорнутої системи (бази даних, сховища даних, застосування тощо) та призначене для отримання доступу до даних з існуючих джерел. Основною перевагою цієї архітектури є простота проектування і можливість використання існуючих систем для побудови сховища. Основним недоліком цієї архітектури є складність аналізу даних у вітринах різної структури та гетерогенність побудованої в результаті системи.

Своєрідним поєднанням підходів Інмона та Кімболла є підхід Хекні [41] - так зване об’єднане (federated) сховище даних або узгоджувана вітрина даних, яке складається з первинних баз даних (реляційні бази даних), вітрин даних (багатовимірні сховища), загальної перехідної області, що виконує операції отримання, перетворення та завантаження даних; власного об’єднаного сховища даних, в яке завантажуються дані з реляційних баз даних і з якого вивантажуються дані для багатовимірних вітрин даних. За такої архітектури передбачається подвійне проектування схеми сховища даних – розроблення нормалізованого центрального (корпоративного сховища) та багатовимірних (побудованих за архітектурою шини) вітрин даних. Корпоративне нормалізоване сховище дозволяє коректно зберігати дані, а ненормалізовані вітрини – швидко виконувати запити користувачів. Особливою рисою цієї архітектури є те, що в ній загальні дані розподіляються між обома сховищами.

Проаналізуємо розглянуті підходи до побудови сховищ даних з точки зору проектування сховища даних на базі реляційної та багатовимірної бази даних.

Підходи централізованого сховища даних і незалежних вітрин даних є неефективними в загальному випадку, оскільки представляють собою реляційну або багатовимірну БД, а тому їх недостатньо для побудови СД гібридного типу.

Підхід об’єднаного сховища даних не є прийнятним, так як він призначений для інтеграції існуючих баз даних і не дозволяє здійснювати проектування СД без наявності уже спроектованих баз даних.

Порівняємо підходи до побудови СД Інмона та Кімболла.

Спільні риси. Найбільш очевидною подібністю між підходами є те, що вони базуються на використанні методології інтеграції даних Extract-Transform-Load (ETL) і використовують дані з відміткою часу. Час є важливим виміром в сховищі даних, оскільки дозволяє порівнювати дані у часі для визначення тенденцій. Вимір часу у Кімболла визначено так, що бізнес-користувач може вибирати певні дати, інтервали дат, і періоди звітності. Вимір формує частину

моделі узгоджених вимірів і тому є доступним в усіх вітринах даних. Підхід Інмона може зберігати дані дат в нормалізованих таблицях, які можуть використовуватися для обчислення даних по інтервалах дат, окремих днях і періодах звітності. Завдяки використанню шару ETL, підходи Інмона та Кімболла є залежними від його функції – вивантаження даних з джерела, застосування правил для завантаження даних у сховище або в вітрину даних даними, які є стандартизованими, і як наслідок, важливими для організації.

Відмінності. Основною відмінністю багатовимірного сховища даних є відсутність у підході Кімболла реляційної бази даних, яка виконує запити користувачів. Крім того, у підході Кімболла використовується вітрини, що об’єднуються в шину (принципи узгодженості вимірів і фактів), а в підході Інмона вітрини є незалежними і проектуються під потреби різних відділів.

Після вибору архітектури сховища виникає питання його побудови, тобто визначення структур даних, які необхідні для збереження цих даних і по можливості оптимального функціонування (швидкодії) сховища.

В силу недоліків моделей Інмона та Кімболла є очевидним поєднання цих архітектур, і створення СД гібридного типу. Це було зроблено в моделі Хекні, однак для розв’язання поставленої задачі даного підходу недостатньо, тому що в ньому не деталізується, які дані повинні міститися в реляційній чи багатовимірній формі.

Тому виникає необхідність у рішенні з побудови СД з використанням гібридного підходу та описом розподілу даних у сховищі.

1.3. Моделі і схеми даних

Для розміщення даних у сховищі необхідна модель даних - абстрактне, самодостатнє, логічне визначення об'єктів, операторів і інших елементів, які в сукупності складають абстрактну машину доступу до даних, з якою взаємодіє користувач [53]. Ці об'єкти дозволяють моделювати структуру даних, а оператори – поведінку даних.

Моделі даних, що запропоновані до використання у сховищах даних, можна класифікувати за основою, яка використовується – реляційною (модель Чена, Entity-Relationship [53]), багатовимірною (універсальна багатовимірна модель Енріко Франконі і Ананда Кемблі [53]) або їх поєднанням, як показано на рис 1.2.

Розглянемо моделі даних на основі узагальненої багатовимірної моделі (УБМ). Основна ідея УБМ даних полягає у представленні збережених даних за допомогою наступного набору множин: фактів, мір, вимірів, ієрархії вимірів,

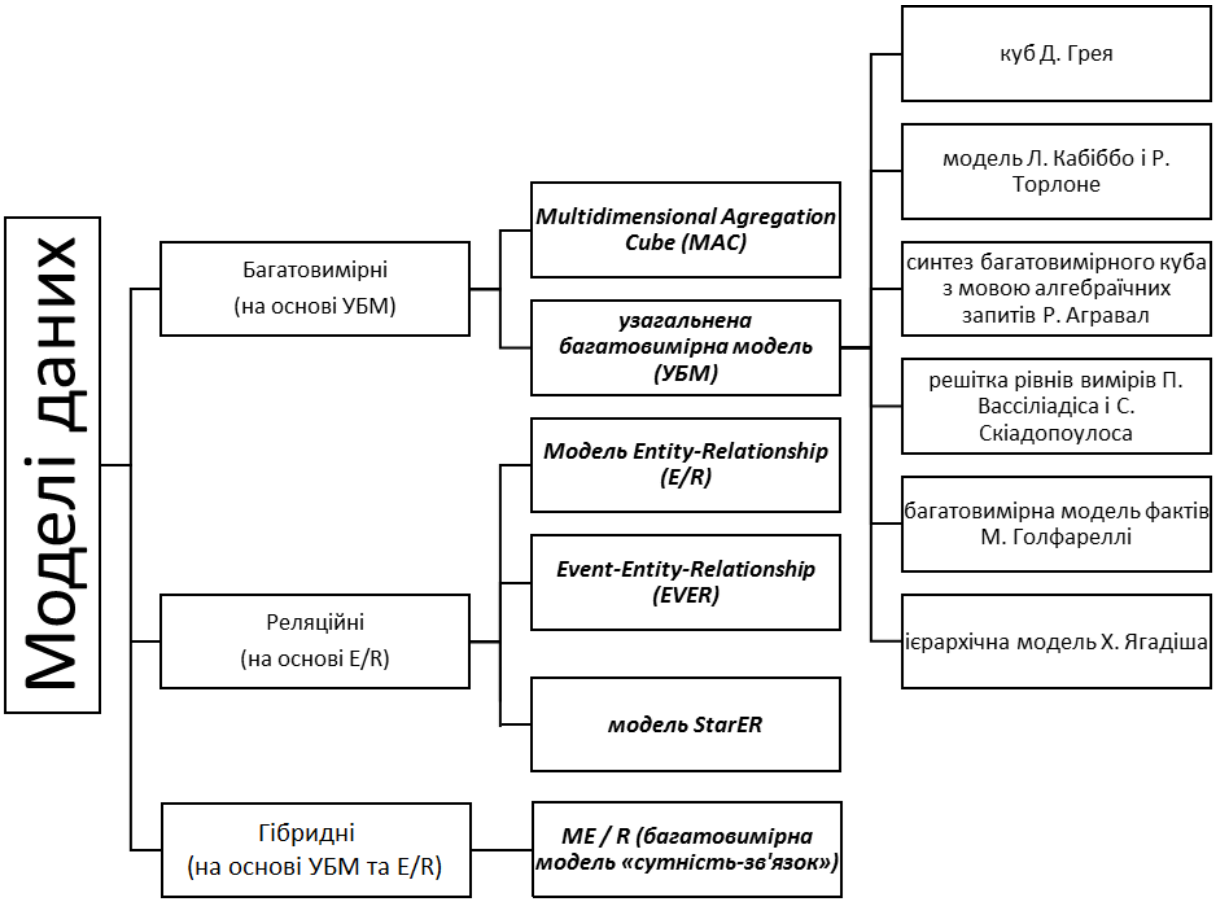


Рис. 1.2. Класифікація моделей даних [Розроблено автором]

типів, атрибутів. Спочатку визначаються ієрархії, потім описується перехід між рівнями ієрархій (операція roll-up), присвоюються назвам атрибутів типи і визначаються факти і відповідні міри. В роботі Е. Франконі і У. Саттлер [54] відзначається, що відсутність інструментарію поетапного проектування СД є серйозним недоліком УБМ даних.

Необхідно зазначити, що УБМ даних створювалася з метою узагальнення існуючих підходів, тому є універсальною структурою, до якої можуть бути приведені інші моделі. Таким чином, в термінах УБМ даних описують схеми організації даних типу «зірка» [55..57] і «сніжинка» [52], [55..57], «куб» Д. Грея [58], модель Л. Кабіббо і Р. Торлоне [59], багатовимірний куб з мовою алгебричних запитів Р. Агравал [60], решітка рівнів вимірів П. Вассіліадіса і С. Скіадопоулоса [61], ієрархічна модель Х.Ягадіша [62], багатовимірна модель фактів М. Голфареллі [63].

Схема організації даних типу «зірка», описана в роботах **[**55..57], складається з таблиці фактів для кожної транзакційної сутності та таблиць вимірів, що отримуються у результаті послідовного застосування оператора згортання ієрархії до компонентних сутностей. Крім того, застосовується оператор агрегації до транзакційних сутностей по ключових атрибутах (вимірах).

Схема організації даних типу «сніжинка», описана в роботі [40], [52], [64] – це схема типу «зірка» з нормалізованими вимірами, у якій для кожної транзакційної сутності створюється таблиця фактів, для компонентних та класифікаційних – таблиці вимірів. Крім того, оператор агрегації застосовується до транзакційних сутностей по ключових атрибутах (вимірах).

М. Голфареллі в роботі [63] представив багатовимірну модель даних, яка називається «багатовимірною моделлю фактів» (БМФ). БМФ – це графічна модель, яка використовує модель типу «зірка» і є незалежною від логічної (багатовимірної або реляційної) моделі. БМФ представляє собою слабо зв'язане дерево (так зване квазі-дерево) атрибутів з коренем, якому приписаний певний факт і з яким з’єднані одномірні атрибути. Важлива відмінність БМФ від інших розглянутих моделей те, що в її основу закладена методологія проектування СД, яка підтримується інструментарієм БМФ. Представлення реальності з використанням БМФ називається вимірною схемою та складається з набору схем фактів, базовими елементами яких є факти, виміри та ієрархії.

Застосування моделі БМФ для створення концептуальної схеми СД описано в роботах [63] та [65].

У зазначених вище роботах не досліджується питання моделювання процесу проектування СД на основі існуючої БД. Виняток становить графічна модель М. Голфареллі, проте істотним недоліком цієї моделі є те, що математичною основою є реляційна алгебра і, отже, проектування СД зводиться до отримання схеми типу «зірка».

Модель Multidimensional Aggregation Cube (MAC) описує дані як виміри, атрибути, рівні вимірів, відношення деталізації, шляхи вимірів [66]. Рівні вимірів відображають класи елементів вимірів. Кожен елемент виміру представляє деякий екземпляр реальної властивості, яка може характеризувати міру OLAP. Різні рівні вимірів можуть бути зв'язані з допомогою відношення деталізації, яке вказує на семантичні відношення між рівнями та описує групування елементів вимірів нижчого рівня в набори, які відповідають елементам виміру вищого рівня.

Набір відношень деталізації при виконанні певних структурних вимог може складати шлях виміру Причому один або кілька шляхів виміру, які поділяють спільні рівні, можуть утворювати вимір. Багатовимірний куб агрегації (MAC) визначається як відношення між доменами одного або декількох вимірів. Такий куб може мати одну або кілька мір, кожна з них може розглядатися як простий атрибут відношення, що представляється як MAC. Екземпляр MAC називається клітинкою MAC або просто клітинкою.

Розглянемо моделі даних на основі моделі «сутність-зв’язок» (Entity-Relationship, E/R). Ця модель, описана в роботі [53], представлена ER-діаграмою, яка використовує три основні графічні символи для відображення реальності: сутності, відношення і атрибути. Сутність визначається як особа, місце, річ або подія, що представляють інтерес для бізнесу або організації. У моделі E/R, визначення унікального ідентифікатора сутності є найбільш важливим завданням. Ці унікальні ідентифікатори називаються ключами-кандидатами з яких можна вибрати ключ, який найбільш часто

використовується для ідентифікації сутності і називається первинним ключем. Відношення між сутностями відображають конструктивну взаємодію та асоціацію між об'єктами в моделі і позначаються лінією між сутностями. Відношення між двома сутностями може бути визначено в термінах потужності, яка є максимальною кількістю екземплярів однієї сутності, що мають відношення до одного екземпляру в іншій таблиці і навпаки. Атрибути описують характеристики властивостей об'єктів.

На базі моделі Entity-Relationship було побудовано низку інших моделей. Зокрема, для проектування сховищ даних було запропоновано використовувати моделі Event-Entity-Relationship (EVER), що підтримує моделювання інформаційних структур з точки зору пов'язаних осіб і подій, та багатовимірну модель «сутність-зв'язок» (ME/R), яка базується також на УБМ даних і розроблена спеціально для багатовимірного моделювання сховищ даних, що описана в роботі «Extending the E/R Model for the Multidimensional Paradigm» [67]. Також, в роботі [68] описана модель StarER, що є концептуальною моделлю для проектування сховищ даних і поєднує семантично багаті конструкції ER моделі з структурою схеми типу «зірка». Дана модель враховує вимоги кінцевого користувача до моделювання сховища даних, які описуються набором концепцій, що повинні бути зосереджені у фазі концептуального моделювання.

На основі розглянутих моделей можна побудувати сховище даних як реляційну або багатовимірну базу даних, та як їх гібридне поєднання, однак не було запропоновано моделі, які поєднують реляційну і багатовимірну модель для підтримки застосувань OLTP, для яких важлива швидкодія виконання запитів, і застосувань OLAP, для яких важлива гнучкість аналізу даних різної структури.

1.4. Проектування сховищ даних

Проектування – це процес визначення архітектури, компонентів, інтерфейсів та інших характеристик системи або її частини (ISO/IEC/IEEE 24765:2010 [69]).

Якщо розглянути систему керування сховищем даних як інформаційну управляючу систему (ІУС) на основі адаптивної технології, то згідно з [70] побудова сховищ даних полягає у визначенні представлень сховища на різних рівнях, кожне з яких отримується з попереднього за допомогою певних етапів проектування (рис.1.3.)

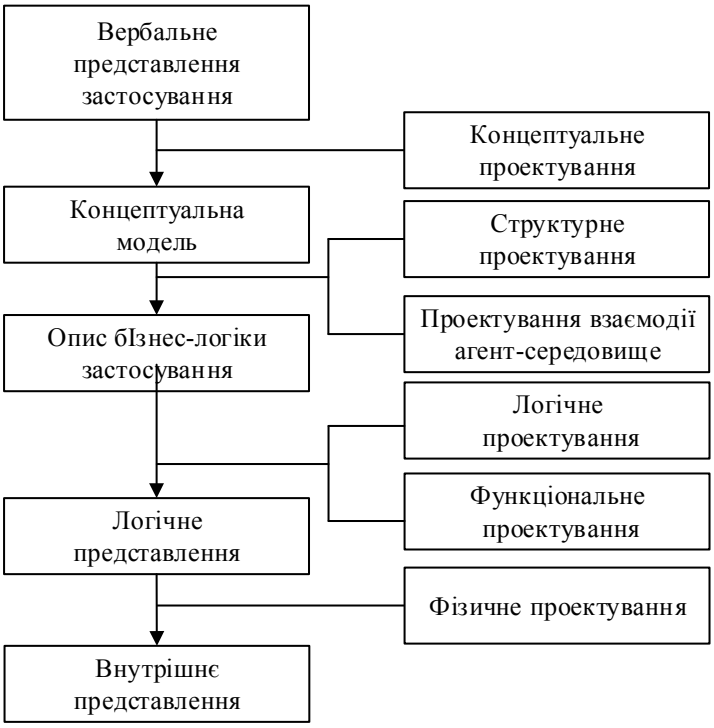


Рис. 1.3. Рівні опису інформаційної інтелектуальної ІУС в адаптивній технології і відповідні міжрівневі відображення (взято з [70])

З показаних на рис. 1.3. етапів побудови для сховищ даних найбільш важливими є концептуальна, логічна та фізична модель, які отримуються в результаті концептуального, логічного та фізичного проектування.

Концептуальна модель відображає структуру даних (елементи даних та зв’язки між ними), що доступна для користувача при побудові запитів до сховища і повинна бути зрозуміла йому. Дана модель отримується у результаті концептуального проектування, при якому створюються вимірні схеми за допомогою побудови дерева атрибутів, відсікання і зрощення дерева атрибутів,

визначення вимірів, мір, ієрархій і здійснюється проектувальником на основі схеми бази даних, фактів та попереднього робочого навантаження [71].

Логічна модель відображає структуру даних (елементи даних та зв'язки між ними), що доступна з носіїв даних сховища для виконання запитів до них безпосередньо або через інструментарій сховища. Для отримання даних через цей інструментарій повинно бути побудоване відображення між концептуальною та логічною моделями. Логічна модель отримується у результаті логічного проектування, яке здійснюється проектувальником. Логічне проектування спрямоване на отримання логічної схеми сховища даних та включає матеріалізацію представлень, трансляцію в таблиці, вертикальну та горизонтальну фрагментації таблиці фактів [71].

Фізична модель відображає параметри конкретних носіїв даних, де зберігаються дані згідно з логічною моделлю, а також параметри функціонування сховища даних як взаємозв'язаної системи. Структура цієї моделі може бути різною в залежності від носіїв даних та процесів взаємодії між носіями у сховищі. Дана модель отримується у результаті побудови фізичної моделі проектувальником, що полягає у оптимальному виборі, як правило евристичними методами, індексів таблиць баз даних сховища [71].

Опишемо ці етапи детальніше.

1.4.1. Концептуальне проектування сховищ даних

Концептуальне проектування відповідає за побудову концептуальної моделі, яка представляє сховище даних з точки зору доступу застосувань та кінцевих користувачів. Цією моделлю описується наявні об'єкти в системі, їх структура (на які інші об'єкти вони поділяються) та операції над ними.

Концептуальна модель може будуватися за одним із трьох підходів [71] до формування «вітрин даних» (представлення сховища даних), з яким працює користувач:

1. Підхід на основі даних. Концептуальна схема будується на основі структур джерел даних і дозволяє пропустити складну задачу т. зв. прив'язування елементів джерел даних до цієї схеми для завантаження даних.

2. Підхід на основі вимог. Концептуальна схема будується на основі інформації від користувачів у випадку коли інформації про дані дуже мало або вона недоступна. При цьому від користувачів необхідно отримати інформацію про виміри, факти і міри сховища.

3. Змішаний підхід. Концептуальна схема будується на основі структур джерел даних з застосуванням інформації про сховище, отриманої на етапі аналізу вимог від користувачів.

У якості концептуальної моделі можуть використовуватися різні моделі даних: реляційна, багатовимірна, об’єктно-орієнтована, XML. Огляд моделей для концептуального моделювання («Entity-Relationship», об’єктно-орієнтованої, спеціальних) здійснено у роботі [71].

Зокрема, перевагами моделі «Entity-Relationship» у цій роботі зазначені наступні:

- випробувана на протязі років;
- проектувальники знайомі з нею;
- показала себе гнучкою і достатньо потужньою для пристосування в різних застосуваннях;
- було отримано ряд важливих результатів для цієї моделі ([54], [68],[69]).

Переваги об’єктно-орієнтованої моделі:

- є більш виразними і краще відображають статичні і динамічні властивості реляційних систем;
- надають потужні механізми для вираження вимог і обмежень;
- об’єктно-орієнтованість є одним з домінуючих напрямків в моделюванні даних;
- UML, зокрема, є стандартом, який природно розширюється ([72], [73]).

У роботі також зазначається, що спеціальні моделі компенсують недолік не знайомості для проектувальників за рахунок того, що ці моделі:

- досягають кращої економії нотування;
- як слід підкреслюють особливості багатовимірної моделі;
- є більш інтуїтивними і можуть бути прочитані користувачами, які не є експертами ([66], [73], [74]).

Відповідно до підходів побудови «вітрин даних», концептуальне проектування може здійснюватися у режимах:

- ручному, коли концептуальна модель формується проектувальником повністю вручну;
- автоматизованому, коли «попередня» концептуальна модель будується за метаданими джерел даних і модифікується до остаточної проектувальником;
- автоматичному, коли концептуальна модель формується лише з метаданих джерел даних.

Найбільш ефективним і в той же час гнучким при наявності джерел даних і їх метаданих є автоматизований режим, оскільки, не потрібно затрачати час на введення всіх даних і можна внести при необхідності потрібні корегування. Розглянемо роботи, в яких описується формування «попередньої» концептуальної моделі.

Формування багатовимірної моделі з схем E/R, які використовуються у реляційних БД, описано у роботі [75] та з файлів XML - у роботі [76].

У цих роботах для відповідних моделей описані: визначення фактів, для кожного факту – побудова дерева атрибутів, гілкування та щеплення дерева атрибутів, визначення вимірів, ієрархії фактів та ієрархій.

Формування моделі ME/R, яка є поєднанням реляційного і багатовимірного представлення даних, описано в роботі [67]. Алгоритм, запропонований у роботі, включає створення вузлів-фактів, числових атрибутів для цих вузлів, вимірів дати або часу, вимірів з іншими атрибутами сутностей та рекурсивний огляд відношення сутностей. Для оптимізації схеми автори

рекомендують усунути непотрібні схеми-кандидати, перевірити, чи міри не є атрибутами; визначити необхідний рівень деталізації інформації щодо дати (часу), необхідність інших розрахункових полів, можливість об'єднання схеми, можливість усунення будь-яких полів, необхідність даних, яких немає в БД OLTP. При проектуванні (пост-проектуванні або оптимізації) сховищ даних на основі запитів відбувається аналіз запитів, результати якого можна також використати для визначення метаданих сховища даних.

Формування концептуальної моделі для сховищ даних на основі XML описана в роботі [76]. Визначаються так звані віртуальні виміри та репозиторій фактів. При формуванні цієї моделі використовується UML.

Формування концептуальної моделі на основі об'єктно-орієнтованих моделей описано в роботі «Designing Data Warehouses with OO Conceptual Models» [77]. Цей підхід вводить набір мінімальних обмежень та розширень до UML для відображення багатовимірних властивостей моделювання. У ньому, виділяються так звані класи фактів, пов'язані відношенням 1-до-багатьох з декількома класами вимірів. Цей підхід вводить також так звані похідні міри, що обчислюються за встановленими правилами.

Виходячи з того, що при проектуванні сховищ даних є доступні метадані сховища, а зміни семантичного характеру в модель можуть бути внесені проектувальником за інформацією від користувачів, для проектування МБСД вибрано змішаний підхід (на основі вимог до даних та користувачів).

1.4.2. Логічне проектування сховищ даних

При логічному проектуванні сховищ даних визначаються одиниці даних тих моделей, якими представляються ці дані.

Історично сховища даних поділяються на наступні три види (з точки зору використання моделей для представлення даних):

- реляційні, що використовують лише схеми типу «зірка», «сніжинка» чи «сузір'я» (модель ROLAP);

- багатовимірні із використанням універсальна багатовимірної моделі (модель MOLAP):
- гібридні (модель HOLAP).

При використанні гібридної моделі відбувається розподіл даних між реляційною та багатовимірною моделлю згідно наступних принців [63]:

- щільні частини кубів в MOLAP і розріджені в ROLAP;
- основні куби в ROLAP і додаткові в MOLAP;
- дані, до яких є частий доступ – в MOLAP, решту даних – в ROLAP.

Згідно [65], для створення логічної схеми при розміщенні даних у реляційній моделі потрібно перетворити схему фактів (концептуальну схему) у логічну схему (типу «зірка», типу «сніжинка», типу «сузір’я»).

При схемі типу «зірка» виділяються таблиці вимірів, кожна з яких має первинний ключ і набір атрибутів, які визначають елементи виміру на різних елементах ієрархії, та таблиця фактів, первинним ключем якої є набір атрибутів, на яких визначені зовнішні ключі до всіх таблиць вимірів.

При схемі типу «сніжинка» таблиці вимірів декомпозуються на декілька пов'язаних зовнішніми ключами таблиць вимірів для усунення транзитивних залежностей у цих таблицях.

При схемі типу «сузір’я» є декілька таблиць фактів, що мають спільний набір вимірів, що представляється згідно схем типу «зірка» або «сніжинка».

У ряді робіт описано побудову моделей носіїв (реляційних, багатовимірних та інших) [66],[73], [74], [78].

Так, у роботі [79], логічне проектування СД складається з наступних етапів:

- «очищення» концептуальної схеми за допомогою додавання нефункціональних вимог, в результаті чого отримуємо "очищену концептуальну схему";
- прив'язка очищеної концептуальної схеми до БД джерел;
- генерування реляційної схеми СД у відповідності до «очищеної концептуальної схеми» та БД джерел.

У роботі [80], логічна модель будується проектувальником на основі визначеного набору правил («Merge», «Higher Level Names», «Materialize Calculation», «Materialize External Expression», «Eliminate Non Used Data», «Roll-Up», «Update Key», «Data Constraint»).

Логічне проектування полягає у визначенні моделей носіїв даних за заданою концептуальною моделлю. Оскільки можуть використовуватися різні моделі у носіях даних, розглянемо роботи, шо стосуються побудови цих моделей.

У статті [80] розглядається питання вибору кубів даних OLAP. Для вирішення цієї проблеми використовується так звана решітка кубів. Ця решітка моделює взаємозв'язки між кубами, що можуть бути побудовані з одного й того ж набору вимірів шляхом групування, як показано на рис. 1.4.

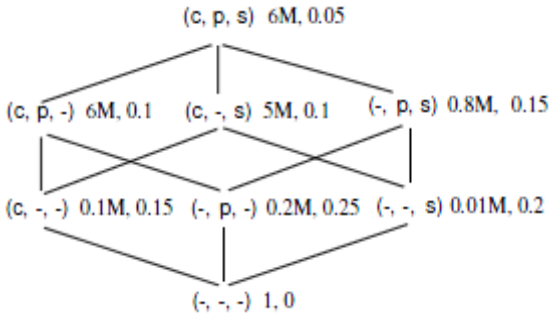


Рис. 1.4. Приклад решітки кубів (взято із статті [80])

У цій роботі використовується генетично-жадібний алгоритм для вибору куба даних OLAP для задачі вибору кубів даних з N вимірами з 2^N можливих. Хромосома складена з одиниць для вибраних кубів і нулів – для не вибраних. За функцію придатності взята функція вартості, яка дорівнює

$$\sum_{i=1}^{n} f_{c_i} E(c_i.M) + g_u \sum_{c \in M} U(c,M)$$

де f_{c_i} - частота звернення до куба і, $E(c_i.M)$ – вартість оцінки куба і при поточній матеріалізації M, g_u - частота вставки в базове відношення, $U(c,M)$ – вартість обслуговування куба і при поточній матеріалізації М. Селекція вибирається шляхом генерації випадкового числа від 0 до 1 і вибору більш придатної хромосоми, якщо це число перевищує 0,75. Схрещування відбувається з

імовірністю (точкою схрещування) 0,7. Мутація відбувається з імовірністю p_m. Крім генетичного алгоритму, використовується жадібний алгоритм для тих рішень, які не є допустимими. Оброблення недопустимих рішень відбувається наступним чином: якщо пара батько-нащадок мають тих самих предків і вони матеріалізовані, тоді треба розматеріалізувати вузол-нащадок, перерахувати загальну вартість і замінити старе рішення новим.

Розв'язок описаної задачі може бути використаним для вибору кубів у багатовимірній базі даних, однак це не вирішує питання побудови всього сховища.

У роботі «X-Warehousing: An XML-Based Approach for Warehousing Complex Data» [81] описується використання моделі XML для представлення даних сховища. Для цього визначаються схеми XML типу «зірка» та «сніжинка» і будується куб XML на основі концептуальної моделі та структур XML-файлів. Єдине дерево атрибутів в такому випадку будується за допомогою гілкування і прищеплювання дерев атрибутів, отриманих з концептуальної моделі та XML-файлів. Для визначення XML-документів, які узгоджуються з концептуальною схемою і будуть завантажені в сховище, вводиться принцип XML-документа мінімального вмісту, згідно з яким такими документами вважаються всі, які містять всі обов'язкові виміри, факти та міри (обов'язковість вказується користувачем у концептуальній моделі).

Для реляційної частини сховищ даних побудова логічної моделі зводиться до двох етапів.

1. Відображення концептуальної моделі сховища у реляційну модель носія даних. Для сховищ, побудованих з реляційної моделі, або ж з концептуальної моделі E/R, це відображення є тривіальним, оскільки існує однозначна відповідність між сутностями концептуальної і відношеннями логічної моделі.

2. Модифікація отриманих структур даних для оптимізації виконання запитів до носіїв даних. Для реляційних СКБД цей процес включає:

- вибір матеріалізованих подань (матеріалізованим представленням є окрема від самого представлення таблиця, що містить результати виконання запиту, визначеного для цього подання, і дозволяє отримати результат запиту з попередньо збережених даних, а не виконувати запит повторно);
- вертикального та горизонтального фрагментування таблиці фактів (фрагментація – це процес поділу таблиць за атрибутами (вертикальна) чи діапазонами значень атрибутів (горизонтальна), що дозволяє оптимізувати виконання запитів за рахунок опущення виконання операцій фільтрування за полями або за умовами на значення атрибутів).

Розглянемо існуючі рішення для вибору матеріалізованих подань та фрагментації сховища.

У статті [82] розглядається проблема вибору схеми горизонтальної фрагментації сховища даних. Кожен атрибут фрагментації може бути представлений масивом з n елементами, де n відповідає числу його піддоменів. Значення цих елементів знаходиться між 1 і *n*. Якщо два елементи мають однакові значення, то вони будуть об'єднані для формування одного. Це кодування може бути використане для представлення фрагментів таблиць вимірів і таблиці фактів. Для селекції використовується метод колеса рулетки, що призначає кожній хромосомі деякий сектор. Хромосома відбирається для створення нащадку, якщо згенероване випадковим чином число знаходиться в межах відповідного їй сектору. Для схрещування вибраних хромосом використовується двоточковий механізм. Значення придатності визначається за допомогою призначення балів по двох показниках: показнику порогу і показнику виконання запитів. Показник порогу визначає, чи перевищує кількість отриманих фрагментів ± 5 відсотків порогу: у разі перевищення будуть призначені 55 балів, в протилежному випадку – менша кількість балів. Показник виконання запитів визначає, чи перевищує вартість запиту встановлене значення, що обчислюється за допомогою функції вартості – при перевищенні цього значення хромосомі призначається менше 3 балів за запит.

Функція вартості визначається наступним чином

$$Cost(Q_k) = \sum_{j=1}^{N} valid(Q_k, S_j) \prod_{i=1}^{M_i} \frac{Sel_F^{p_i} \times \|F\| \times L}{PS},$$

де M_i, F ,L та PS позначають відповідно кількість предикатів вибору, що визначаються фрагментом факту підсхеми типу «зірка» S_j, потужність таблиці фактів (кількість кортежів) F, довжину в байтах кортежу таблиці F та розмір сторінки системи, відповідно, а $valid(Q_k, S_j)$=1, якщо підсхема типу S_j необхідна для Q, і дорівнює 0 в інших випадках.

Загальна вартість виконання набору запитів Q становить:

$$TC(Q) = \sum_{j=1}^{m} Cost(Q_j).$$

Мутація відбувається з нормою 6-30 %. Дане рішення може бути використано для вибору схеми розбивки як реляційної, так і багатовимірної бази даних, а також для вибору матеріалізованих представлень.

У статтях [83..84] також розглядається проблема вибору матеріалізованих представлень шляхом «об'єднання» планів виконання запитів. За хромосоми взято номери планів виконання запитів, об'єднані в бінарний рядок. Функція придатності дорівнює C_{max}-$c(x)$, де $c(x)$ – функція вартості, а $C_{\max}$ – максимальне значення $c(x)$ в популяції або в останніх k поколіннях. Використовується одне точкове схрещування з випадково вибраною точкою схрещування від 1 до n, де n – довжина хромосоми. Мутація відбувається за допомогою інвертування біта від 1 до n, що випадково вибирається в хромосомі. Застосовується турнірна селекція, де порівнюються індивіди, і кращий з них проходить до наступної популяції. Розмір турніру від 4 до 7 індивідів.

Дане рішення також може бути використано для вибору матеріалізованих представлень.

У статті «Multiobjective Genetic Algorithms for Materialized View Selection in OLAP Data Warehouses» [85] автора Michael Lawrence розглядаються багатоцільові алгоритми для вибору матеріалізованих представлень в сховищах даних OLAP.

Проблема вибору представлень може бути формально визначена наступним чином. Для кожного подання v є деяка оцінка кількості записів r_v у цьому поданні. В роботі використовується лінійна модель вартості, де вартість відповіді на запит v є r_v. Вартість $q(v,M)$ відповіді на агреговані запити до представлення v з використанням множини матеріалізованих представлень M дорівнює кількості записів у найменшому представленні в M, яке є предком v у решітці кубів даних, як описано в роботі [78]. Загальний час запиту з використанням M є зважена сума

$$Q(M) = \sum_v f_v q(v, M)$$

Розмір $S(M)$ є сумою розмірів кожного з представлень в M. Обмеження полягає в тому, що $S(M) \leq S_{max}$ для деякого максимального розміру S_{max}. Вартість обслуговування $m(v,M)$ матеріалізованого подання v з M моделюється, базуючись на вартості, присвоєній кожному куту $(v_1;v_2)$ решітки, та відображає вартість обслуговування v_2 з використанням оновлень з v_1. Вартість $m(v,M)$ матеріалізованого подання v є сумою вартостей найдешевшого шляху з матеріалізованого предка цього подання. Кожен вузол v також має частоту оновлення g_v, загальна вартість оновлення для множини матеріалізованих представлень M складає

$$Q(M) = \sum_v g_v m(v, M)$$

У статті «Complete Algorithm for fragmentation in Data warehouse» автори Ziyati Elhoussaine, Driss Aboutajdine, El Qadi Abderrahim розглядають алгоритм фрагментації в сховищах даних [86].

Для кожної схеми фрагментації атрибути представляються масивом з n елементів, де n відповідає кількості значень. Тоді, кожна хромосома (схема фрагментації) представляється багатовимірними масивами (кожен масив представляє атрибут фрагментації). При цьому використовується наступні селекція, схрещування та мутація.

Селекція. Метод колеса рулетки використовується в обох алгоритмах: кожна хромосома асоціюється з її значенням придатності, обчисленим за допомогою моделі вартості, наведеної вище. Хромосома з вищими значеннями придатності має більші шанси бути вибраною.

Схрещування. При горизонтальному та вертикальному розбитті використовується двоточковий механізм, що дає одинакові шанси атрибутам з великою та малою кількостями піддоменів.

Мутація. Мутація необхідна для створення нових генів, що можуть бути не присутні у популяції і дають змогу алгоритму досягти всіх можливих рішень в просторі пошуку.

Запропоновані рішення по автоматизації формування логічних моделей (реляційної, багатовимірної, XML) не описують чіткого механізму визначення моделі, згідно якої будуть представлені дані, та розподілу даних між носіями, що використовують різні моделі даних.

1.4.3. Фізичне проектування сховищ даних

Фізичне проектування сховищ даних за [63] передбачає вибір індексів для БД (об'єктів СУБД, які не змінюють логічну структуру). Індекс - це об’єкт бази даних, який дозволяє отримувати дані з таблиць швидше за допомогою збереження словників даних, які будуються за одним або декількома стовпцями таблиці, для яких визначається індекс. Різні запити можуть вимагати різних індексів, тому найчастіше будується набір індексів для тих запитів, які виконуються частіше або потребують більшої швидкодії.

Вибір індексів описано у багатьох дослідженнях, зокрема у роботі [87] описано вибір індексів, матеріалізованих представлень для покращення реляційних баз даних на основі гібридного генетично-жадібного алгоритму.

Для розв’язку поставленої задачі оптимізації об’єкт (просторові відношення, агрегатні представлення та індекси) представляється масивом k+1 бітів, які об’єднуються в хромосому.

Іншим компонентом фізичної моделі є параметри взаємозв’язку носіїв даних між собою, зокрема розподіл даних між окремими серверами, що забезпечують паралельну роботу сховища даних.

Робота [88] описує фрагментацію багатовимірних баз даних для виконання запитів у паралельному режимі, тобто отримання результату одного запиту при допомозі декількох баз даних, у яких зберігаються частини даних сховища.

У цій роботі, розподіл даних з таблиці фактів і растрових індексів визначається в два етапи. На першому етапі визначається горизонтальна фрагментація багатовимірної таблиці фактів, в результаті чого отримуємо n непересічних фрагментів фактів. Ці фрагменти є одиницями для розміщення на диску, а також для обробки запитів. Фрагментація таблиці факт застосовується до растрових індексів: кожен растр будь-якого бітового індексу розділений на *n* растрових фрагментів. Це гарантує, що біти реєстрового фрагмента відповідають рівно одному фрагменту факту і, таким чином, різні фрагменти фактів можуть оброблятися паралельно.

Другий етап розподілу призначає всі фрагменти на дисках. Як правило, кількість фактичних фрагментів, *N*, набагато більша, ніж число дисків, *d*, для більш ефективного використання обладнання.

Паралельне розміщення даних у сховищах даних OLAP також розглядається у роботі [89]. У цій роботі описано алгоритм, який визначає ресурси сховища (процесорний час і пам'ять) для оброблення таблиць вимірів, а також паралельне природне злиття за схемою «зірка».

Розглядається архітектура «shared-nothing» з *N* процесорами та використовується d-мірне сховище даних, спроектоване механізмом DataIndexing, описаним в роботі. Даний підхід полягає в наступному: N процесорів розподіляється в $d + 1$ (потенційно взаємно невиключних) груп процесорів. Після цього, таблиця вимірів *Di*, розроблена у відповідності зі механізмом DataIndexing, і таблиця фактів *JDI*, що відповідає відповідному значенню ключа *Di,* будуть призначені групі процесорів *i*. Усередині групи

процесорів, гібридна стратегія буде використовуватися для призначення записів окремому процесору.

Проектування сховищ даних, фрагментованих за вузлами (Node Partitioned Data Warehouse, NPDW) розглядається у статті [90].

У цій роботі використовується схема типу «зірка». Таблиці фактів фрагментуються між вузлами, таблиці вимірів дублюються. Паралельне виконання запитів у такій системі полягає у отриманні часткових агрегованих даних з кожного вузла і наступною покроковою агрегацією отриманих часткових результатів.

У роботі визначається т.зв. хеш-фрагментація, при якій враховуються вартості фрагментації, повторної фрагментації, з'єднання з даними, локальної обробки та об'єднання. На основі порівняння вартостей локального зберігання та повторної фрагментації визначається доцільність повторної фрагментації.

Крім того, при фізичному проектуванні повинно бути вирішено питання, на якому рівні проводиться розпаралелювання даних у сховищі – чи на рівні окремих баз даних (підхід Virtual Database), чи на рівні однієї БД (з використанням механізму MapReduce). Розглянемо ці підходи.

Базовою архітектурою сховищ даних з використанням інтеграції даних є так звана "віртуальна база даних" (Virtual Database, ВБД) [91].

Ця архітектура реалізує "єдине подання" для отримання даних з різних джерел, здебільшого гетерогенних. Архітектура ВБД показана на рис 1.5.

Для організації такого уніфікованого доступу до різних джерел даних використовуються наступні модулі:

- обгортка (wrapper), який надає табличне подання джерела із використанням мови SDL (Source Description Language) і відповідає фазі Load;
- прив'язка (mapper), що встановлює відповідність між глобальною схемою віртуальної бази даних та локальною схемою джерела, із використанням мови MDL (Map Description Language) і відповідає фазі Transform;

- екстрактор (extractor), який перетворює неструктуровані дані в структуровані із застосуванням мови EDL (Extract Description Language) і відповідає фазі Transform.

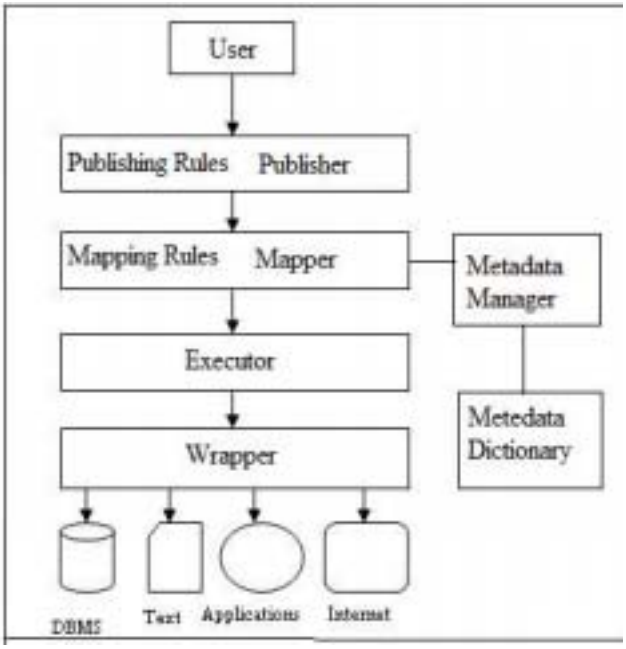


Рис. 1.5. Приклад решітки кубів (взято із [75])

Особливістю такого підходу є те, що інтеграція даних відбувається на логічному рівні, а не на фізичному, що приводить до економії часу на представлення даних у потрібному форматі (глобальна схема) порівняно з класичними сховищами даних.

Крім архітектури класичних СКБД та архітектури ВБД, як локальних рішень зберігання даних, можна виділити системи «Big Data», що спеціалізуються на розподіленому зберіганні і обробці даних.

Особливістю цих систем є розподілення даних на маленькі частини, найчастіше на пари «ключ-значення» з наступною обробкою на великій кількості малопотужних екземплярів, в ролі яких виступають фізичні чи віртуальні машини з встановленими програмами-агентами.

Прикладом системи «Big Data» є програмне забезпечення Hadoop компанії Apache, що використовується у платформі даних HortonWorks, архітектура якої представлена на рис.1.6. При такому підході дані зберігаються у вигляді «файлів» файлової системи HDFS2 і для користувачів можуть представлятися як «реляційною» (SQL-like) базою даних Apache Hive із

використанням мови HiveQL, так і за допомогою noSQL баз даних Hbase або Accumulo.

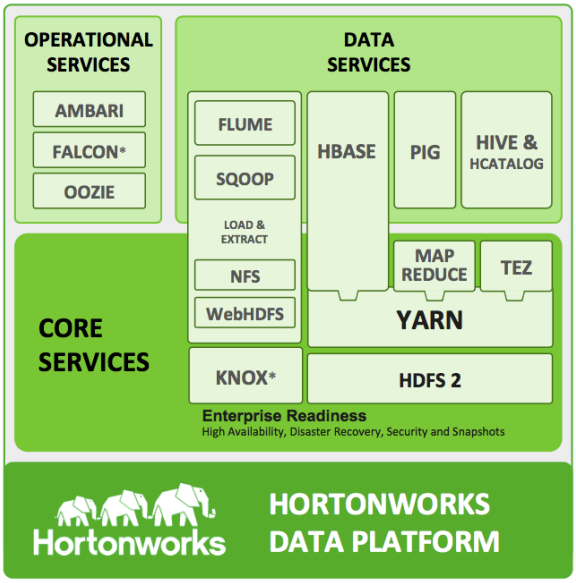


Рис. 1.6. Платформа даних HortonWorks (взято із [92])

Суттєвою особливістю системи Hadoop є використання механізму MapReduce, схема якого показана на рис.1.7.

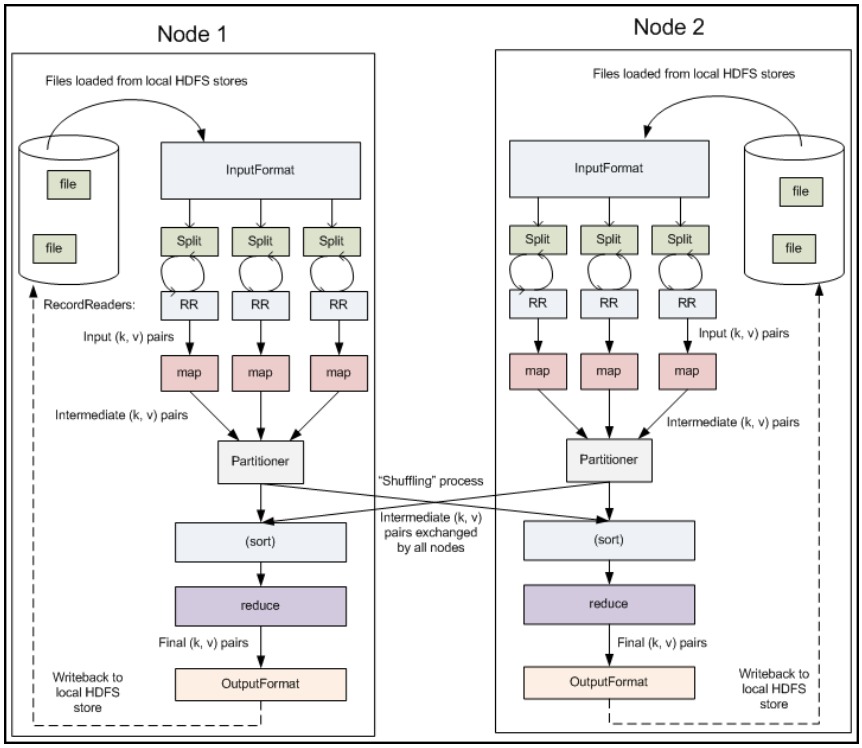


Рис. 1.7. Схема оброблення даних за допомогою MapReduce (взято з [93])

При використанні MapReduce дані обробляються за наступним алгоритмом.

1. Дані беруться з HDFS і перетворюються з формату InputFormat у формат <ключ-значення> (аналог фази Extract).

2. Дані в форматі <ключ-значення> обробляються процедурою map() окремими обробниками (аналог фази Transform).

3. Дані об'єднуються і обробляються процедурою reduce() (аналог фази Transform).

4. Дані перетворюються в формат OutputFormat і зберігаються в HDFS (аналог фази Load).

Якщо порівняти архітектуру Virtual Database і платформу HortonWorks, можна вказати на те, що перевагами HortonWorks є висока швидкодія виконання запитів, а недоліком HortonWorks є забезпечення цілісності окремим застосуванням, що не у всіх випадках може бути прийнятним.

Для використання переваг обох архітектур були запропоновані гібридні рішення, що базуються на використанні MapReduce у архітектурі Virtual Database, наприклад в [94]. У таких рішеннях MapReduce застосовується не до HDFS, а до баз даних. Результат спільного запиту отримується за допомогою виконання операції *reduce()*, що виконується над парами «ключ-значення» (операція map), отриманих з запитів, що виконуються окремими базами даних.

Існуючі рішення фізичного проектування сховищ даних описують питання визначення налаштувань носіїв сховища, таких як індекси таблиць у реляційних базах даних. Крім того, в них розглядається питання паралельного зберігання даних за рахунок як фрагментації даних, так і за рахунок використання різних носіїв у одному сховищі. Однак в них не запропоновано моделі, яка б урегульовувала розподіл даних у сховищі і на основі цього забезпечила б оптимізацію сукупної вартості розміщення даних сховища та виконання запитів до нього. З точки зору інкапсуляції доступу до даних оптимальним є поєднання використання цих підходів: сховище доступне за абстракцією VDB, що дозволяє поєднати різні СКБД для виконання запитів до даних, а окремі дані в рамках баз даних NoSQL можуть горизонтально масштабуватися в рамках відведених їм вузлів.

1.5. Постановка задачі дослідження

У розглянутих роботах виявлено, що існуючі підходи до побудови сховищ даних не розв'язують комплексну задачу побудови РСД ГТ з мінімальною сукупною вартістю збереження та обробки даних із врахуванням інформації про дані та запити до них.

З огляду на це, визначимо мету та завдання дослідження.

Метою дисертаційної роботи є зниження сукупної вартості збереження та обробки даних у розподілених сховищах даних гібридного типу за рахунок нової ІТ, створеної на основі комплексу математичних моделей і методів аналізу та оцінювання процесів опрацювання даних у сховищах зазначеного типу і оптимізації їх структури.

Для досягнення поставленої мети в дисертаційній роботі вирішуються такі **завдання**:

1) аналіз процесу та методів побудови РСД ГТ і проблем, що виникають при цьому;

2) розроблення методу побудови РСД ГТ з мінімальною сукупною вартістю збереження та обробки даних при обмеженнях на час виконання запитів і кількість реплікованих копій даних;

3) розроблення математичних моделей багаторівневого описання РСД ГТ та їх відображень і взаємодії;

4) розроблення моделей та методів логічного і фізичного розподілу даних у РСД ГТ і дослідження ефективності та збіжності розроблених методів;

5) створення ІТ побудови РСД ГТ на основі розроблених моделей, методів та засобів;

6) експериментальне дослідження створеної ІТ побудови РСД ГТ.

Для вирішення цих завдань сформулюємо концпцію побудови РСД ГТ.

1.6. Концепція побудови розподілених сховищ даних гібридного типу як багаторівневої системи зберігання даних

Побудова систем зберігання даних є ітеративно-послідовним процесом реалізації міжрівневих відображень описів систем. Для цього задаються моделі

кожного рівня опису, моделі міжрівневих відображень елементів рівнів, концепції множинного вибору розподілу даних між вузлами та маршруту реплікації в вигляді відповідних цільових функцій та обмежень на час виконання запитів та забезпечення доступності сховища.

Характерною особливістю підходу до проектування РСД ГТ є те, що побудова таких сховищ здійснюється з існуючих і попередньо визначених джерел даних, тобто перетворення рівнів опису сховища починаються з відомих (за метаданими) логічних моделей джерел даних, між елементами яких та елементами концептуальної моделі сховища задається взаємовідповідність (рис.1.8.).

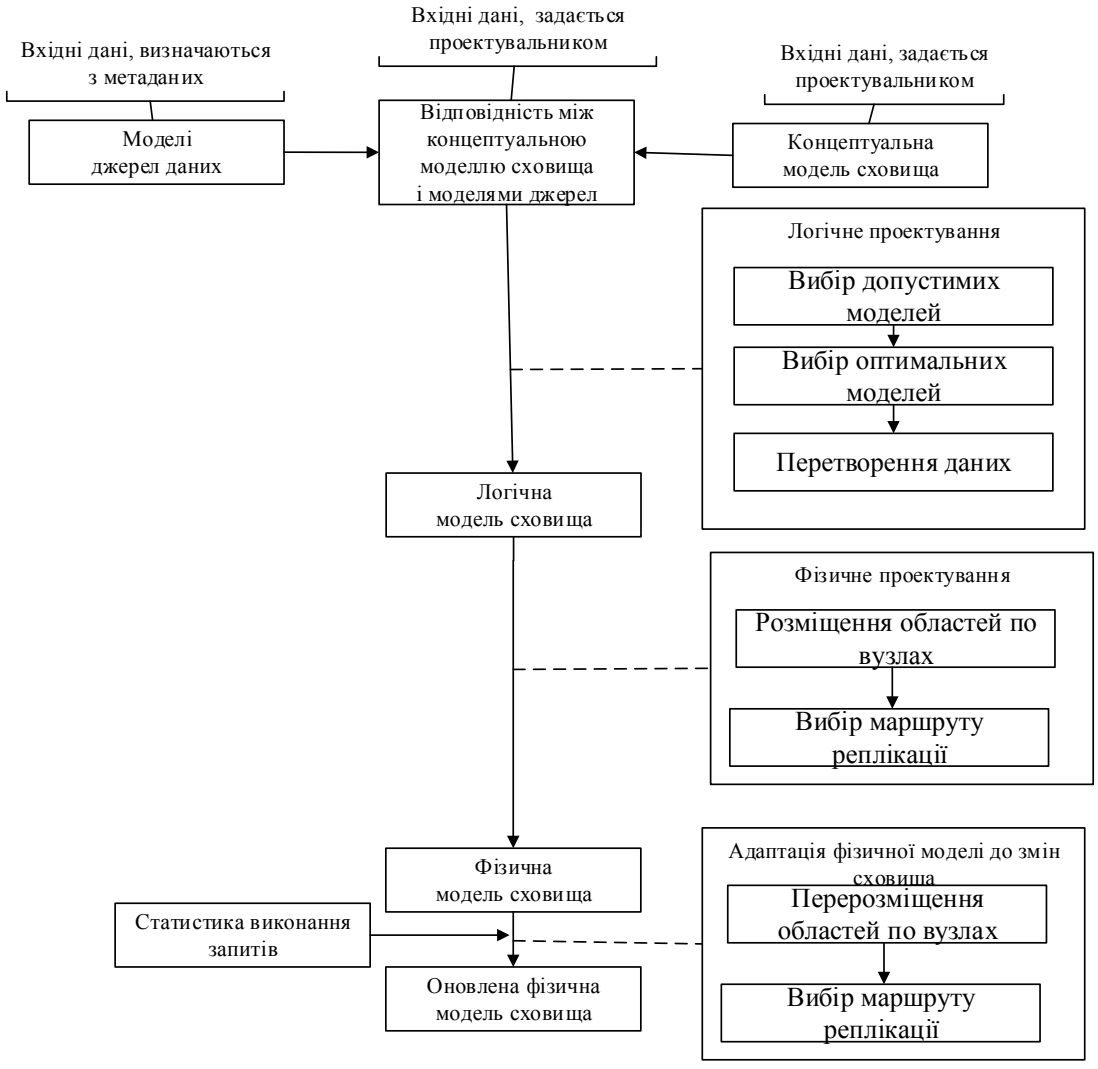


Рис. 1.8. Рівні опису розподілених сховищ даних гібридного типу з відповідними міжрівневими відображеннями [Розроблено автором]

В таблиці 1.1. описано моделі джерел даних, що відображають структуру даних джерел, представляються елементами даних, їх атрибутами і відношення між ними.

Таблиця 1.1.

Моделі джерел даних та міжрівневі перетворення. [Розроблено автором]

Модель джерел даних	**Елемент**	**Атрибути**	**Відношення**
Реляційна	Відношення	Атрибути відношень	Первинні та зовнішні ключі
XML	Теги	Атрибути тегів	Ієрархія тегів
Багатовимірна	Куби	Виміри, міри	Ієрархії в вимірах, визначення мір на вимірах
NoSQL	Документи	Поля даних	Порядку, включеності
Об’єктно-орієнтована	Об’єкти	Члени	Включеності
RDF	Суб’єкти	Значення предикатів	-
Структурований текст	Текст	Стовпці	-

Побудова концептуальної моделі. Концептуальна модель сховища відображає наявні у сховищі сутності, які задаються їх атрибутами, та пов’язані визначеними відношеннями (модель Entity-Relationship). За зв’язками у сховищі формуються так звані області – підмножини множини сутностей, у кожній з яких зв’язки формують зв’язний граф.

Після задання концептуальної моделі проектувальником сховища задаються відношення між сутностями сховища і елементами даних джерел, які показують, які елементи даних у відповідних моделях джерел відповідають яким сутностям і які зв’язки яким відношенням. На основі цієї взаємовідповідності, будується логічна модель сховища.

Побудова логічної моделі. На основі заданих моделей джерел, концептуальної моделі сховища та відношення між ними будується логічна модель, що відображає елементи баз даних, які відповідають сутностям з концептуальної моделі. Оскільки, РСД ГТ не лише зберігають дані

абстраговано, але і дані можуть бути представлені різними моделями, то логічна модель таких сховищ даних складається з взаємовідповідностей між сутностями, атрибутами сутностей і відношеннями концептуальної моделі та елементами, атрибутами елементів та зв'язками даних з вказанням моделі, яка використовується.

Вибір допустимих моделей. Для того, щоб вибрати моделі для збереження даних, спочатку знаходяться ті моделі даних, при використанні яких виконуються умови швидкодії та доступності сховища. Ці моделі визначаються на основі аналізу даних.

Вибір оптимальних моделей. У РСД ГТ дані розміщуються у паралельних носіях з різними моделями даних. Вибір моделей представлення даних тих чи інших сутностей проводиться на основі критерію оптимальної вартості з тих моделей, при використанні яких виконуються умови до швидкодії та доступності сховища.

Застосування цього критерію означає мінімізацію вартості перетворення даних у вибрані моделі та виконання запитів до даних у цих моделях при обмеженні на час виконання запитів.

Перетворення даних. За розв'язком задачі будуємо множини вибраних моделей для розміщення сутностей. Далі відповідно до цих моделей інтегруємо у сховище дані. При співпадінні вибраної моделі з тією, у якій дані представлені в носії, використовується це джерело, інакше дані перетворюються до відповідних моделей носіїв даних.

Побудова фізичної моделі. Після визначення елементів логічної моделі можна розгорнути сховище на одному вузлі. Для цього визначається так звані конфігурації розгортання та реплікації, які задають фізичну модель сховища. Конфігурація розгортання описує розміщення областей сховища по вузлах (які області розміщуються у яких вузлах), а конфігурація реплікації описує маршрут реплікації даних між вузлами (які області реплікуються через які вузли).

Розміщення даних по вузлах отримується внаслідок розв'язку задачі мінімізації вартості (розгортання сховища та виконання запитів) при

розподіленні навантаження між декількома вузлами для забезпечення виконання заданої кількості запитів при часі їх виконання не вище граничного та забезпеченою доступністю областей. З метою визначення значення критерію та перевірки виконання обмежень на часу виконання та забезпечення доступності генеруються тестові запити або ж використовується зібрана статистика.

Вибір маршруту реплікації. Після розміщення даних областей у вибраних вузлах потрібно для кожної області визначити маршрут реплікації, тобто задати, між якими вузлами здійснюється реплікація – за критерієм мінімізації сумарного часу реплікації згідно з цим маршрутом при обмеженнях на сумарне по всіх областях використання ресурсів вузлів на реплікацію даних.

Після розв’язку цих задач розміщення даних по вузлах та вибору маршруту реплікації відбувається фізична інтеграція даних з джерел в носії сховища, після якої сховище починає виконувати запити і збирається їх статистика, яка включає інформацію про те, які запити виконувалися і скільки часу виконувався кожний запит.

Адаптація фізичної моделі до змін сховища. Якщо згідно отриманої статистики запити виконуються довше граничного значення, або доступність областей зменшується нижче допустимої, то необхідно провести повторне логічне і фізичне проектування сховища на основі існуючої статистики виконаних запитів.

Запропонована концепція побудови РСД ГТ як багаторівневої системи зберігання та обробки даних може бути реалізована за допомогою створення нової ІТ побудови РСД ГТ на основі певних моделей, методів та засобів. Визначимо сукупність вимог до цієї ІТ.

Так, інформаційна технологія побудови РСД ГТ повинна:

– виконувати запити користувачів, транслюючи ці запити до носіїв сховища;

– здійснювати аналіз даних, зокрема їх структурованості;

– здійснювати логічний і фізичний розподіл даних, на основі властивостей даних і статистики запитів тощо.

Технічними вимогами до програмного забезпечення, що реалізує розроблену ІТ є підтримка:

– 64-розрядних обчислень;

– основних клієнтських систем (Windows, UNIX, Mac OS);

– розподіленого збереження та обробки даних.

Програмні вимоги до програмного забезпечення, що реалізує розроблену ІТ полягають в підтримці:

– розширюваності (можливості підключення інших СКБД);

– доступу інших програмних засобів до функцій цього ПЗ через програмний інтерфейс;

– взаємодію інших програмних засобів з цим ПЗ через веб-інтерфейс (протоколи HTTP/S).

Вимоги до ефективності роботи ІТ. Застосування розробленої ІТ повинно забезпечити:

– зменшення сукупної вартості зберігання та обробки розподілених даних гібридного типу щонайменше на 10% у порівнянні з інформаційною системою, яка не використовує дану ІТ;

– виконання запитів за час нижче s/1000, де s – обсяг даних, що обробляється за запитом до сховища;

– резервування даних сховища у кількості не менше двох доступних копій.

Висновки до розділу 1

На основі проведеного аналізу підходів, моделей, методів та засобів побудови сховищ даних зроблено висновок щодо відсутності комплексних досліджень, які би враховували властивостей джерел даних, структуру РСД ГТ та можливість підвищення ефективності роботи сховища в процесі його експлуатації. Таким чином, невирішеною залишається задача побудови ефективних РСД ГТ, яка зумовила актуальність науково-прикладного завдання

створення ІТ побудови сховищ зазначеного типу та вимагає концептуально-методологічних інновацій у цій сфері на основі проведення ґрунтовних досліджень.

Виходячи з цього, у розділі сформульовано завдання дослідження і запропонована концепція побудови РСД ГТ, яка за рахунок розроблення та використання нової ІТ, створеної на основі комплексу математичних моделей і методів аналізу та оцінювання процесів опрацювання даних у сховищах зазначеного типу і оптимізації їх структури, може забезпечити зниження сукупної вартості збереження та обробки даних у РСД ГТ.

Результати цього розділу опубліковані в роботах [49..50, 95..96].

РОЗДІЛ 2. МУЛЬТИБАЗОВІ СХОВИЩА ДАНИХ

У даному розділі вводиться описове та формальне визначення мультибазових сховищ даних, описано моделі даних сховища, а також міжрівневі переходи між моделями.

2.1. Описове визначення МБСД

Визначимо мультибазове сховище даних наступним чином.

Визначення 2.1. Мультибазове сховище даних (МБСД) - це РСД ГТ, побудоване за трьома рівнями абстракції даних – рівень вузлів сховища, рівень носіїв сховища та рівень даних, наступним чином (рис. 2.1.).

1. *Рівень вузлів сховища*. Зберігання та обробка даних сховища здійснюється в окремих апаратно-програмних одиницях, що називаються **вузлами**. Кожен вузол має свою систему керування, включаючи систему доступу до даних, розміщених у цьому вузлі. Вузли поділяються на **центральні** та **регіональні**. Служби доступу центральних вузлів відповідають на всі запити до даних сховища, включаючи до даних, які не розміщені в цих вузлах, в той час як служби доступу до даних регіональних вузлів відповідають лише на запити до тих даних, які містяться в відповідних вузлах.

2. *Рівень носіїв сховища.* Кожен вузол, як центральний, так і регіональний, складається з **носіїв основних** та **допоміжних** даних, а також пов’язаних з ними **джерел** даних. Основні носії даних, а саме реляційні бази даних і бази даних NoSQL зберігають всі дані вузла відповідно до їх структурованості. Ці вузли призначені для OLTP-обробки даних. Допоміжні носії даних, а саме багатовимірні бази даних і бази даних XML, зберігають ті дані, запити до яких оптимізуються за рахунок використання відповідних моделей даних. Допоміжні вузли призначені для OLAP-обробки даних. За допомогою основних та допоміжних баз даних формується адаптивне подання для доступу до даних сховища, яке дозволяє звертатися до даних єдиним способом незалежно від їх розташування і формату зберігання. З джерел даних завантажуються дані в носії сховища, і після завантаження можливо виконувати запити до джерел даних, користуючись сховищем даних, однак ці дані є

статичним поданням, тобто не підлягають синхронізації даних між носіями та вузлами, а запити, які виконуються до цих даних, не оптимізуються.

3. *Рівень даних* передбачає завантаження в носії сховища з джерел, які можуть обробляти запити до даних і які не беруть участі в розподілі даних.

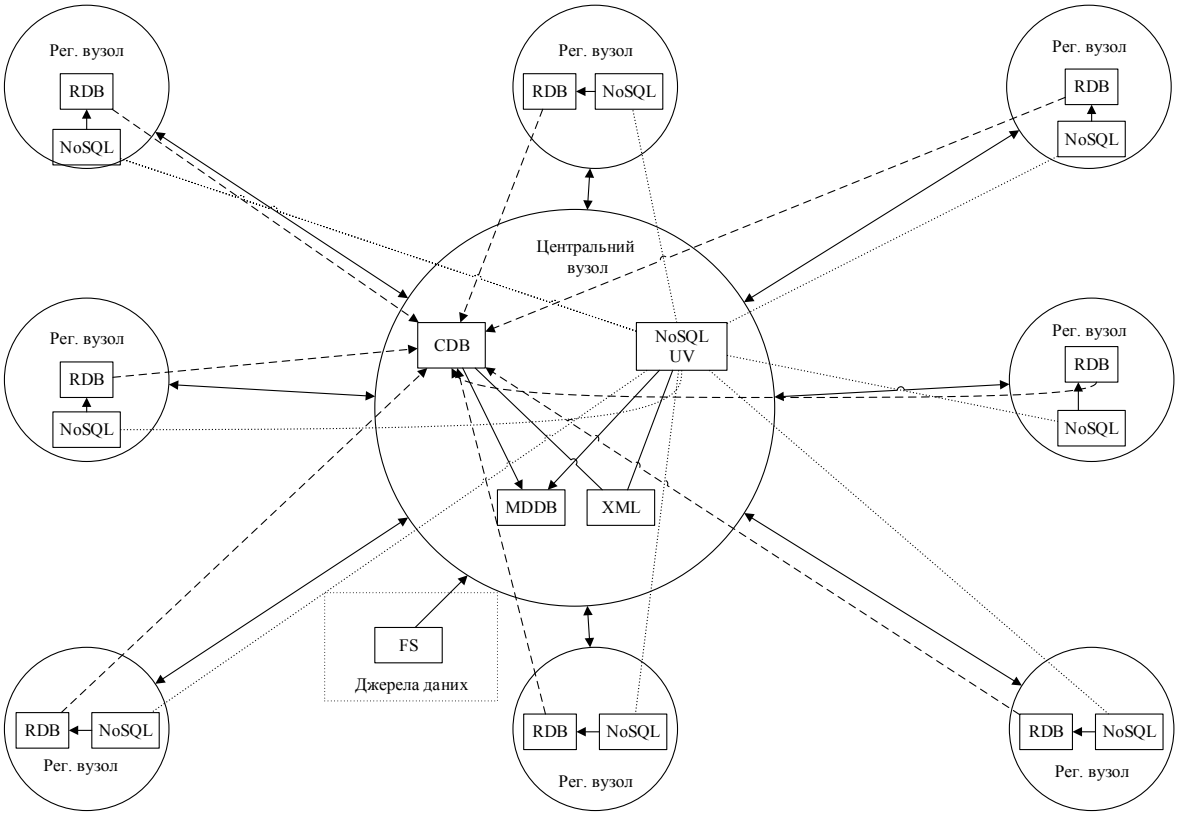


Рис. 2.1. Архітектура мультибазового сховища даних.
[Розроблено автором]

На рис. 2.1. зображені компоненти мультибазового сховища даних, а саме:

- центральний вузол 0, що включає центральну реляційну базу даних CDB, подання уніфікованого доступу до територіально розподіленої мережі баз даних NoSQL UV, багатовимірної БД MDDB, БД XML і файлового сховища FS;
- регіональні вузли 1..n з базами даних РБД та екземпляри БД NoSQL.

У кожному з вузлів можуть бути присутні чи відсутні окремі бази даних, в залежності від варіанту розгортання, який визначається під час оптимізації сховища.

Виходячи з такої трьохрівневої структури сховища при побудові МБСД для забезпечення його оптимального функціонування у цілому пропонується

поєднати підхід побудови на основі структурованості джерел даних (для визначення розподілу даних між носіями) з підходом оптимізації на базі статистики запитів до сховища даних (для розподілу навантаження між вузлами).

Характерною особливістю МБСД є доступність даних через «адаптивне подання» (представлення, розташування і реплікація відповідної частини даних залежить від структур цих даних та запитів, що до них виконуються), яке забезпечують основні та допоміжні носії, та «статичне подання» (представлення та розташування відповідної частини даних не змінюється), яке забезпечують-джерела даних.

МБСД реалізує дві з трьох основних абстракцій доступу до даних:

- медіатор, який отримує даних як з носіїв, так і з джерел даних, та дозволяє отримувати дані уніфіковано, однак не обов'язково одинаковим чином;
- сховище даних, яке надає уніфікований доступ до носіїв даних, в яких дані розміщуються у вибраних носіях даних в залежності від властивостей цих даних і запитів, що виконуються до цих даних.

Такий підхід відрізняється від існуючих наступним чином:

- мультибазові системи, зокрема віртуальна база даних – МБСД мають не лише медіаторну частину, на якій побудовані мультибазові системи, але і частину сховища даних;
- федеративні бази даних – схеми БД, що входять у МБСД, не залежні одна від одної;
- сховища даних – МБСД відрізняються наявністю зовнішнього медіатора з доступом до частини, яка не включена в глобальну схему;
- медіатори – МБСД відрізняються наявністю як частини, включену в глобальну схему (інформація якої інтегрована в носіях даних), так і не включену, інформація якої розміщена в джерелах даних як для інтеграції в носії даних, так і для безпосереднього виконання запитів.

Таким чином, МБСД може розглядатися як комбінація підходів медіатора і сховищ даних для отримання рівня абстракції даних.

Однак, мультибазові сховища даних передбачають розподілене розміщення даних по так званих вузлах, які містять двигуни систем керування базами даних і агенти МБСД для трансляції запитів, виконання операцій рівня МБСД, контролю цілісності даних (де її можливо забезпечити).

Отже, мультибазові сховища даних є більш вузьким поняттям, ніж мультибазові системи, і відрізняються іншим видом абстракції даних (поєднання медіатора і сховища даних) та способом розподілу даних (за вузлами та за СКБД, які використовуються).

2.2. Формальне визначення моделей МБСД

Введемо необхідні визначення.

Визначення 2.2. *Атрибутом даних b* МБСД називається іменована одиниця даних, яка задається його назвою *n* та доменом (можливим діапазоном значень) значень $d \subseteq type$, де *type* – тип даних.

Визначення 2.3. *Сутністю* МБСД *e* називається іменований набір відношення

$$e_i = \{b_1,...b_j,..b_m\} \in E_{MDW}.$$

Визначення 2.4. *Відношенням в МБСД* називається кортеж розміру 2

$$< b_i, b_j > \in R_{MDW}$$

Якщо $b_i = b_j$, такий зв'язок називається первинним. Атрибут, на якому задано первинний зв'язок, має унікальні значення у кожному кортежі.

Якщо $b_i \neq b_j$, таке обмеження називається зовнішнім ключем. Воно зв'язує два різні атрибути однієї сутності або два стопці однієї сутності, причому перший стовпець є первинним ключем, а другий – приймає свої значення з множини значень першого.

Визначення 2.5. Сутності e_1 та e_2 називаються *зв'язаними* ($e_1 \Lambda e_2$), якщо між їх стовпцями існує обмеження цілісності

$$a_1 \in e_1, a_2 \in e_2,$$
$$< a_1, e_2 > \in L$$

За зв'язками у сховищі формуються так звані **області МБСД** – підмножини множини сутностей, у кожній з яких зв'язки формують зв'язний граф, тобто:

$$A = \{a_i \mid a_i = \{r_z\}, \forall t_j \in a_i \exists t_k \in a_i (e_j \Lambda e_k),\}, i = \overline{1,n};$$
$$\forall t_j, t_k \in a_i \exists \{t_l\} \subset a_i (e_j \Lambda e_{l_1}, e_{l_1} \Lambda e_{l_2}, ..., e_{l_2} \Lambda e_{l_n}, e_{l_n} \Lambda e_k).$$

Опишемо МБСД у термінах концепції мультибазових сховищ даних як багаторівневої системи зберігання даних.

Згідно концепції, побудова МБСД, як РСД ГТ складається з таких етапів:

1. концептуального – на цьому етапі здійснюється задання метаданих джерел у термінах їх моделей, визначення сутностей, зв'язків сутностей і областей сховища, аналіз структурованості даних та виділення структурованих і слабко-структурованих даних;

2. логічного, що поділяється на підетапи вибору комбінації моделей для представлення даних, при якій виконуються умови до інформаційної технології та досягається якісне значення критерію вартості і перетворення даних до вибраних моделей;

3. фізичного, що поділяється на підетапи вибору розташування та маршруту реплікації областей сховища по його вузлах.

Розглянемо моделі даних, які використовуються при побудові сховищ даних.

2.3. Моделі джерел даних

Моделі джерел відображають структуру даних джерел і взаємозв'язки між ними. Всі джерела даних в метаописі сховища представляються наступними відношеннями:

$\{< m, \Omega^m >\}$ - перелік всіх моделей даних, які можуть використовуватись у конкретному сховищі, з операціями, що підтримуються даним сховищем;

$\{< m^{Src}, e^{Src} >\}$ - перелік елементів даних джерела з вказанням моделі даних;

$\{< e^{Src}, b^{Src} >\}$ - перелік атрибутів елементів даних;

$\{< b_1^{Src}, b_2^{Src} >\}$ - перелік зв'язків між атрибутами.

Опишемо у цих термінах наступні моделі даних:

- реляційна модель даних (реляційні бази даних, структурований текст);
- багатовимірна модель даних (багатовимірні бази даних);
- модель XML, що є узагальненням ієрархічної та мережної моделей (бази даних XML);
- модель NoSQL (бази NoSQL);
- об'єктно-орієнтована модель даних (об'єктно-орієнтовані бази даних);
- модель RDF.

Задамо моделі, що підтримуються сховищем, визначенням елементів носія, операцій над елементами носія та відношень між елементами носія.

Реляційна база даних – це алгебрична система, у якої носієм є множина реляційних відношень ***r***, множиною операцій – реляційна алгебра $\Re = \langle \pi, \sigma, \triangleright\triangleleft, \cup, \cap, - \rangle$, множиною відношень – множина обмежень цілісності ***Z***, тобто $RDB =< r, \Re, Z >$.

Відношення визначаються як впорядкований набір атрибутів (іменованих сутностей, які задаються їх назвами та доменами значень) відношення $RR = \{(r_1, r_2, ..., r_n)\}$, де *RR* – відношення, а r_i - атрибути відношення.

Множина *K* називається (первинним) ключем відношення *RR*, якщо для множини $K \subseteq R$ виконується $t_1(K) \neq t_2(K)$ і ця нерівність не виконується для будь-якої множини $K' \subseteq K$. Атрибути, що належать ключу, називаються ключовими.

Множина *F* називається зовнішнім ключем відношення RQ, якщо $\forall a \in F \; \forall t(a) \in u(a)$, де $u(a)$ - значення ключового атрибута *a* у відношенні *RR*.

Якщо два відношення $RR_1 = (r_1, r_2, \ldots r_1, r_{l+1}, \ldots r_{l+p})$ та $RR_2 = (r_l, r_{l+1}, \ldots r_{l+p}, \ldots r_{l+k})$ мають спільні атрибути $r_1, r_{l+1}, \ldots r_{l+p}$,то в одному з відношень цей набір атрибутів є первинним ключем і приймає унікальні значення, а в іншому він є зовнішнім ключем. Це відображається в схемі даних БД *R*.

Багатовимірна модель даних є підсистемою алгебраїчної системи класу реляційна модель, у якої носієм є множина кубів *c* як множина реляційних відношень, алгеброю є $\Re = \langle зріз, обертання, згортка, деталізація \rangle$, а в множину відношень *R* входять зв'язки між вимірами, їх елементами ієрархії та мірами

$$MDDB = < c, \Re, R > .$$

Структура даних БМД задається набором атрибутів, які називаються вимірами $D = \{d_i \mid i = \overline{1,n}\}$ та мірами $M = \{m_i \mid i = \overline{1,n}\}$. До складу відношень моделі входять відношення метаданих вигляду $RD = \{< d_i, a_{i1}, a_{i2}, \ldots, a_{in} >\}$, які містять як службові атрибути (назва, батьківський елемент, рівень ієрархії), так і довільні атрибути та відношення фактів $RM = \{< d_1, d_2, \ldots d_n, m_1, \ldots m_n >\}$. Ці відношення складають множину $R = RD \cup RM$.

У випадку *структурованих файлів XML* також може бути застосована реляційна модель – дані задаються за допомогою кортежів даних (тегів) X, схеми XML S, яка відображає зв'язки між тегами, та визначених операцій $\Re$

$$XMLDB = < V, \Re, \mathrm{E} > ,$$

де $V = V_{element} \cup V_{value}$,

V_{elemer}- вершини-елементи (теги),

V_{valu}- вершини даних (атрибути та значення в тегах),

$\Re = \langle \pi, \sigma, \triangleright\triangleleft, \cup, \cap, \chi, \delta, \Sigma \rangle$,

$$E = E_E \cup E_A \cup E_R \cup E_D ,$$

E_E - зв'язки «елемент»- «елемент»,

E_A - зв'язки «елемент»- «атрибут»,

E_R - зв'язки «елемент»- «посилання»,

E_D - зв'язки «елемент»- «дані».

Також цією моделлю можуть бути представлені ієрархічна та мережева моделі.

Окремо варто виділити моделі даних, що використовуються в базах даних NoSQL. Системи NoSQL використовуються для скорочення часу виконання окремих типів запитів (зокрема до великих обсягів даних) за рахунок відходу від реляційної моделі, детипізованого зберігання даних та підтримки цілісності на рівні BASE, а не ACID.

Модель даних «ключ-значення». *Найпростішою формою баз даних NoSQL є сховище пар «*ключ-значення». У сховищі не зберігаються метадані цих даних, тому неможливо виконувати більш складні операції, ніж додавання або видалення та запис.

На фізичному рівні модель «ключ-значення» описується кортежами вигляду:

$$\{< k, v >\},$$

де k- ключ, який приймає унікальні значення у кожній парі, v - значення, що відповідає цьому ключу.

Підтримуються операції :

$$\Re = \langle \pi, \sigma \rangle,$$

де π - вибір атрибутів (ключ або значення), σ - фільтрація атрибутів (вибір значення за ключем, ключів за значенням), $\triangleright\triangleleft$ - злиття (підтримується лише для вкладених документів).

Ця модель може бути зведена до моделі XML наступним чином.

Множиною вузлів-елементів є

$$V_{element} = \{S\} \cup \{P_i \mid i = \overline{1,n}\} \cup \{K, V\},$$

де S – набір пар $P_{1..}P_n$, K – атрибут «ключ», V – атрибут «значення».

Множина вузлів-значень визначається як

$$V_{value} = \{k_i \mid i = 1, n, \forall i \forall j \; i \neq j \Rightarrow k_i \neq k_j\} \cup \{v_i \mid i = \overline{1,n}\}$$

Для побудови ієрархічної структури визначимо зв'язки між вузлами.

Зв'язки «елемент-елемент» визначаються наступним чином:

$$E_E = \{< S, P_i > | \, i = \overline{1,n}\}.$$

Зв'язки «елемент-атрибут» визначаються наступним чином:

$$E_A = \{< P_i, K > | \, i = \overline{1,n}\} \cup \{< P_i, V > | \, i = \overline{1,n}\}.$$

Зв'язки «елемент-дані» визначаються наступним чином:

$$E_D = \{< K, k_i > | \, i = \overline{1,n}\} \cup \{< V, v_i > | \, i = \overline{1,n}\}.$$

Для структурованого зберігання різних властивостей даних у системах NoSQL вводяться сховища типу «ідентифікатор – структура» і «ідентифікатор-документ».

У сховищах типу «ідентифікатор – структура» в якості значення може зберігатися не лише атомарне (число, рядок, дата), але і структура (списки, множини). Додаються операції додавання/видалення елементів списку і множин. Такі сховища можуть бути описані моделлю «ключ-значення».

Сховища типу «ідентифікатор-документ» забезпечують структуроване зберігання властивостей даних. У кожному елементі даних, що називається документом, визначаються атрибути (властивості) та задаються їх значення.

При цьому весь документ зберігається в деякому форматі (JSON, XML або інші), що дозволяє оперувати документами як цілим, але при потребі отримувати значення атрибутів.

Підтримуються операції

$$\Re = \langle \pi, \sigma, \triangleright\triangleleft \rangle,$$

де π - вибір атрибутів, σ - фільтрація атрибутів, $\triangleright\triangleleft$ - злиття (підтримується лише для вкладених документів)

Модель *«ідентифікатор-документ»* описується кортежами вигляду:

$$\{< f_0, < f_1 : v_1, f_2 : v_2, f_n : v_n, f_{n+1} : d_1, f_2 : d_2, f_{n+l} : d_l >\},$$

де f_0- ідентифікатор документу, $f_1..f_m$-атрибути документу, $v_1..v_m$-атомарні значення атрибутів $f_1..f_m$, $d_1..d_l$ - посилання на інші документи.

Ця модель може бути зведена до моделі XML з наступним змістом:

Множиною вузлів-елементів є

$$V_{element} = \{C\} \cup \{D_i \mid i = \overline{1,n}\} \cup \{f_{ij} \mid i = \overline{0,n+l}\},$$

де C – колекція документів $D_1..D_n$, f_{ij} -атрибути (f_{i0} -ідентифікатор, $f_{i(1..m)}$ - атомарні, $f_{i(m+1).i(n+l)}$ (посилання) документів $D_1..D_n$.

Множиною вузлів-значень визначається як

$$V_{value} = \{v_{ij} \mid i = \overline{1,n}, j = \overline{0,m+l}\},$$

де v_{ij}- значення атрибутів f_{ij} .

Зв'язки задаються наступним чином.

Зв'язки «елемент-елемент» визначаються наступним чином:

$$E_E = \{< C, D_i > \mid i = \overline{1,n}\}$$

Зв'язки «елемент-атрибут» визначаються наступним чином:

$$E_A = \{< D_i, f_{ij} > \mid i = \overline{1,n}, j = \overline{0,m+l}\}$$

Зв'язки «елемент-посилання» визначаються наступним чином:

$$E_R = \{< f_{ij}, D_k > \mid i = \overline{1,n}, j = \overline{m+1,m+l}, k \in \{1,n\}\}$$

Зв'язки «елемент-дані» визначаються наступним чином:

$$E_D = \{< f_{ij}, v_{ij} > \mid i = \overline{1,n}, j = \overline{0,m+l}\}$$

Для версійного розподіленого зберігання великих обсягів даних була спроектована модель, що використовується у системі BigTable компанії Google.

У цій моделі ключ ідентифікує рядок, який містить дані, що зберігаються в одному чи декількох сімействах стовпчиків. В рамках таких сімейств кожен рядок може мати багато значень стовпчиків, за допомогою вказання мітки часу.

На фізичному рівні модель системи Bigtable описуються кортежами вигляду

<$r,t,k_1:v_1, ...,k_n:v_n$>, де k – ключ, v- значення.

Підтримуються операції $\Re = \langle \pi, \sigma, \triangleright\triangleleft \rangle$.

Ця модель може бути зведена до моделі XML з наступним змістом:

множиною вузлів-елементів є

$$V_{element} = \{C\} \cup \{D_i \mid i = \overline{1,n}\} \cup \{f_{ij} \mid i = \overline{0,n+l}\}$$

де C – колекція документів $D_1..D_n$, f_{ij} -атрибути документів, зокрема f_{i0} - ідентифікатор рядка, f_{i1} - мітка часу, $f_{i2} .. f_{i(m+1)}$ - атрибути даних.

Множина вузлів-значень визначається як

$$V_{value} = \{v_{ij} \mid i = \overline{1,n}, j = \overline{0,m}+1\},$$

де v_{ij} - значення атрибутів f_{ij}.

Зв'язки задаються наступним чином:

Зв'язки «елемент-елемент» визначаються наступним чином:

$$E_E = \{< C, D_i > \mid i = \overline{1,n}\}.$$

Зв'язки «елемент-атрибут» визначаються наступним чином:

$$E_A = \{< D_i, f_{ij} > \mid i = \overline{1,n}, j = \overline{0,m+1}\}.$$

Зв'язки «елемент-дані» визначаються наступним чином:

$$E_D = \{< f_{ij}, v_{ij} > \mid i = \overline{1,n}, j = \overline{0,m+1}\}.$$

У *БД NoSQL на основі графів* дані відповідають як вершинам деякого графу, так і напрямленим дугам між ними. На фізичному рівні модель БД на основі графів описується наборами «ключ – список ключів» і «ключ-дані».

Підтримуються операції

$$\Re = \langle \pi, \sigma \rangle,$$

де π - отримання даних (з вершини чи дуги), σ - навігація від вершини до дуг і з дуги до вершин

$$\{< id_i, < id_1, id_2, ..., id_n >\} \text{ і } \{< id_i, < v_1, v_2, ..., v_m >\},$$

де id_i - ідентифікатори вершин, $v_1 .. v_m$ - дані вершин.

Ця модель може бути зведена до моделі XML з наступним змістом:

$$V_{element} = \{G\} \cup \{V_i \mid i = \overline{1,n}\} \cup \{f_{ij} \mid i = \overline{1,n}, j = \overline{1,m}\} \cup \{r_{ij} \mid i = \overline{1,n}, j = \overline{1,l}\},$$

де V_i – вузли графу G, f_{ij} – атрибути даних, що зберігаються у вузлах V_i, r_{ij} – посилання на інші вузли.

Множина вузлів-значень визначається як

$$V_{value} = \{v_{ij} \mid i = \overline{1,n}, j = \overline{0,m}\} \cup \{l_{ij} \mid i = \overline{1,n}, j = \overline{1,l}\}$$

де v_{ij} - значення атрибутів f_{ij}, l_{ij} -значення посилань на інші вузли r_{ij}.

Зв'язки задаються наступним чином.

Зв'язки «елемент-елемент» визначаються наступним чином:

$$E_E = \{< G, V_i > \mid i = \overline{1,n}\}$$

Зв'язки «елемент-атрибут» визначаються наступним чином:

$$E_A = \{< D_i, f_{ij} > | \, i = \overline{1,n}, \, j = \overline{1,m}\} \cup \{< D_i, r_{ij} > | \, i = \overline{1,n}, \, j = \overline{1,l}\}$$

Зв'язки «елемент-посилання» визначаються наступним чином:

$$E_R = \{< r_{ij}, l_{ij} > | \, i = \overline{1,n}, \, j = \overline{1,l}, \exists \, k \in \{1,n\} \, l_{ij} = V_k\}$$

Зв'язки «елемент-дані» визначаються наступним чином:

$$E_D = \{< f_{ij}, v_{ij} > | \, i = \overline{1,n}, \, j = \overline{1,m}\}$$

Об'єктно-орієнтована база даних (OODB) може бути представлена розширеною реляційною моделлю, у якій носієм є об'єкти пари ідентифікатор - значення), множиною операцій – алгебра $\Re$, множиною відношень – відношення між об'єктами *R,* тобто

$$OODB =< \Theta, \Re, R >,$$

де

$$\Theta = \{< id, v >\},$$

$$v = \begin{bmatrix} id \in O \\ \{o_1,...,o_n\}, o_i \in V \\ [A_1 : o_1, A_2 : o_2,..., A_n : o_n] \end{bmatrix}$$

$$\Re = \langle \pi, \sigma, \cup, \cap, -, apply, apply\_at, close, tuple \rangle$$

Дані моделі *RDF* можуть бути представлені реляційною моделлю (A relational algebra for SPARQL), у якої носієм є кортежі RDF, множиною операцій – реляційна алгебра $\Re$, множиною відношень – відношення між кортежами (схема RDF, яка може використовуватися як OWL) *R*, тобто

$$RDF =< r, \Re, R >,$$

$$\Re = \langle \pi, \sigma, \triangleright\triangleleft, \cup, \cap, - \rangle.$$

На фізичному рівні RDF описується кортежами вигляду

$$RDF =< s, p, o >,$$

де s - суб'єкт,

p - предикат,

o - об'єкт

Файлове сховище FS може бути представлене у вигляді моделі даних XML з наступним тлумаченням:

V_{elemer}- вершини – елементи (папки та файли);

V_{valu}- вершини даних (атрибути файлів та папок, вміст файлів);

E_E - зв'язки «папка»- «папка» чи «папка-файл»;

E_A - зв'язки «елемент»-«атрибут» (атрибути файлів та папок);

E_R - зв'язки «елемент»- «посилання» (наприклад, символічні посилання);

E_D - зв'язки «файл»- «дані» (вміст файлу).

Структурований текст може бути представлений реляційною моделлю, у якої носієм є кортежі, складені з наборів значень атрибутів $SF = \{sf_i \mid i = \overline{1,n}\}$, множиною операцій – реляційна алгебра $\Re$, множина відношень є пустою.

Неструктурований текст та *мультимедійна інформація* може бути структурована неоднозначно, тому в загальному випадку їх неможливо представити математичною моделлю. Вони можуть бути використані в сховищі даних у вигляді файлів, а користувач може отримувати їх дані за допомогою програми-обробника.

2.4. Побудова моделей МБСД

2.4.1. Побудова концептуальної моделі МБСД

Концептуальна модель сховища задається як алгебрична система з декількома основами:

$$W = < \Xi, \Omega, \Re >, \tag{2.1}$$

де Ξ - множина основ, Ω - сигнатура, $\Re$ - відношення на основах.

Дана алгебрична система відрізняється від існуючих наявністю нових носіїв та операцій, що дозволяє побудувати сховище і виконувати запити до даних, представлених різними моделями.

Множина основ складається з усіх об'єктів, які потрібні для побудови сховища і виконання запитів :

$$\Xi = \{E, T, M, X, N\},$$

де E - сутності сховища; T - елементи реляційних БД; M - елементи багатовимірних БД; X - елементи БД XML; N - елементи БД NoSQL.

Сигнатура складається з усіх операцій, які потрібні для побудови сховища і виконання запитів :

$$\Omega = \{\Omega_{E1}, \Omega_{E2}, \Omega_{Md}, \Omega_{Xml}\},$$

де $\Omega_{E1} = \{\cup, \cap, /, \times, \pi, \sigma, \triangleright \triangleleft\}$ - базові операції (реляційної алгебри); $\Omega_{E2} = \{\mathrm{E}, \Phi, \mathrm{H}, \mathrm{Z}, \mathrm{T}, \mathrm{M}, \mathrm{X}, \mathrm{N}\}$ - операції перетворення носіїв; $\Omega_{Md} = \{зріз, обертання, згортка, деталізація\}$ - операції багатовимірної моделі; $\Omega_{Xml} = \{\chi, \delta, \Sigma\}$ - додаткові операції моделі даних XML.

Модель даних БД NoSQL не має ніяких додаткових операцій.

Множина відношень складається з усіх відношень, визначених на носіях сховища:

$$\Re = \Re_{EE} \cup \Re_{ET} \cup \Re_{EM} \cup \Re_{EX} \cup \Re_{EN} ,$$

де $\Re_{EE}$ - відношення між сутностями сховища; $\Re_{EM}$ - відношення між сутностями і об'єктами реляційних БД; $\Re_{EX}$ - відношення між сутностями і об'єктами БД XML; $\Re_{EN}$ - відношення між сутностями і об'єктами БД NoSQL.

Таким чином задана концептуальна модель відповідає наступним відношенням у метаописі:

- $\{< a, e >\}$ - перелік областей даних з вказанням сутностей, з яких вони складаються;
- $\{< e, b >\}$ - перелік сутностей з вказанням атрибутів, які визначені для цих сутностей;
- $\{< b_1, b_2 >\}$ - перелік зв'язків між атрибутами.

Опишемо базові операції (реляційної алгебри) $\Omega_{E1} = \{\cup, \cap, /, \times, \pi, \sigma, \triangleright \triangleleft\}$.

$\cup_T$ - ***оператор об'єднання*** – бінарний оператор над двома сутностями, у яких повинні бути одинакова кількість стовпчиків. Повертає сутність, яке містить рядки першої та другої сутностей;

$\cap_T$ - ***оператор перетину*** – бінарний оператор над двома сутностями, у яких повинні бути одинакова кількість стовпчиків. Повертає сутність, яке містить рядки, що належать одночасно першій та другій сутності;

$^-{}_T$ - ***оператор різниці*** – бінарний оператор над двома сутностями, у яких повинні бути одинакова кількість стовпчиків. Повертає сутність, яке містить рядки, що належать першій, але не другій сутності;

$\times_T$ - ***оператор декартового добутку*** – бінарний оператор над двома сутностями. Повертає сутність, стовпчиками якої є стовчики першої та другої сутностей, а рядками – множина всіх пар рядків першої та другої сутності;

π_T - ***оператор проекції*** - унарний оператор над сутністю з параметром – множиною стовпчиків. Повертає значення рядків сутності у заданих стовпчиках;

σ_T - ***оператор селекції*** - унарний оператор над сутністю з параметром – умовою вибору. Повертає значення частини рядків сутності, які відповідають вказаній умові;

$\triangleright\triangleleft_T$ -***оператор природного з'єднання*** (***злиття***) – бінарний оператор над двома сутностями з параметром - умовою відбору, визначеною на стовпчиках двох сутностей. Є еквівалентним виконанню операторів декартового добутку двох сутностей з наступною селекцією отриманої таблиця по вказаній умові відбору;

Опишемо операції перетворення носіїв $\Omega_{E2} = \{\mathrm{E}, \Phi, \mathrm{H}, \mathrm{Z}, \mathrm{T}, \mathrm{M}, \mathrm{X}, \mathrm{N}\}$.

E - ***оператор формування сутності*** – унарний оператор над об'єктом РБД, БВБД, БД XML, БД NoSQL. Повертає сутність сховища, що відповідає об'єкту;

Ф - ***оператор виділення структурованої частини*** – унарний оператор над сутністю, що повертає сутність, що є структурованою частиною даної сутності;

H - ***оператор виділення слабко-структурованої частини*** – унарний оператор над сутністю, що повертає сутність, що є слабко-структурованою частиною даної сутності;

Z - ***оператор з'єднання сутності -*** складений бінарна оператор над двома сутностями, що повертає сутність із впорядкуванням рядків за вказаним атрибутом. Вона використовується для отримання частково-структурованих даних сутності e, що зберігаються в сутностях e_1 і e_2 та має наступні властивості операції декартового добутку, а саме є дистрибутивною відносно операцій $\cup, \cap, /$ не є комутативною, асоціативною та ідемподентною.

T - ***оператор формування об'єкту РБД*** – унарний оператор над сутністю. Повертає об’єкт РБД, що відповідає сутності;

M - ***оператор формування об'єкту БВБД*** – унарний оператор над сутністю. Повертає об’єкт БВБД, що відповідає сутності;

X - ***оператор формування об'єкту БД XML*** – унарний оператор над сутністю. Повертає об’єкт БД XML, що відповідає сутності;

N - ***оператор формування об'єкту БД NoSQL*** – унарний оператор над сутністю. Повертає об’єкт БД NoSQL, що відповідає сутності;

Опишемо додаткові операції моделі даних XML $\Omega_{Xml} = \{\chi, \delta, \Sigma\}$.

δ_T - ***оператор видалення дублікатів*** – унарний оператор над сутністю. Повертає сутність, у якої будь-які два рядки не є однаковими по всіх значеннях;

Σ_T - ***оператор групування*** – унарний оператор над сутністю з параметрами: стовпчики, по яких відбувається групування c_g, стовчики, значення яких групуються c_{vg}, і функції f_g, які виконуються над цими стовпчиками. Повертає сутність, що складається з проекції c_g з видаленням дублікатів (нехай це буде *t’*), та значень виконання f_g над селекцією проекції c_{vg} по умові, що c_g має значення з *t’*;

Θ_T - ***оператор сортування*** – унарний оператор над таблицею з параметрами - стопчики та функція сортування. Повертає рядки сутності у порядку, визначеному функцією сортування.

Для розподілу даних у сховищі необхідно їх класифікувати за областями МБСД (множинами сутностей, відношення між якими утворюють зв’язний граф), а також за класами структурованості даних.

В концептуальній моделі сховища множина його сутностей розбивається на підмножини A_i, що називаються областями сховища, кожна сутність у одній області має відношення до іншої сутності в тій самій області і для кожної з яких можна побудувати зв’язний граф, причому дві різні області не перетинаються між собою:

$$E_{Cpt} = \bigcup_{i=1}^{n} A_i,$$

де $A_i = \{e_j \mid j = \overline{1,|A_i|}\}$, $\forall e_j \in A_i \; \exists e_k < e_j, e_k > \in E : E$, $\forall A_i, A_j \mid i \neq j \; : \; A_i \cap A_j = \varnothing$

Виділення таких областей дозволяє ефективно виконувати запити до декількох зв'язаних сутностей в силу спільного фізичного розташування. Розбиття робиться за допомогою операцією "побудова зв'язної множини" розширеної реляційної моделі, що дозволяє зменшити витрати часу на визначення областей сховища.

Сутності сховища класифікуються також за їх структурованістю, в результаті чого розрізняють структуровані $E_{стр}$, частково-структуровані $E_{чст}$, слабко-структуровані $E_{слс}$ та неструктуровані дані.

$$E_{Cpt} = E_{стр} \cup E_{чст} \cup E_{слс}$$

$$E_{чст} : \{e_{чст} \mid \Phi(e_{чст}) \neq e_{чст} \wedge \mathrm{H}(e_{чст}) \neq e_{чст}\}$$

$$E_{слс} = \{e_{слс} \mid \mathrm{H}(e_{слс}) = e_{слс}\}$$

Частково-структуровані дані розподіляються на структуровану $E_{стр1}$ і слабко-структуровану $E_{слс1}$ частини, в результаті чого сховище складається лише з структурованих і слабко-структурованих даних:

$$\forall e_{чст} : e_{стр1} = \Phi(e_{чст})$$

$$\forall e_{чст} : e_{слс1} = \mathrm{H}(e_{чст})$$

$$E_{стр} = \{e_{стр0}\} \cup \{e_{стр1}\}$$

$$E_{слс} = \{e_{слс0}\} \cup \{e_{слс1}\}$$

$$E_{Cpt'} = \{E_{стр0} \cup E_{чст} \cup E_{слс0}\}.$$

За визначеними сутностями з переліком атрибутів формується перелік областей сховища наступним чином.

1. Формування відношення між сутностями. Для цього отримуємо відношення зв'язку сутності, її атрибутів і атрибутів, пов'язаних до цих атрибутів:

$$\{<e,b>\} \mid\!><\!\mid \{<b_1,b_2>\} \rightarrow \{<e,b_1,b_2>\}$$

Далі з цього відношення отримуємо відношення зв'язку сутностей через зв'язок атрибутів:

$$\{<e,b_1,b_2>\}|><|\{<e,b>\}\rightarrow\{<e_1,b_1,b_2,e_2>\}$$

З цього відношення отримуємо відношення зв'язку сутностей, виключаючи зв'язок сутності на себе:

$$\{<e_1,b_1,b_2,e_2>\}\rightarrow\{<e_1,e_2>\}=EE$$
$$EE=EE/\{<e_i,e_i>\}$$

2. За допомогою функції побудови зв'язних множин $f(EE)$ визначаються області сховища. Це відбувається шляхом повторного застосування функції до всіх сутностей до тих пір, поки кількість атрибутів відношення буде зростати. Функція побудови зв'язних множин $f(EE)$ визначається наступним чином:

$$f(E'):$$
$$\begin{cases} E'_1=E' \\ E'_1 \underset{E'_1.b_{|B_{EE_1}|}=E'.b_1}{|\triangleright\triangleleft|} E'=\{<e_{b_1}..,e_{b_{|B_{EE_1}|}},e_{EE.b_2}>\}\rightarrow E'_1\text{'} \\ f(E')=E'_1/\sigma_{E'_1.b_{|B_{E'_1}-1|}\in\pi_{b_1}E'_1} \end{cases}$$

де E'_1 - відношення, у якому за допомогою операцій зовнішнього сполучення, множинної різниці, селекції та проекції формується переліки сутностей множин; $B_{E'_1}$ - перелік всіх атрибутів цього відношення, що розширюється після виконання процедури зовнішнього сполучення; $E'b_1$ - першний атрибут відношення E'_1; $E'_1.b_{|EE_1|-1}$ - передостанній атрибут відношення E'_1; $E'_1.b_{|EE_e|}$ останній атрибут відношення E'_1.

Отримані області ми послідовно нумеруємо, для того, щоб розрізнити їх між собою:

$$\{<e_1..e_n>\}\rightarrow\{<num,e_1..e_n>\}.$$

Після цього ми перетворюємо опис областей з формату кортежу в формат «один до багатьох», де кожній сутності виділено окремий кортеж:

$$\{<a=num_i,e_1..e_j..e_n>\}\rightarrow\{<a=num_i,e_j>|\ j=\overline{1,n}\}.$$

Після задання концептуальної моделі проектувальником сховища задаються **відношення між сутностями сховища і елементами даних джерел,**

які показують, які елементи даних в відповідних моделях джерел відповідають яким сутностям, і які зв'язки - яким відношенням.

Ці відношення схеми відображаються наступними реляційними відношеннями в метаописі сховища:

$\{< e, m^{e}_{Src}, e_{Src} >\}$ - відношення між сутностями та елементами даних джерел, де $e \in E$ - сутність, e_{Src} - елемент сховища в відповідній моделі m^{e}_{Src};

$\{< a^{e_i}_{j}, m_{Src}, a^{e_h}_{k} >\}$ - відношення між атрибутами сутностей і атрибутами відповідних моделей, де $a^{e_h}_{k}$ - «атрибут» в моделі m_{Src}, що відповідає атрибуту сутності $a^{e_i}_{j}$.

На основі заданих моделей джерел, концептуальної моделі сховища та відношення між ними будується логічна модель.

2.3.2. Побудова логічної моделі МБСД

Логічна модель МБСД пов'язує сутності з елементами даних і формулюється у вигляді алгебричної системи сховищ даних гібридного типу (2.1), де множина основ є наступною:

$$\Xi = \{E, T_{Ds}, M_{Ds}, X_{Ds}, N_{Ds}\},$$

де E - сутності сховища; T_{Ds} - елементи реляційних носіїв сховища; M_{Ds} - елементи багатовимірних носіїв сховища; X_{Ds} - елементи носіїв сховища XML; N_{Ds} - елементи носіїв сховища NoSQL.

З метою отримання цих елементів даних сховища визначаються множини моделей для представлення даних**,** при яких виконуються умови до інформаційної технології, з яких в свою чергу відбираються ті їх елементи, які дозволяють досягнути якісного значення критерію мінімальної сукупної вартості збереження та обробки даних.

Логічна модель відображає елементи баз даних, які відповідають сутностям з концептуальної моделі. Оскільки, МБСД не лише зберігають дані абстраговано, але й дані можуть бути представлені різними моделями, то логічна модель МБСД складається з взаємовідповідностей між сутностями,

атрибутами сутностей і відношеннями концептуальної моделі та елементами, атрибутами елементів та зв'язками даних з вказанням моделі, яка використовується.

Логічна модель задається наступними реляційними відношеннями у метаописі сховища:

- $\{< e, m, de >\}$ - відношення між сутностями та елементами даних сховища : $e \in E$ - сутність, e - елемент сховища в моделі $m \subset M^e$;

- $\{< a_j^{e_i}, m, b_k^{e_h} >\}$ - відношення між атрибутами сутностей і атрибутами відповідних моделей де $b_k^{e_h}$ -«атрибут» в моделі m, що відповідає атрибуту сутності $a_j^{e_i}$.

Множини моделей для представлення даних - $m_e^{доп} \in \{M_{Rel}, M_{NoSQL}, M_{MD}, M_{XML}\}$, де M_{Rel} - реляційна модель даних, M_{NoSQL} - модель даних NoSQL, M_{MD} - багатовимірна модель даних, M_{XML} - модель даних XML, визначаються наступним чином: для кожної сутності *e* визначаються допустимі моделі її зберігання, а також структурованість сутності *e*, тобто наявність однієї структури даних, декількох структур даних, або змішаної структури, при якій деяка частина атрибутів складає єдину структуру, а інша частина має декілька структур. Відповідно до структурованості даних моделі даних включаються у множину допустимих моделей, і для кожної сутності *e* та моделі з множини допустимих визначаються відповідні об'єкти баз даних:

- реляційної БД $\forall e_{стр} : t_{e_{стр}} = \mathrm{T}(e_{стр})$;
- багатовимірної БД $\forall a_i : m_{a_i} = \mathrm{M}(a_i)$;
- БД XML $\forall e_{слс1} : x_{e_{слс1}} = X(e_{слс1})$;
- БД NoSQL $\forall e_{слс0} : n_{e_{слс0}} = \mathrm{N}(e_{слс0})$.

Для кожної області $a_i \in A$ за допомогою оператора представлення багатовимірною моделлю визначається, чи може вона бути представлена багатовимірною моделлю, у разі позитивного висновку ця модель додається у множину допустимих $\{< e, m_e^{доп} >\} = \{< e, m_e^{доп} >\} \cup < e, m_{MD} >$.

Для знаходження моделей, при яких досягається якісне значення критерію мінімальної сукупної вартості збереження та обробки даних, потрібно отримати таку підмножину його кортежів, яка б задовільняла критерію вартості, а саме:

$$\begin{array}{l} < e, m, e_{Src} > \to c = c_1(e_{Src}) + c_2(q, e_{Src}) \mid q = gen(e_{Src}) \\ \{< e, m, de >\} \to \{< e, m_{Ds}, de_{Ds} > \mid \sum c \to \min\} \end{array},$$

де $c_1(e_{Ds})$ - функція вартості розміщення даних в елементі e_{Ds}; $c_2(q, e_{Ds})$ - функція вартості виконання запиту q на елементі даних e_{Ds}; $gen_m(e_{Ds})$ - функція генерування запиту до елементу даних e_{Ds}, заданого в моделі m.

Для знаходження моделей, при яких досягається якісне значення критерію мінімальної сукупної вартості збереження та обробки даних, у роботі використовується метод спрощення формул алгебричної системи РСД ГТ за рахунок визначення перспективних множин інваріантів з наступним деталізованим вибором кращого у цих множинах, який відрізняється від існуючих наявністю правил оцінювання інваріантів формули запиту на основі критерію вартості, що дозволяє отримати інваріант виразу запиту до сховища з мінімальною сукупною вартістю збереження та обробки даних.

Спрощення формул запитів відбувається на етапі вибору логічної моделі МБСД сховища даних шляхом вибору моделей для виконання запитів до областей даних сховища. При цьому інваріант цих формул з мінімальною сукупною вартістю збереження та обробки даних вибирається шляхом оцінки наявних інваріантів і визначення мінімального за попарним порівнянням оцінених інваріантів.

Для вибору такого інваріанту спрощення формул, яким можуть бути представлено запити $q \in Q$, використовується наступна оцінка:

$$O = \sum_{j=1}^{l} C_j \left(U_j^P(\mu) + U_j^Q(q, \mu) \right) \text{ при виконанні умови } \forall q : t(q, \mu) < f(s_q), \qquad (2.2)$$

де: $\mu = < \mu_{Rel}, \mu_{Md}, \mu_{XML}, \mu_{NoSQL} >$ – бінарні ознаки вибору моделей для розміщення даної сутності або області (1 – дана модель використовується для даної сутності або області, 0 – не використовується), причому кожна комбінація значень яких

дає набір моделей для представлення даних і відповідний множинний інваріант виконання запитів до цих даних; C_j – питома вартість використання *j*-го ресурсу (грн/мс); $U_j^P(\mu)$ – кількість використаного *j*-го ресурсу для розміщення даних відповідно до вибраних моделей; $U_j^Q(q,\mu)$ – кількість використаного *j*-го ресурсу для виконання запитів відповідно до вибраних моделей.

Для оцінки функцій $U_j^P(\mu)$ та $U_j^Q(q,\mu)$ потрібно попередньо задати у відповідних носіях елементи даних, що відповідають цим сутностям:

$$< e, m > \rightarrow < e, m, de > | \; de = conv\,(e, m)\,,$$

де $conv\,(e, m)$ - функція представлення сутності e моделлю m наступним чином:

– якщо використовується модель $m = m_{Src} \in M_e$, якою ця сутність представлена в джерелі даних, то відповідні елементи, атрибути, і зв'язки даних беруться з джерела: $e_{DW_j} \rightarrow de_{DS_i} = de_{Src_i}$, $a_k^{e_{DW_i}} \rightarrow da_k^{DS_i} = da_k^{Src_i}$, $r_{DW_j} \rightarrow dl_{DS_i} = dl_{Src_i}$

– для моделей $m \in M_e, m \neq m_{Src}^e$, відмінних від моделі, яка використовується для представлення цієї сутності в джерелі даних, відбувається процедура «перетворення даних» для представлення даних цими моделями за допомогою відповідних операторів алгебричної системи

Дані оператори детально описані в розділі 3.

Після визначення елементів логічної моделі області розгортаються на одному вузлі. Однак для того, щоб визначити сховище як розподілену систему зберігання даних, потрібно задати його фізичну модель, що описано нижче.

2.3.3. Побудова фізичної моделі МБСД

Фізична модель задає змінні параметри сховища, які впливають на його характеристики – вартість, швидкодію і доступність, однак при їх зміні не змінюється логічна модель. Такими параметрами є розподіл даних по вузлах сховища, кожен з яких містить набір носіїв для підтримки всіх моделей даних, та маршрут реплікації, що відображає, у якому порядку реплікуються вузли сховища.

Фізична модель сховища задається наступними відношеннями у метаописі сховища:

$\{< a, n >\}$ – перелік всіх вузлів *n* сховища із вказанням областей даних *a*, що зберігаються і обробляються в цих вузлах;

$\{< a, n_1, n_2 >\}$ – перелік областей даних джерела *a* з вказанням шляхів реплікації даних між вузлами n_1, n_2.

Для визначення кортежів цих відношень перед побудовою фізичної моделі МБСД адміністратором спочатку задаються початкові відношення, а саме:

- $\{< n, a >\}_{пот}$ - перелік всіх вузлів сховища із вказанням потенційних областей даних, що можуть зберігатися і обробляються в даних вузлах;
- $\{< n, m >\}_{пот}$ - перелік всіх вузлів сховища із вказанням потенційних моделей даних, що можуть використовуватися для представлення в даних вузлах.

Після формування відношення відповідності вузлів і областей даних з потрібно отримати таку підмножину його кортежів, яка б задовільняла критерію вартості, а саме:

$$< n, a > \rightarrow c = c_1(a, n) + c_2(Q, n) \mid Q = gen(a)$$
$$\{< n, a >\}_{пот} \rightarrow \{< n, a > \mid \sum_a c \rightarrow \min\}$$

де $c_1(a, n)$ - функція вартості розміщення даних області *a* в носіях даних вузла *n*; $c_2(q, n)$ - функція вартості виконання набору запитів *Q* в носіях даних вузла *n*; $gen(a)$ - функція генерування набору запитів *Q* до областей даних *a*.

Для визначення конфігурації розташування даних, яка б виконувала дані вимоги, розв'язується задача оптимізації сукупної вартості збереження та обробки даних, що формулюється наступним чином:

$$\sum_{i=1}^{k}\sum_{j=1}^{l} C_j^i \cdot \left(U_{ij}^D(\delta) + U_{ij}^Q(\delta, Q)\right) \rightarrow \min \qquad (2.3)$$

за таких умов:

$$\forall q \in Q : t_q \le f(s_q), \qquad (2.4)$$

$$\forall j \in \overline{1,l} : \sum_{i=1}^{k} \delta_j^i \ge \nu \qquad (2.5)$$

де $\delta = [\delta_j^i]$ – конфігурація розгортання; δ_j^i – розміщення області даних *j* у вузлі *i*, що визначає, у яких вузлах будуть збережені дані та виконуватися запити до певних областей ($\delta_j^i = 1$-область *j* розміщується у вузлі , $\delta_j^i = 0$ - не розміщується); C_j^i – вартість використання *j*-го ресурсу у *i*-му вузлі; $U_{ij}^Q(\delta, Q_\partial)$ і $U_{ij}^D(\delta)$ кількість використаного *j*-го ресурсу у *i*-му вузлі для виконання набору запитів і розміщення даних (залежить від розташованих в даному вузлі областей і набору запитів); t_q – час виконання запиту *q*; $f(s_q)$ – обмежувальна функція, визначена на обсязі даних запиту s_q; v – задана нижня межа кількості реплікованих копіїдоступності областей сховища; Q – множина запитів, для якої здійснюється розміщення;

З метою перевірки виконання вищезазначених умов, генеруються тестові запити Q_m кількістю $n_q = \max(n_{зад}, n_A)$, де $n_{зад}$-задане число запитів; n_A-число областей.

Для проведення обчислень у роботі прийнято наступне:

– $U_{ij}^Q(\delta, Q_m)$ визначається як кількість МБ розміщуваних даних,

– C_j^i - 1 грн за МБ;

– $U_{ij}^D(\delta)$ визначається як сумарна кількість часу у мс на виконання набору запитів Q;

– C_j^i - 1 грн за мс.

Ця задача розв'язується для набору запитів Q, який формується наступним чином: якщо на момент фізичного проектування

- не було виконано жодного запиту, то генеруються тестові запити Q_m кількістю $n_q = \max(n_{зад}, n_A)$, де: $n_{зад}$-задане число запитів, n_A-число областей. Запити генеруються за процедурою «генерування запитів» (п.3.5.4.) довільним чином з таким розрахунком, щоб до кожної області повинен бути хоча б один запит, тобто послідовно по всіх областях сховища;

- була зібрана статистика $Stat = \{< Q, t >\}$ (Q - визначення запиту, t - час виконання запиту), то з статистики вибираються всі запити, які запізнилися, але не більше $n_{зад}$. Якщо таких запитів, які запізнилися, менше $n_{зад}$, то вони доповнюються випадково вибраними запитами з статистики.

Задача оптимізації може бути декомпозована у випадку, якщо на деяких вузлах можуть розміщуватися лише деякі області, а інші області не розміщуються на цих вузлах:

$$\exists\{n\} \in N_{a_i} \mid \{n\} \notin N_{a_j}, j = \overline{1,n}, j \neq i$$

Для всіх таких множин вузлів окремо розв'язується задача оптимізації, яка відрізняється тим набором запитів, які беруться лише з цих областей, і в змінні входять лише ознаки розміщення для тих вузлів, на яких розміщуються ці області.

Такі часткові задачі розв'язуються послідовно, в порядку зростання кількості вузлів та кількості запитів, що запізнилися.

Також задача може бути декомпозована у випадку, якщо використання всіх активних вузлів близьке до 100%, і є запити, що запізнюються. Тоді розв'язується підзадача, вузлами якої є додаткові вузли, а набором тестових запитів – запити, що запізнюються. Окремо розв'язується задача оптимізації для решти вузлів і запитів, які не запізнюються, після чого отримані результати з'єднуються і це є розв'язком задачі.

Для розміщених областей у вузлах сховища потрібно вибрати шляхи реплікації областей по вузлах сховища, на яких вони розміщені:

$$\begin{array}{l} \{< a, n >\} \mid\!><\!\mid \{< n >\} \to \{< a, n, n_2 >\} \\ \{< a, n, n_2 >\} \to c = c_r(a, n, n_2) \\ \{< a, n, n_2 >\} \to \{< a, n_1, n_2 > \mid \sum_a c \to \min\} \end{array},$$

де $c_r(a, n, n_2)$ - функція вартості реплікації області a з вузла n в вузол n_2 .

Для цього розв’язується задача оптимізації за критерієм мінімальної вартості реплікації області сховища:

$$\sum_{i=1}^{k} \sum_{j=1}^{l} C_j^i \cdot U_{ij}^R(\rho^a) \to \min, \qquad (2.6)$$

за умови

$$\sum_{a \in A_i} u_{ij}^{R}(\rho^a) \le R_{\lim} \qquad (2.7)$$

де $\rho^a = \left[\rho_{i,j}^a\right]$ - ознаки реплікації вузлів для області a ($\rho_{i,j}^a$=1 область a реплікується з вузла i в вузол j, $\rho_{i,j}^a$=0 – не реплікується); $U_{ij}^{R}(\rho^a)$ - споживання ресурсу i вузла j (у % від максимальної потужності); $u_{ij}^{R}(\rho^a) = \max_t U_{ij}^{R}(\rho^a)$ - максимальне споживання ресурсу і вузла j (у % від максимальної потужності); $R_{\lim}$ - ліміт споживання ресурсів для реплікації (у % від максимальної потужності).

Для визначення компонентів фізичної моделі ставиться задача мінімізація вартості при розподіленні навантаження між декількома вузлами для забезпечення виконання до $n_{зад}$ запитів при граничному часі виконання $f(s_q)$ та забезпеченою доступністю областей *av*.

Це здійснюється за допомогою розміщення областей сховища по вузлах та визначення маршруту реплікації даних між вузлами, які складають фізичну модель сховища.

Перепроектування сховища відбувається у випадку перевищення допустимого часу виконання запитів $f(s_q)$ або у випадку зниження доступності сховища нижче заданого допустимого значення *av* .

Висновки до розділу 2

У даному розділі вводиться формальне визначення МБСД як розподіленого сховища даних гібридного типу, побудованого за трьома рівнями абстракції даних – рівнем даних, рівнем вузлів сховища та рівнем носіїв сховища. Вузли МБСД складаються з реляційних баз даних, багатовимірних баз даних, баз даних XML, файлів файлової системи, що дозволяє використати переваги різних носіїв даних для підвищення ефективності виконання запитів. Розроблено математичні моделі концептуального, логічного та фізичного рівня опису МБСД з врахуванням наявності різних моделей даних (реляційної, багатовимірної, XML, NoSQL) для представлення ними даних та різних

фізичних вузлів для збереження та обробки даних у них, що дозволяє автоматизувати процес формування метаопису розподіленого сховища гібридного типу із здійсненням розподілу даних між різними моделями та вузлами даних.

Концептуальна модель МБСД представлена у вигляді формальної алгебричної системи сховищ даних гібридного типу, яка відрізняється від існуючих набором основ (сутності, елементи даних РБД, БВБД, БД XML і NoSQL) та операцій (новими є операції перетворення основ між собою, виділення структурованої і слабко-структурованої частини відношення, з'єднання сутностей, побудови зв'язної множини), дослідження властивостей яких дозволило оперувати даними, представленими різними моделями, та будувати для них РСД ГТ.

Логічна модель МБСД відображає відповідність сутностей концептуальної моделі і елементів даних моделей носіїв, у яких власне зберігаються дані. Вибір моделей для збереження даних впливає на вартість розміщення даних та виконання запитів до них.

Фізична модель МБСД відображає розміщення даних між вузлами сховища та порядок реплікації між ними, що впливають на вартість функціонування сховища, швидкодію виконання запитів та доступність даних.

Для всіх етапів побудови МБСД описані перетворення відношень у метаописі сховища, що задані розширеною реляційною моделлю, з використанням стандартних реляційних операторів, а також визначеної функції «побудови зв'язних множин».

Для побудови логічної та фізичної моделей МБСД запропоновано моделі вартості, які залежать від обраних моделей, конфігурацій розміщення та реплікації даних у сховищі. Описані моделі даних використовують як для представлення даних джерел, так і носіїв даних сховища.

Описані вище моделі дозволяють формалізувати процес інтеграції даних у сховище мінімальної вартості з дотриманням вимог необхідної швидкодії та доступності сховища.

Результати розділу 2 опубліковані в роботах [49..50,97..102].

РОЗДІЛ 3. МЕТОД ПОБУДОВИ РСД ГТ

У даному розділі описано метод побудови РСД ГТ з мінімальною сукупною вартістю збереження та обробки даних, зокрема побудова його концептуальної, логічної та фізичної моделей, методи та алгоритми розв’язання задач визначення моделей, розміщення даних, та вибору маршруту реплікації. Наводяться результати експериментів, пов'язаних з особливостями застосування евристичних алгоритмів для поставлених задач.

3.1. Загальна схема методу побудови РСД ГТ

Процес побудови РСД ГТ полягає в побудові концептуальної, логічної та фізичної моделей РСД ГТ за критерієм сукупної вартості збереження та обробки даних на основі розглянутих у розділі 2 міжрівневих переходів. Загальна схема побудови сховища задається наступним алгоритмом:

1. Побудова концептуальної моделі РСД ГТ:
   1.1. Задання джерел даних;
   1.2. Виділення сутностей і зв'язків сутностей;
   1.3. Виділення областей сховища даних;
   1.4. Класифікація даних за структурованістю даних;
2. Побудова логічної моделі РСД ГТ:

2.1. Вибір комбінації моделей для представлення даних, при якій виконуються умови до інформаційної технології (п. 3.2.1.). На цьому етапі здійснюється вибір перспективних множин інваріантів на основі аналізу структурованості даних;

2.2. Вибір комбінації моделей для представлення даних, при яких досягається якісне значення критерію вартості (п. 3.2.2.);

   2.3. Представлення даних у вибраних моделях (п. 3.2.3.);
3. Побудова фізичної моделі РСД ГТ:
   3.1. Вибір розташування областей по вузлах (п. 3.3.1.);
   3.2. Вибір маршруту реплікації областей по вузлах (п. 3.3.2.)

Побудова концептуальної моделі РСД ГТ. Адміністратор сховища даних вибирає множину дозволених для використання в сховищі моделей із множини

моделей, що містять реляційну, багатовимірну, XML та NoSQL моделі та задає джерела та носії даних.

На основі результатів роботи адміністратора проектувальник сховища даних визначає концептуальну модель, а саме сутності, атрибути, їх зв'язки та встановлює відповідність між концептуальною моделлю та відповідними елементами джерел.

У відповідністю з заданою концептуальною моделлю, процедурою, описаною в п. 3.2.1, визначаються моделі, які є допустимими для розміщення областей і сутностей шляхом аналізу структур даних. Серед цих допустимих моделей шляхом локального пошуку за критерієм вартості визначаються моделі, які є оптимальними для розміщення областей і сутностей.

На основі концептуальної і логічної моделей формується фізична модель даних. Розміщення даних областей по вузлах сховища визначається за критерієм мінімальної вартості та модифікованого генетичного алгоритму (п.3.3.1). Для цього розміщення вибирається маршрут реплікації даних сховища (п.3.3.2).

Розглянемо більш детальніше процес формування логічної та фізичної моделей сховища для сутностей, заданих концептуальною моделлю.

3.2. Опис операторів алгебричної системи РСД ГТ

3.2.1. Оператор формування сутності

Для реляційних баз даних:

1. Відношення РБД, що додаються до сховища, r стають носієм структурованої частини T

$$\begin{aligned} r' &= \{< a_1, a_2, ..., a_n >\} \\ h' &= \{a_1, a_2, ..., a_n\} \\ t' &= \{< r_j, h_j >\} \end{aligned} \tag{3.1}$$

2. Обмеження цілісності, взяті з схеми РБД, стають відношеннями структурованої частини.

Для первинних ключів K, що складаються з атрибутів a_i, додаємо обмеження цілісності на відповідних стовцях $(c_i.c_i)$

$$a_i \in K \rightarrow (c_i.c_i) = r_T^{'}.$$

Для зовнішніх ключів F, що пов'язують атрибути первинного ключа a_i з атрибутами a_j a_i, додаємо обмеження цілісності на відповідних стовпцях $(c_i.c_j)$

$$F - \text{зовнішній ключ}: \forall a_i \in F \; \forall t(a_i) \in u(a_j) \rightarrow (c_i, c_j) \Rightarrow r_T^{'}.$$

3. Перевіряється, чи не відповідає таблиця t', а саме схема h' і відношення цілісності h' існуючим. Для цього:

3.1.Для кожної таблиці $t' \in T$ перевіряється $h' \subset h$.

3.2. Якщо $h' \subset h$, то в залежності від рішення проектувальника, яке надається через інтерфейс, можливі наступні варіанти:

– створюється нова таблиця $t' \Rightarrow T$,

– дані додаються до існуючої таблиці $t' \Rightarrow t \in T$.

Для багатовимірних баз даних:

1. Визначаються відношення метаданих та фактів, зокрема виділяються віддношення вимірів $RD = \{< d_i, a_{i1}, a_{i2}, ..., a_{in} >\}$ та відношення мір $RM = \{< d_1, d_2, ..., d_n, m_1, ..., m_n >\}$. Ці відношення стають носієм структурованої частини $RD \cup RM \Rightarrow r'$.

2. Обмеження цілісності, побудовані з отриманих відношень (первинні ключі у відношеннях метаданих c_i, унікальні ключі у відношеннях фактів c_i та зовнішні ключі між відношеннями фактів та метаданих по вимірах c_j), стають відношеннями структурованої частини

$$\begin{aligned} &d_i \Rightarrow c_i, < c_i, c_i > \Rightarrow r_T^{'}, \\ &d_i \Rightarrow c_j, < c_i, c_j > \Rightarrow r_T^{'}. \end{aligned} \qquad (3.2)$$

3. Перевіряється, чи не відповідає таблиця t', а саме відношення r' і відношення цілісності $r_T^{'}$ існуючим. Для цього:Для кожної таблиці $t' \in T$ перевіряється $r' \subset r$.

3.2. Якщо $r' \subset r$, то в залежності від рішення проектувальника, яке надається через інтерфейс, можливі наступні варіанти:

– створюється нова таблиця $t' \Rightarrow T$;

– дані додаються до існуючої таблиці $t' \Rightarrow t \in T$.

Для файлів XML та баз даних XML:

1. Визначаються відношення на атрибутах A тегів одної назви $name$, вкладені теги $b_1 . b_n$ заміняються ідентифікатором відповідного відношення

$$\begin{array}{l} \forall n = \delta(\pi(A, name)): \\ n(b_1, b_2 .., b_n) = \{id\} \cup \pi[E, n] \Rightarrow r' \end{array} \quad (3.3)$$

2. Будуються обмеження цілісності між відношеннями (первинні ключі у ідентифікаторах атрибутів id, зовнішні ключі між доданими атрибутами вкладених тегів b_e і відповідними ідентифікаторами id)

$$\begin{array}{l} < id, id > \Rightarrow r_T^{'} \\ < id, b_e > \Rightarrow r_T^{'} \end{array} \quad (3.4)$$

3. Перевіряється, чи не відповідає таблиця t', а саме відношення r' і відношення цілісності h' існуючим. Для цього:

3.1. Для кожної таблиці $t' \in T$ перевіряється $r' \subset r$.

3.2. Якщо $r' \subset r$, то в залежності від рішення проектувальника, яке надається через інтерфейс, можливі наступні варіанти:

– створюється нова таблиця $t' \Rightarrow T$;

– дані додаються до існуючої таблиці $t' \Rightarrow t \in T$.

Для БД NoSQL використовується пряме перетворення або таке через представлення моделлю XML:

1. Якщо всі значення є атомарними і для кожного атрибуту даних значення в різних записах мають один тип, то використовується пряма інтеграція даних, а саме з цих джерел отримуємо наступні відношення у схемі сховища :

1.1. Для БД NoSQL типу «ключ-значення : $< k, v > \Rightarrow r'$, де k – атрибут «ключ», v – атрибут «значення»;

1.2. Для БД NoSQL типу «ключ-документ» отримуємо наступні відношення у схемі сховища:

$$< id , v_1 , v_2 ..., v_n > \Rightarrow r' ,$$

де *id* – атрибут «ідентифікатор», v_i-атрибути значень, що задані хоча б у одному з документів колекції.

1.3. Для БД NoSQL типу «BigTable» отримуємо наступні відношення у схемі сховища:

$$< r, t, v_1, v_2, ..., v_n > \Rightarrow r',$$

де r – атрибут «ідентифікатор рядка», t – атрибут «мітка часу» , v_i-атрибути значень атрибути значень, що задані хоча б у одному з документів колекції

1.4. Дані БД NoSQL на базі графів можна розмістити в сховищі за допомогою відношення «вершина - дані», що відображає дані d , задані для вершини v:

$$< v, d > \Rightarrow r',$$

Крім того, застосовується відношення «вершина-вершина», що відображає існування ребер між вершинами графу з ідентифікаторами id_v :

$$< id_v, id_v > \Rightarrow r'.$$

2. Якщо хоча б одне значення не є атомарним або в межах одного атрибута є значення з різними типами даних, то спочатку потрібно представити дані моделлю XML (як вказано в п. 2.3), а потім розмістити дані за допомогою співвідношень (3.13) та (3.14).

3. Перевіряється, чи не відповідає таблиця t', а саме відношення r' існуючим. Для цього:

3.3. Для кожної таблиці $t' \in T$ перевіряється $r' \subset r$.

3.4. Якщо $r' \subset r$, то в залежності від рішення проектувальника, яке надається через інтерфейс, можливі наступні варіанти:

- створюється нова таблиця $t' \Rightarrow T$;
- дані додаються до існуючої таблиці $t' \Rightarrow t \in T$.

Дані об'єктно-орієнтованих баз даних можна розмістити у сховищі наступним чином.

1. Визначаються відношення на членах класу однієї назви, значення об’єктів інших класів заміняються ідентифікаторами відношень тих класів

$$class \quad (id \ , a_1 , a_2 .., \ a_n) \Rightarrow \ r' , \tag{3. 5}$$

де *id* – атрибут «ідентифікатор», a_i - атрибути класу

2. Будуються обмеження цілісності між відношеннями (первинні ключі у ідентифікаторах атрибутів *id*, зовнішні ключі між доданими атрибутами вкладених тегів a_{id}ч і відповідними ідентифікаторами a_{id})

$$\begin{aligned} &< id, id > \Rightarrow r_T^{'} \\ &< id, a_{id} > \Rightarrow r_T^{'} \end{aligned} \tag{3. 6}$$

3. Перевіряється, чи не відповідає таблиця *t*', а саме схема існуючим. Для цього:

3.1. Для кожної таблиці $t' \in T$ перевіряється $r' \subset r$.

3.2. Якщо $r' \subset r$, то в залежності від рішення проектувальника, яке надається через інтерфейс, можливі наступні варіанти:

- створюється нова таблиця $t' \Rightarrow T$;
- дані додаються до існуючої таблиці $t' \Rightarrow t \in T$.

Дані RDF можна розмістити у сховищі наступним чином.

1. Визначаються відношення RDF як набори кортежів з спільним суб’єктом S, їх атрибутами є предикати P_i, значеннями – об’єкти

$$S \ (S_0 , P_1 , P_2 .., \ P_n) \Rightarrow \ r' . \tag{3. 7}$$

2. Встановлюються обмеження цілісності між суб'єктами S_0 і об’єктами в інших суб’єктах за предикатом P_S:

$$\begin{aligned} &< S_0, S_0 > \Rightarrow r_T^{'} \\ &< S_0, P_S > \Rightarrow r_T^{'} \end{aligned} \tag{3. 8}$$

3. Перевіряється, чи не відповідає таблиця *t*', а саме схема існуючим. Для цього:

3.1. Для кожної таблиці $t' \in T$ перевіряється $r' \subset r$.

3.2. Якщо $r' \subset r$, то в залежності від рішення проектувальника, яке надається через інтерфейс, можливі наступні варіанти:

- створюється нова таблиця $t' \Rightarrow T$;
- дані додаються до існуючої таблиці $t' \Rightarrow t \in T$.

Дані структурованих файлів можна розмістити у сховищі наступним чином.

1. Визначаються відношення як набори кортежів, сформованих з рядків структурованого тексту (розбивка на атрибути sf_i визначається роздільником або задані попередньо)

$$(sf_1, sf_2 .., sf_n) \Rightarrow r'. \quad (3.9)$$

2. Перевіряється, чи відповідають ці існуючим. Для цього:

2.1. Для кожної таблиці $t' \in T$ перевіряється $r' \subset r$.

2.2. Якщо $r' \subset r$, то в залежності від рішення проектувальника, яке надається через інтерфейс, можливі наступні варіанти:

– створюється нова таблиця $t' \Rightarrow T$;

– дані додаються до існуючої таблиці $t' \Rightarrow t \in T$.

3.2.2. Оператори формування елементів носіїв з сутностей сховища

Оператор формування елементів даних реляційних БД.

1. Відношення сховища T (носій структурованої частини) стають відношеннями РБД r

$$T \Rightarrow r. \quad (3.10)$$

2. У схему РБД Z включаються первинні ключі та зовнішні ключі, визначені у схемі структурованої частини R_T (на тих відношеннях, які розміщуються в реляційну БД).

$$R_T \Rightarrow Z \quad (3.11)$$

Оператор формування елементів даних багатовимірних БД.

1. Якщо у таблиці відсутні як первинні ключі, так і зовнішні ключі на інші таблиці, то розміщення таблиці у БВБД неможливе, бо неможливо представити атрибути цієї таблиці як виміри або міри.

2. Первинні ключі, що не співпадають з зовнішніми, стають ідентифікаторами елементів вимірів a_d, відповідні відношення – відношеннями метаданих R_d. Неключові атрибути відношень метаданих $a_1, a_2 ., a_n$ стають атрибутами вимірів.

$$r(a_d.a_1, a_2.., a_n) \text{-> } R_d;$$
$$d :< a_d, a_d > \in R_T \Rightarrow D. \quad (3.\ 12)$$

3. Відношення, що містять зовнішні ключі a_d на відношення метаданих, стають відношеннями фактів R_T. Неключові атрибути відношень метаданих $m_1,..,m_n$ стають мірами.

$$< a_d, a_{d_i} > \in R_T;$$
$$r(a_{d_1},.., a_{d_n}, m_1,.., m_n) \Rightarrow R_f. \quad (3.\ 13)$$

Оператор формування елементів даних БД XML.

Кожен кортеж відношення $c_1..c_n$ стає тегом XML, причому ключові атрибути відношення (c_i) записуються як атрибути тега, інші (c_n) – як атрибути в тілі тега, отримаємо наступне представлення:

$$T(c_0, c_1..c_n);$$
$$c_1..c_n \Rightarrow V;$$
$$< c_i, c_j > \in R_T \Rightarrow E_E;$$
$$< c_0, c_i > \Rightarrow E_A;$$
$$< c_0, c_n > \Rightarrow E_D. \quad (3.\ 14)$$

Оператор формування елементів даних БД NoSQL.

Для розміщення даних у сховищі використовується БД типу «ідентифікатор -документ» кожен рядок таблиці представляється як документ колекції, що відповідає цій таблиці:

$$T(c_0, c_1..c_n) \Rightarrow \{id, a_1..a_n\}. \quad (3.\ 15)$$

За допомого представлення даних моделями носіїв сховища можна розмістити дані в носіях і мати змогу виконувати запити відповідно до операцій, що підтримують ці носії.

3.3. Метод спрощення формул алгебричної системи РСД ГТ

Для того, щоб визначити моделі, якими будуть представлені дані джерел у сховищі, в роботі застосовується метод спрощення формул алгебричної системи РСД ГТ. Цей метод в роботі удосконалено за рахунок визначення перспективних множин інваріантів з наступним деталізованим вибором кращого у цих множинах, який відрізняється від існуючих наявністю правил

оцінювання інваріантів формули запиту на основі критерію вартості, що дозволяє отримати інваріант виразу запиту до сховища з мінімальною сукупною вартістю збереження та обробки даних.

Введемо потрібні для подальшого опису поняття.

Визначення 3.1. *Формулою запиту* називається вираз у алгебричній системі (2.1), визначений на основах Ξ і сигнатурі Ω.

Визначення 3.2. *Множинним інваріантом запитів* $I(\mu)$ називається варіант запису двох чи більше формул запиту, що може бути отриманий шляхом одинакових тотожніх перетворень, де μ – комбінація моделей дл япредставлення даних, що породжує інваріант.

Визначення 3.3. *Оцінкою множинного інваріанту запитів* $I(\mu)$ є функція, визначена на множині інваріантів $\mathrm{I} = \{I(\mu)\}$ та множині запитів $Q = \{q\}$, що приймає лише додатні значення:

$$O_Q^{I(\mu)} = f\left(I(\mu), Q\right) > 0.$$

Визначення 3.4. Оцінкою множинного інваріанту запитів $I(\mu)$ є:

$$O = \sum_{j=1}^{l} C_j \left(U_j^P(\mu) + U_j^Q(q,\mu)\right). \tag{3.16}$$

Визначення 3.5. . *Інваріант запиту* $I(\mu_1)$ називається кращим за інваріант запиту $I(\mu_2)$ за оцінкою $O_Q^{I(\mu)}$, якщо $O_Q^{I(\mu_1)} < O_Q^{I(\mu_2)}$.

Визначення 3.6. Перпективною множиною інваріантів I_n називається множина інваріантів, кожен з яких є кращим за будь-який, що не належить даній множині:

$$\mathrm{I}_n = \left\{ I(\mu_n) \,|\, O_Q^{I(\mu_n)} > O_Q^{I(\mu')} \wedge I(\mu') \notin \mathrm{I}_n \right\}$$

Визначення 3.7. Допустимим інваріантом запитів називається інваріант, який відповідає умові допустимості:

$$\forall q \in Q : t(q, \mathrm{I}(\mu)) < f(s_q) \tag{3.17}$$

Визначення 3.8. Найкращим інваріантом запиту за оцінкою $O(I(\mu), Q)$ називається інваріант, який має мінімальну оцінку $\min\limits_{I(\mu)} O(I(\mu), Q)$ серед всіх можливих інваріантів.

У даних поняттях метод відображається наступною схемою (рис.3.1). Метод спрощення починається з множини всіх можливих інваріантів запитів, з якої відбираються перспективні підмножини на основі аналізу структурованості, у яких визначаємо допустимі інваріанти і посеред останніх вибираємо найкращий.

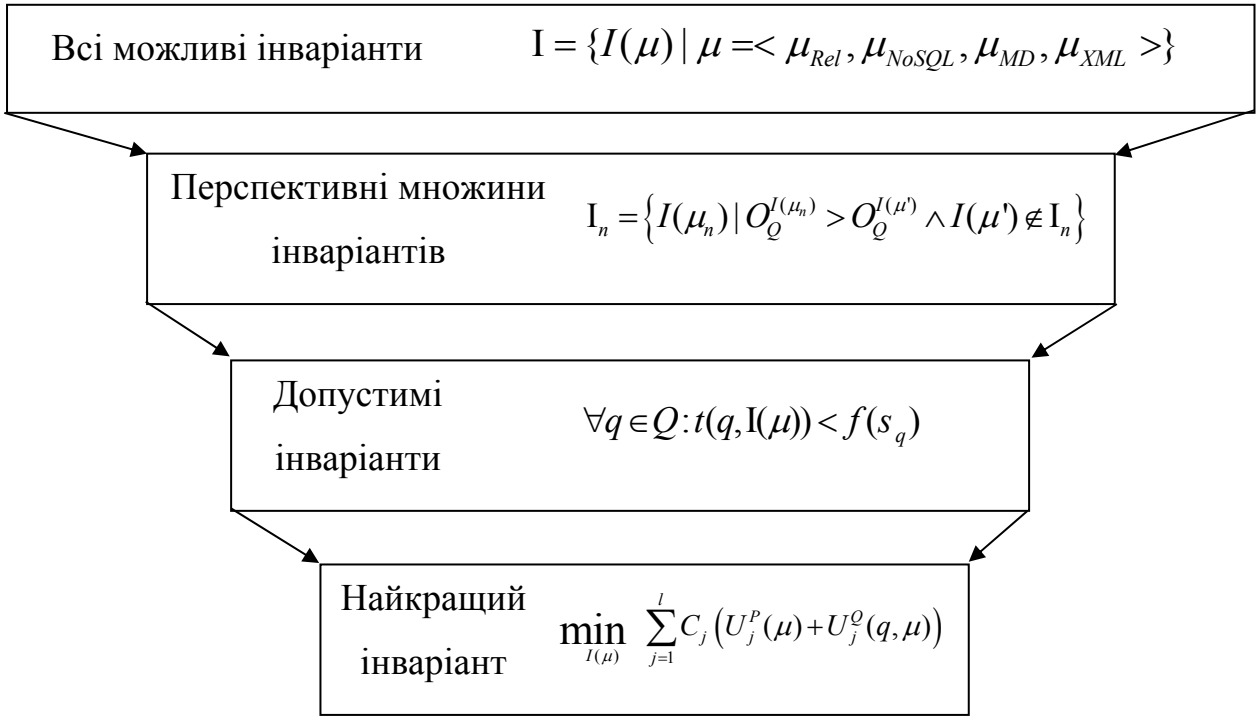


Рис. 3.1. Схема методу спрощення формул для АС РСД ГТ [Розроблено автором].

Розглянемо детальніше вищезазначені фази методу.

3.3.1. Визначення перспективних множин інваріантів

Для того, щоб вибрати перспективні множини інваріантів, що відповідають моделі даних, у які можуть бути представлені ці дані без падіння швидкодії виконання запитів до них, проводиться аналіз структур даних джерел.

Однією з істотних характеристик даних, яка суттєво впливає на дисковий простір, є їх структурованість. Відомо, що зберігання неструктурованих і слабко-структурованих даних здійснюється «розріджено», що в загальному випадку призводить до падіння швидкодії роботи зі сховищем даних, зокрема при реалізації операцій «введення – виведення».

Структурованість даних традиційно є одним з ключових чинників при ухваленні рішення про відповідний формат представлення даних (наприклад, реляційна бази даних для структурованих даних, форматі XML для частково структурованих даних та ін.). Структурованість також суттєво впливає на доступ до даних (наприклад, за допомогою SQL запитів для реляційних баз даних і XPath / Xquery для XML).

Дані у джерелах можна класифікувати за рівнем структурованості наборів даних наступним чином.

Структуровані дані мають чітко задану структуру й тому не виникає питань щодо їх розміщення у реляційних базах даних.

Структуровані дані можна поділити на такі два підкласи:

- ідеально-структуровані дані, для яких значення ST = 1 (клас джерела SCs.= 0). Оскільки у них немає невизначених значень, то множина атрибутів кожного елемента набору даних співпадає з множиною всіх атрибутів набору, тобто має місце

$$\forall\, r_l^i \neq \eta \ , \tag{3.18}$$

де тут і надалі η - невизначене або відсутнє значення;

- сильно-структуровані дані, для яких лише один стовпець має не визначені значення

$$\exists\, !\, i \ \ \exists \ \ r_l^{\,i} \ = \ \eta \ . \tag{3.19}$$

Для джерел з сильно-структурованими даними значення SC_s.= 1.

Слабко-структуровані дані. Для таких даних відсутня фіксована, жорстка схема, якій повинні відповідати дані, оскільки, структури цих даних є нерегулярними, неявними та носять частковий характер. В порівнянні з структурованими даними слабко-структуровані дані є більш гнучкими, оскільки можуть бути представлені різними схемами.

Основними характеристиками слабко-структурованих даних з точки зору схеми даних є те, що дані або можуть не відповідати заданій схемі, або для них може бути не визначена схема. Для слабко-структурованих даних виконується наступна умова:

$$\forall \quad i \quad \exists \quad r_l{}^i \quad = \perp \qquad . \tag{3.20}$$

Для джерел з слабко-структурованими даними значення $SC_s = 2$.

Частково-структуровані дані. Частково-структуровані дані складаються з множини структурованих даних і множини слабко-структурованих даних, що пов'язані між собою.

Для таких даних множину елементів відношення I можна поділити на множину I_1, значення атрибутів якої не мають невизначених елементів, і множину I_2, кожен атрибут якої має хоча б одне невизначене значення:

$$\begin{aligned} &\forall\, i \in I_1 \quad r_l^i \neq \eta\,; \\ &\forall\, i \in I_2 \quad \exists\, r_l^i = \eta\,; \\ &I_1 \cap I_2 = \{\}; \\ &I_1 \cup I_2 = I. \end{aligned} \tag{3.21}$$

Структурованість таких даних обмежена знизу наступним значенням

$$ST > 1/n. \tag{3.22}$$

Для джерел з частково-структурованими даними значення $SC_s = 3$.

Неструктуровані дані, для яких не може бути не задана жодна структура. Для таких даних постають суттєві проблеми їх структурованого зберігання, оскільки вони не мають метаданих, а тому відсутня будь-яка інформація щодо типів даних і рекомендацій щодо їх оптимального зберігання.

У таких даних не можна виділити атрибутів, тому значення їх структурованості є невизначеним.

Для джерел з неструктурованими даними значення $SC_s = 4$.

Варто зазначити, що така класифікація поширюється не тільки на дані, а й на джерела даних:

- сильно (ідеально) – структурованими джерелами називаються джерела даних, що містять лише сильно (ідеально) – структуровані дані;
- частково – структуровані джерела даних містять як сильно (ідеально) – структуровані, так і слабко – структуровані дані, а також частково – структуровані дані;
- слабко – структуровані джерела містять лише слабко – структуровані дані;

- неструктуровані джерела містять лише неструктуровані дані.

Окремо можна виділити структурування даних наборів даних, пов'язаних між собою, що відповідає багатовимірній моделі даних.

Дане структурування визначається існуванням для деякої області *a* такої таблиці *t*, що пов'язана стовпчиками $c_1 . c_m$ з іншими таблицями і містить інші стовпці, що відповідають атрибутам числового типу, тобто:

$$A: \exists c_1 . c_m \in t \in A, c_i^{'} \in t^{'} \in A | < c_1, c_i^{'} > \in R_T, \quad \exists c_m . c_{m+k} \in t \in A - числові \tag{3.23}$$

Таку область ми будемо називати областю з багатовимірними даними.

Опишемо розрахунок структурованості відношень джерел даних.

Нехай задано довільне відношення $r = \{r_j =< r_1^j, r_2^j, ..., r_n^j >, j = \overline{1, m}\}$

Тоді значення його кортежів можуть бути або значеннями з домену відповідного атрибуту a_i , або невизначеними значеннями:

$$r_j^i = D(a_i) \cup \{\eta\} \tag{3.24},$$

Тоді невизначеність $u(a_i,r)$ атрибута a_i у сутності *r* визначається як кількість кортежів, на яких цей атрибут приймає невизначене значення:

$$u(a_i, r) = \left| \{ r_j \in r \mid r_j^i = \eta \} \right|. \tag{3.25}$$

Структурованість для сутності r визначається за формулою:

$$ST_r = 1 - \frac{\sum_{i=1}^{k} (n - u(a_i, r))}{m \times n}, \tag{3.26}$$

де $m \times n -$ розмір таблиці, $u(a_i,r)$ – невизначеність атрибута a_i сутності *r* (визначається як кількість кортежів, на яких цей атрибут приймає невизначене значення).

Структурованість ST_a зв'язаної області *a* визначається за формулою:

$$ST_a = \sum_{r' \in a} \frac{ST_{r'} (m_r + n_r)}{\sum_{r \in a} m_r + n}, \tag{3.27}$$

Після визначення структурованості даних отримуємо множину допустимих моделей розташування даних по носіях сховища за наступною процедурою:

1. У випадку структурованих даних допустимими моделями є M_{Rel}, M_{XML}, M_{NoSQL}.

2. У випадку слабкоструктурованих даних допустимими моделями є M_{XML}, M_{NoSQL}.

3. У випадку частково-структурованих даних за допомогою відповідних операторів алгебричної системи $e_1 = \Phi(e)$ та $e_2 = \mathrm{H}(e)$ виділяються структурована та неструктурована частини. Для структурованої частини допустимими моделями є відповідно M_{Rel}, M_{XML}, M_{NoSQL}, для слабко-структурованої - M_{XML}, M_{NoSQL}. Дані сутності e можна отримати через подання за допомогою оператора з'єднання сутностей $e=Z(e_1, e_2)$.

4. Для неструктурованих даних допустимі моделі визначаються після аналізу цих даних, інакше вони зберігаються файлами файлової системи. Дані, для яких неможливо визначити структурованість, відносяться до неструктурованих і зберігаються в файловому сховищі, яке є джерелом даних.

Допоміжні носії для розміщення даних вибираються на основі наступного принципу: вибирається та модель даних, яка може дати перевагу у виконанні певних запитів над моделлю основного носія. Для багатовимірних даних, які описуються схемою зірка (одна чи декілька таблиць пов'язані обмеженнями цілісності зі всіма іншими і містять хоча б один числовий атрибут) вибирається багатовимірна БД.

Вибір допустимих моделей здійснюється на основі порівняння часу виконання запитів у різних базах даних. Для такого порівняння оцінимо виконання операції сполучення для різних моделей даних.

Для оцінки часу виконання запитів T_{Rel} до реляційної бази даних виконується:

$$T_{Rel} \sim \sum T_q + \sum N_1 N_2 T_n + \sum N_{12} T_з ,$$

де T_q - час виконання підзапитів або отримання даних таблиць, якщо цей підзапит складається з однієї таблиці; T_n - час порівняння рядків таблиць 1 і 2; N_1 - кількість рядків підзапиту або таблиці 1; N_2 - кількість рядків підзапиту або таблиці 2; N_{12} - кількість рядків результату запиту; $T_з$ - час з'єднання рядків.

Для оцінки часу виконання запитів T_{MD} до багатовимірної даних виконується:

$$T_{MD} \sim T_ф + \sum T_в + \sum N_1 N_2 T_з ,$$

де $T_ф$ - час отримання даних з таблиці фактів; $T_в$ - час отримання даних з таблиці вимірів.

Завдяки агрегації даних у таблиці фактів виконується $T_ф < T_q$ для тих самих даних.

В інших випадках швидкодія виконання запитів до багатовимірної бази даних обмежена швидкодією виконання запитів до реляційної:

$$T_{DM} \le T_{Rel} .$$

Для оцінки часу виконання запитів T_{XML} до багатовимірної даних виконується:

$$T_{XML} = \sum T_i + \min(T_{з_1}, T_{з_2})$$
$$T_{з_1} = N_1 N_2 T_n + N_{12} T_з$$
$$T_{з_2} = (N_1 \log_2 N_1 + N_2 \log_2 N_2) T_c + (N_1 + N_2) T_n + N_{12} T_з + N_{12} \log N_{12}$$

При виконанні запитів до неструктурованих даних $T_i < T_q$, а отже і БД XML працює ефективніше:

$$T_{Rel} \ge T_{XML}$$

При виконанні запитів до структурованих даних БД швидкодія БД XML обмежена швидкодією виконання запитів даних обмежена швидкодією виконання запитів до реляційної БД: $T_{Rel} \le T_{XML}$.

Отже, в загальному випадку доцільним є вибір реляційної моделі як основної для збереження даних, крім випадків, описаних вище.

3.3.2. Вибір найкращого інваріанту

Після того, як визначені допустимі моделі для представлення сутностей сховища даних, з них вибираються оптимальні моделі на основі оцінки (2.2).

Для того, щоб отримати значення оцінки, використовується набір запитів, на яких спостерігається зміна значення критерію, а саме:

- для випадку структурованих даних – будь-який запит до сутності;
- для випадку слабко-структурованих чи частково-структурованих даних – запит із звертанням до атрибутів, які породжують слабку чи часткову структурованість;
- для випадку багатовимірних даних – запит агрегування.

Оскільки розмірність задачі низька (15 допустимих розв'язки і менше), то для вибору оптимальних моделей спочатку дані представляються всіма можливими моделями, використовуючи відповідні оператори формальної алгебричної системи, а далі виконується вибір досконалого інваріанту з всіх можливих, для чого використовується метод повного перебору.

При цьому оцінювання компонентів $U_j^P(\mu)$ та $U_j^Q(q,\mu)$ відбувається наступним чином.

1. Дані розміщуються у всіх доступних моделях з фіксацією часу (і ресурсів), витрачених на завантаження даних, на базі нього визначається компонент $U_j^P(\mu)$.

2. Перебираються всі можливі комбінації допустимих моделей, визначається витрата ресурсів на виконання запитів $U_j^Q(q,\mu)$. До неї додається відповідна вартість ресурсів, визначена на кроці 1.

3. Як розв'язок даної задачі з оцінених варіантів вибирається варіант з найменшою вартістю.

Множина інваріантів для області *a* має розмір $2^{|m_{доп}|} - 1$, де $m_{доп}$- множина всіх допустимих моделей даних для виконання запитів до даної області.

За кожною з комбінацій значень ознак розмішення $\mu = <\mu_{Rel}, \mu_{Md}, \mu_{XML}, \mu_{NoSQL}>$ формується множина моделей, які використовуються

для представлення даних заданої області (відповідні ознаки = 1). Далі для кожного інваріанту обчислюється значення оцінки (2.2) і попарним порівнянням знаходиться досконалий інваріант, який має найменшу оцінку. Щоб виконати отримані запити до сховища, задамо тотожні перетворення операцій між собою:

Оператор проекції $\pi_T(X)$ визначається наступним чином**:**

$$\pi_T(X) = \begin{cases} \pi_x(r), & \text{для РБД;} \\ \pi_x(\mathrm{E}(Cube)), & \text{для БВБД;} \\ \pi_x(V), & \text{для БД XML;} \\ \pi_x(Col), & \text{для БД NoSQL.} \end{cases}$$

Оператор селекції σ_T задається як

$$\sigma_p(e) = \begin{cases} \sigma_p(r), & \text{для РБД;} \\ Slice_p(Cube), & \text{для БВБД;} \\ \sigma_p(V), & \text{для БД XML;} \\ \sigma_p(Col), & \text{для БД XML.} \end{cases}$$

Оператор декартового добутку $\times_T$ визначається як

$$e_1 \times e_2 = \begin{cases} e_1 \times e_2 = r_1 \times r_2, & \text{для РБД;} \\ e_1 \times e_2 = \mathrm{E}(C_1) \times \mathrm{E}(C_2), & \text{для БВБД;} \\ e_1 \times e_2 = \mathrm{E}(V_1) \times \mathrm{E}(V_2), & \text{для БД XML;} \\ e_1 \times e_2 = \mathrm{E}(Col_1) \times \mathrm{E}(Col_2), & \text{для БД XML.} \end{cases}$$

Оператор об'єднання $\cup_T$ визначається як

$$e_1 \cup_T e_2 = \begin{cases} r_1 \cup r_2, & \text{для РБД} \\ \mathrm{E}(Cube_1) \cup \mathrm{E}(Cube_2), & \text{для БВБД} \\ V_1 \cup V_2, & \text{для БД XML} \\ \mathrm{E}(Col_1) \cup \mathrm{E}(Col_2), & \text{для БД NoSQL} \end{cases}$$

Оператор перетину $\cap_T$ визначається як

$$e_1 \cap_T e_2 = \begin{cases} r_1 \cap r_2, & \text{для РБД;} \\ \mathrm{E}(C_1) \cap \mathrm{E}(C_2), & \text{для БВБД;} \\ V_1 \cap V_2, & \text{для БД XML;} \\ \mathrm{E}(Col_1) \cap \mathrm{E}(Col_2), & \text{для БД NoSQL.} \end{cases}$$

Оператор природного з'єднання (злиття) $\triangleright\triangleleft_T$ визначається як

$$e_1 \triangleright\triangleleft_T e = \begin{cases} e_1 \triangleright\triangleleft_X e_2 = r_1 \triangleright\triangleleft_X r_2, & \text{для РБД;} \\ e_1 \triangleright\triangleleft_X e_2 = \mathrm{E}(C_1) \triangleright\triangleleft_X \mathrm{E}(C_2), & \text{для БВБД;} \\ e_1 \triangleright\triangleleft_X e_2 = \mathrm{E}(V_1) \triangleright\triangleleft_X \mathrm{E}(V_2), & \text{для БД XML;} \\ e_1 \triangleright\triangleleft_X e_2 = Col_1 \triangleright\triangleleft_X Col_2, & \text{для БД NoSQL, якщо документи } Col_2 \text{ вкладені в } Col_1 \text{ або навпаки;} \\ e_1 \triangleright\triangleleft_X e_2 = \mathrm{E}(Col_1) \triangleright\triangleleft_X \mathrm{E}(Col_2), & \text{для БД NoSQL, якщо } Col_1 \text{ і } Col_2 \text{ не пов'язані.} \end{cases}$$

Оператор різниці $\setminus_T$ визначається як

$$e_1 \setminus_T e_2 = \begin{cases} r_1 - r_2, & \text{для РБД;} \\ \mathrm{E}(C_1) \setminus \mathrm{E}(C_2), & \text{для БВБД;} \\ \mathrm{E}(V_1) \setminus \mathrm{E}(V_2), & \text{для БД XML;} \\ \mathrm{E}(Col_1) \setminus \mathrm{E}(Col_2) & \text{для БД NoSQL.} \end{cases}$$

Оператор видалення дублікатів δ_T визначається як

$$\delta_T(T) = \begin{cases} \delta(V), & \text{для РБД;} \\ C \text{ (дані видаються без дублікатів)}, & \text{для БВБД;} \\ \delta(V), & \text{для БД XML;} \\ \delta(T(Col)), & \text{для БД NoSQL.} \end{cases}$$

Якщо зібраної статистики запитів немає, то запити генеруються за наступною рекурсивною процедурою:

1. З ймовірністю 50% в корені дерева вибирається довільним чином або сутність, або операція відповідно із множин елементів або сутностей. Якщо робиться запит до багатовимірних даних – вибирається операція агрегування на відповідній таблиці фактів і пов'язаних таблицях вимірів;

2. Довільним чином вибираються аргументи *i* для кожного з них рекурсивно повторюється крок 1.

3. Довільним чином залежно від оператора вибираються параметри (умови вибору, стовпці і т.д.). Якщо запит проводиться до слабко-структурованих чи частково-структурованих даних, то вибирається один чи декілька атрибутів, що породжують слабку структурованість.

4. Процес завершується, коли на всіх листових елементах дерева запиту вибираються сутності, а не операції. Крім того, задається максимальна глибина запиту d_{max}, на якій обов'язково вибирається сутність, а не операція.

3.4. Модифікований генетичний алгоритм

Для вибору конфігурації розташування та реплікації даних у сховищі, розв'язуються задачі оптимізації (2.3) - (2.5) і (2.6) - (2.7) за допомогою генетичного алгоритму з використанням фази попереднього формування початкової популяції бджолиними алгоритмом. Дана додаткова фаза введена для отримання більш якісної (ближчої до оптимуму) початкової популяції, при якій генетичному алгоритму потрібно менше ітерацій.

Евристичні алгоритми вибрані тому, що функції використання ресурсів, які є компонентами критерію вартості, не є аналітичними функціями, а в силу потенційно великої області допустимих рішень використання точних алгоритмів займає набагато більше часу.

Даний модифікований генетичний алгоритм має наступні характеристики:

- селекція виконується методом рулетки;
- схрещування: блокове (блоком вважається набір ознак , що відповідають одному вузлу), з імовірністю схрещування двох особин 0.8 та імовірністю обміну значеннями блоків $p_c = \frac{(i-1)}{n_N} + \frac{1}{n_N \cdot 2}$, де i-номер вузла, n_N-кількість вузлів;
- мутація: багатоточкова, з ймовірністю $p_m = \frac{n_A(n_N - 1)}{n_A \cdot n_N \cdot 2}$, де n_A-кількість областей сховища.

Фаза формування початкової популяції має наступні характеристики:

- кількість початкових хромосом : $n_П = n_N * 10$;
- кількість кращих хромосом : $n_к = \frac{n_П}{2}$;
- кількість перспективних хромосом : $n_п = \left[\frac{n_П}{4}\right]$;
- кількість додаткових хромосом : $n_д = \left[\frac{n_П}{4}\right]$;
- відстань області для кращих хромосом : $d_к = n_A(n_N - 2)$;

- відстань області для перспективних хромосом : $d_n = n_A(n_N - 1)$.

Цей алгоритм записується наступним чином:

Вхідні дані: кількість областей сховища, кількість вузлів , набір запитів Q.

Вихідні дані: конфігурація розміщення областей по вузлах сховища з оптимальною вартістю.

1. Вибрати $n_П = n_N * 10$ початкових хромосом випадковим чином і додати її до поточного покоління.

2. Оцінити функцію придатності цих особин.

3. З отриманих значень вибрати $n_к = \frac{n_П}{2}$ кращих та $n_n = \left[\frac{n_П}{4}\right]$ перспективних ділянок, а також $n_д = \left[\frac{n_П}{4}\right]$ точок.

3.1. Для кращих ділянок – вибрати випадкових $n_к = \left[\frac{n_П}{4}\right]$ хромосом з відстанню Хеммінга не більше $d_к = n_A(n_N - 2)$.

3.2. Для перспективних ділянок – вибрати $n_n = \frac{n_П}{2}$ хромосом з відстанню Хеммінга не більше $d_n = n_A(n_N - 1)$.

3.3. $n_д = \left[\frac{n_П}{4}\right]$ додаткових хромосом вибрати випадковим чином.

4. Оцінити ці хромосоми за допомогою критерію вартості , отримавши популяцію особин.

5. Виконати селекцію методом рулетки, отримаємо особини i та i_2.

6. Провести схрещування, отримати особину i'.

7. Здійснити мутацію і отримати i'_m.

8. Знайти особу з найменшою придатністю. Якщо $f_k^* - f_{k+1}^* < \varepsilon$, перейти на крок 3, інакше перейти на крок 9.

9. Завершення алгоритму. f_{k+1}^* - є оптимальним значенням.

Процес синтезу на основі генетичного алгоритму повторюється до тих пір, поки буде досягнутий певний рівень збіжності (на протязі двох поколінь значення цільової функції (ЦФ) не змінюється із заданою точністю).

При цьому збіжність алгоритму випливає зі збіжності стандартного генетичного алгоритму. Згідно теореми схем Голланда [56] кількість індивідів, що зберігають певний шаблон (сукупність генів, які потрібно зберегти в наступній популяції), зростає для кожної наступної популяції, що забезпечує отримання нових особин зі збереженням цього шаблону та кращим значенням ЦФ. У підсумку це дозволяє знайти оптимальне значення ЦФ.

Часова складність пропонованого алгоритму є $o(n^2)$, оскільки він складається з фази бджолиного і фази генетичного, а часова складність цих алгоритмів є $o(n^2)$.

3.5. Експериментальні дослідження

У рамках дисертаційної роботи були проведені дослідження впливу параметрів модифікованого генетичного алгоритму вибору розташування даних і маршруту реплікації даних на ефективність і збіжність цього алгоритму.

Метою даних експериментів було визначити, наскільки ефективним є запропонований алгоритм оптимізації МБСД в порівнянні з відомими методами розв’язку даної задачі, такими як стандартний генетичний алгоритм, бджолиний алгоритм , метод повного перебору, метод гілок і меж.

Експериментальні дослідження пропонованого модифікованого генетичного алгоритму включали наступне.

1. Дослідження впливу фази формування початкової популяції на час виконання алгоритму та вартість побудованого сховища, зокрема значень параметрів цієї фази.
2. Дослідження впливу фази генетичного пошуку на час виконання алгоритму та вартість побудованого сховища, зокрема значень параметрів цієї фази.
3. Порівняння алгоритму з відомими методами розв’язку задачі.

3.5.1. Умови та порядок проведення досліджень

Експериментальні дослідження проводилися шляхом моделювання значення ЦФ на сховищі, що складається з одного вузла з наступними характеристиками:

- ЦП: Intel Core 2 Quad 3 ГГц;
- ОЗП: 8 ГБ ;
- ОС: Windows Server 2012 R2;
- реляційна БД: Microsoft SQL Server;
- багатовимірна БД: Hyperion Essbase;
- БД XML: BaseX;
- БД NoSQL: MongoDB.

Для проведення дослідження запити виконувалися наступним чином.

Запити до сховища формулюються на мові SQL без синтаксичних відмінностей, але з семантичною відмінністю – об'єкти, що запитуються (блок FROM) можуть відповідати як за структуровані дані, так і за не структуровані.

При цьому результатами (під)запитів до структурованих даних є відношення, що представляються таблицями МБСД. Результатами операцій до слабко-структурованих даних є документи JSON.

Для того, щоб забезпечити виконання запитів між даними різного рівня структурованості, застосовується наступний порядок виконання запитів:

1. Побудова дерева запиту згідно реляційної теорії. Наприклад, запит SELECT x FROM y WHERE z<c записується як $\pi_x \sigma_{z<c} y$;

2. Обробка всіх підзапитів, які можуть бути виконані повністю носіями даних (обхід дерева знизу вгору і виділення всіх піддерев, які мають сутності лише одної моделі даних):

2.1.виділення підзапитів;

2.2.перетворення запиту до сутності в запит до моделі даних (3.5.2 , 3.5.3.);

2.3.виконання підзапиту;

2.4.перетворення отриманих результатів з моделей даних носїів в реляційну модель сховища (3.5.1.);

3. Виконання отриманого запиту в моделі даних сховища.

Час отримання користувачем результату запиту залежить від наступних складових:

$$t = t_{од} + t_{пд\_МСД} = f(t_{о\_МСД}, t_{о_1}, .. t_{о_n}, t_{пд_1}, .. t_{пд_n}) + t_{пд\_МСД}$$, де:

де $t_{од}$ – час отримання результату запиту у сховищі даних; $t_{о\_МСД}$ – час виконання операцій запиту двигуном МБСД; $t_{о_i}$ – час виконання операцій двигуном носія даних *i*; $t_{пд}$ – час передачі даних з носія даних і до двигуна сховища; $T_{пд\_МСД}$ – час передачі даних від служб доступу до сховища до клієнтського застосування.

Серед цих компонент час отримання результату запиту, час виконання операцій двигуном носія даних *i*, час передачі даних з носія даних і до двигуна сховища залежать від набору вузлів та носіїв, в яких містяться дані, потрібні для виконання результату запиту.

Визначення 3.9. Запізнювання запиту – це величина, відносна до часу виконання запиту t_q , що визначається як $(t_q - t_{\lim}) / t_{\lim} \mid t_{\lim} = f(s_q)$.

Якщо один чи більше запитів мають запізнювання >1, то сховище даних вважається таким, запити до якого запізнюються, та ініціюється процес повторної побудови сховища для усунення запізнювання.

Для так званого точкового дослідження, при якому вибирається задача конкретної розмірності і встановлюються початкові значення параметрів для проведення цього дослідження:

- кількість вузлів – $n_N = 3$;
- кількість областей – $n_A = 3$;
- розмірність задачі - $n_N \cdot n_A = 9$, ($2^{n_N \cdot n_A} = 512$ можливих розв’язків).

Параметри фази формування початкової популяції задані наступним чином:

- початкових хромосом – $n_П = 3 * 10 = 30$;

- кращих хромосом – $n_к = \frac{30}{2} = 15$;
- перспективних хромосом - $n_п = \left[\frac{30}{4}\right] = 7$;
- додаткових хромосом - $n_д = \left[\frac{30}{4}\right] = 7$;
- радіус області для кращих хромосом – $d_к = 3(3-2) = 3$;
- радіус області для перспективних хромосом – $d_п = 3(3-1) = 6$.

Параметри основної фази пошуку рішення встановимо наступним чином:

- селекція – метод рулетки;
- схрещування - блокове (блоком вважається набір ознак , що відповідають одному вузлу), з імовірністю схрещування двох особин 0.8 та імовірністю обміну значеннями блоків $\frac{1}{3 \cdot 2} = \frac{1}{6}$ для 1-го, $\frac{1}{3} + \frac{1}{3 \cdot 2} = \frac{1}{2}$ - 2-го, $\frac{2}{3} + \frac{1}{3 \cdot 2} = \frac{5}{6}$ - 3-го вузла;
- мутація - багатоточкова, з імовірністю $p_м = \frac{3(3-1)}{3 \cdot 3 \cdot 2} = \frac{1}{3}$;
- редукція – прибираються всі гірші хромосоми, якщо кількість особин перевищує початкову.

В наступних пунктах наведені результати проведених досліджень для цих вхідних даних.

3.5.2. Дослідження параметрів фази формування початкової популяції на час виконання алгоритму та вартість побудованого сховища

Використовуючи вищенаведені параметри, була досліджена залежність значення ЦФ та часу від кількості «кращих областей» та вибраних хромосом в них для різних кількостей початкових хромосом.

Результати для значення цільової функції при 10 початкових хромосомах відображено на рис 3.2. Час виконання алгоритму, за який отримані ці результати, наведений на рис 3.3.

Тут і далі для діаграм значень цільової функції - значення ЦФ подане у грн., на осі абсцис – кількість областей у % від кількості початкових хромосом, різні стовпці гістограми відображають різні кількості вибраних хромосом областей.

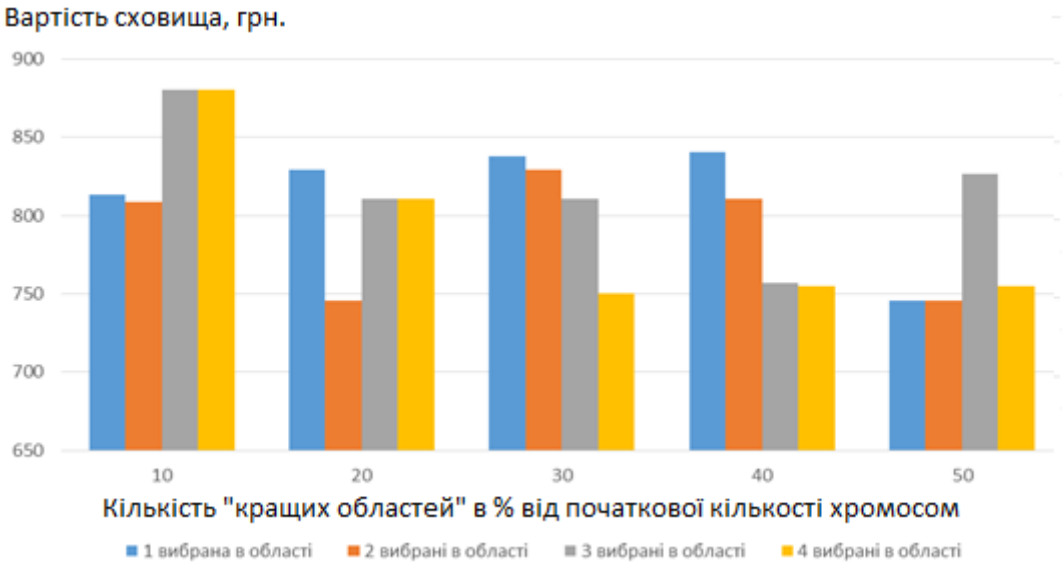


Рис.3.2. Діаграма залежності значення цільової функції від кількості «кращих областей» та вибраних хромосом у цих областях при 10 початкових хромосомах. [Розроблено автором]

Рису

Рис.3.3. Діаграма залежності значення часу виконання початкової фази алгоритму від кількості «кращих областей» та вибраних хромосом у цих областях при 10 початкових хромосомах. [Розроблено автором]

Згідно наведених діаграм, оптимальне значення з найменшим часом досягається при 50% «кращих областей» і одній вибраній хромосомі для кожної області.

Аналогічний експеримент для 20 початкових хромосом дав наступний результат для цільової функції (рис 3.4.):

Рис.3.4. Діаграма залежності значення цільової функції від кількості «кращих областей» та вибраних хромосом у цих областях при 20 початкових хромосомах. [Розроблено автором]

Ці результати отримані за наступний час (рис 3.5.):

Рис.3.5. Діаграма залежності значення часу виконання алгоритму від кількості «кращих областей» та вибраних хромосом у цих областях при 20 початкових хромосомах. [Розроблено автором]

Згідно наведених діаграм, оптимальне значення з найменшим часом досягається при 40% «кращих областей» і одній вибраній хромосомі для кожної області.

Аналогічний експеримент для 30 початкових хромосом дав наступний результат для цільової функції (рис 3.6.):

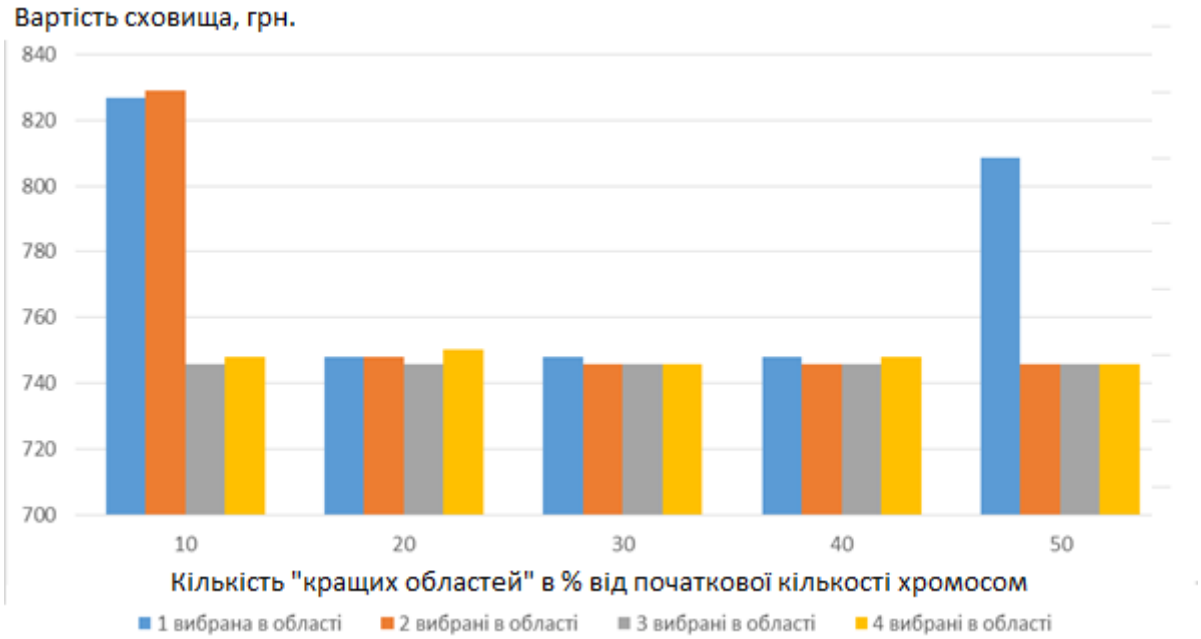


Рис.3.6. Діаграма залежності значення цільової функції від кількості «кращих областей» та вибраних хромосом у цих областях при 30 початкових хромосомах. [Розроблено автором]

Ці результати отримані за час, проілюстрований діаграмою на рис. 3.7.

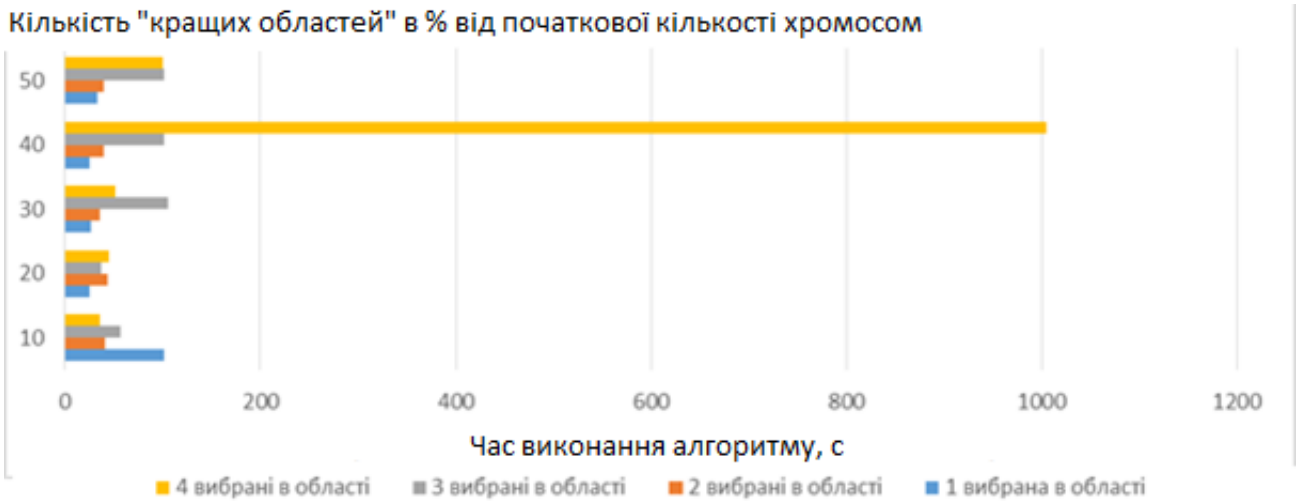


Рис.3.7. Діаграма залежності значення часу виконання алгоритму від кількості «кращих областей» та вибраних хромосом у цих областях при 30 початкових хромосомах. [Розроблено автором]

Згідно наведених діаграм, оптимальне значення з найменшим часом досягається при 20% «кращих областей» і трьох вибраних хромосомах для кожної області.

Аналогічний експеримент для 50 початкових хромосом дав результат для цільової функції, відображений діаграмою на рис 3.8. Результат для часу наведено на рис. 3.9.

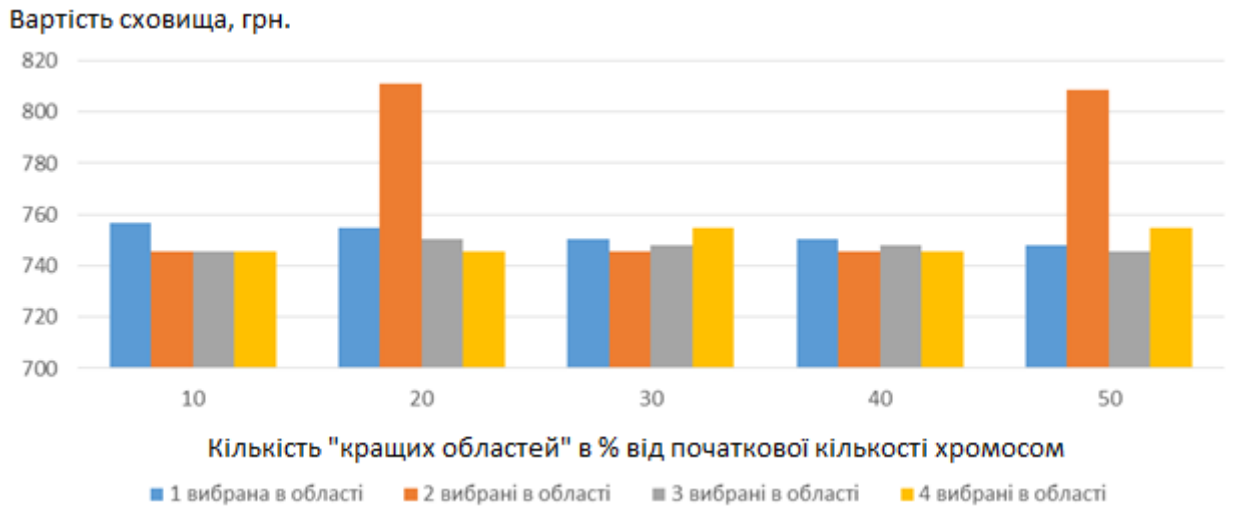


Рис.3.8. Діаграма залежності значення цільової функції від кількості «кращих областей» та вибраних хромосом у цих областях при 40 початкових хромосомах. [Розроблено автором]

Рис.3.9. Діаграма залежності значення часу виконання алгоритму від кількості «кращих областей» та вибраних хромосом у цих областях при 50 початкових хромосомах. [Розроблено автором]

Згідно наведених діаграм (рис. 3.8, 3.9), оптимальне значення з найменшим часом досягається при 50% «кращих областей» і одній вибраній хромосомі для кожної області.

Результат для цільової функції аналогічного експерименту для 50 початкових хромосом представлено на рис 3.10.

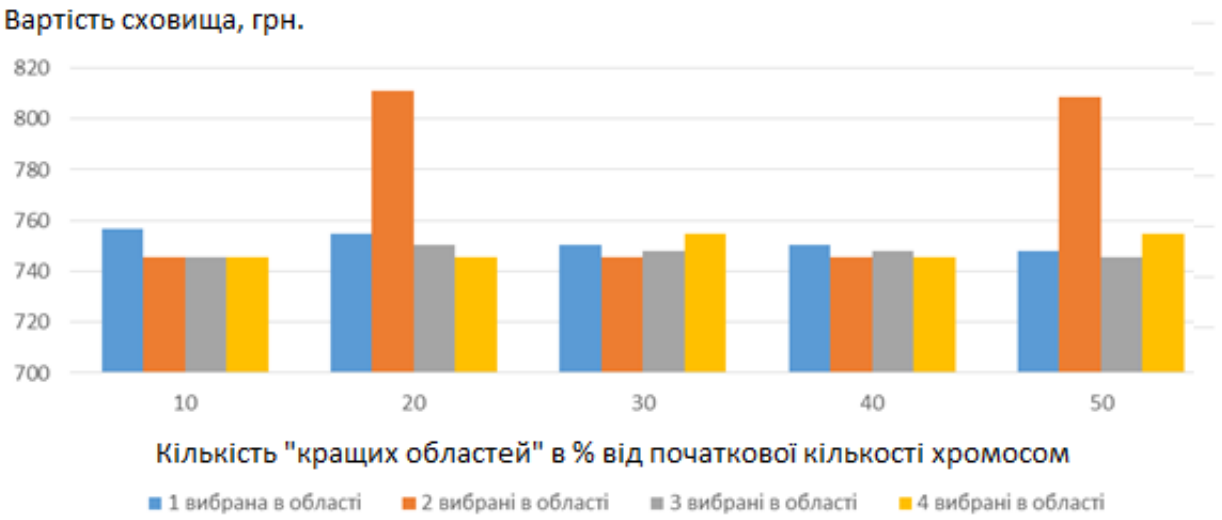


Рис.3.10. Діаграма залежності значення цільової функції від кількості «кращих областей» та вибраних хромосом у цих областях при 50 початкових хромосомах. [Розроблено автором]

Ці результати отримані за наступний час (рис 3.11.):

Рис.3.11. Діаграма залежності значення часу виконання алгоритму від кількості «кращих областей» та вибраних хромосом у цих областях при 50 початкових хромосомах. [Розроблено автором]

Згідно наведених діаграм, оптимальне значення з найменшим часом досягається при 30% «кращих областей» і двох вибраних хромосомах для кожної області.

Результат аналогічного експерименту для цільової функції для 60 початкових хромосом відображений діаграмою на рис 3.12, а результат для часу - на рис. 3.13.

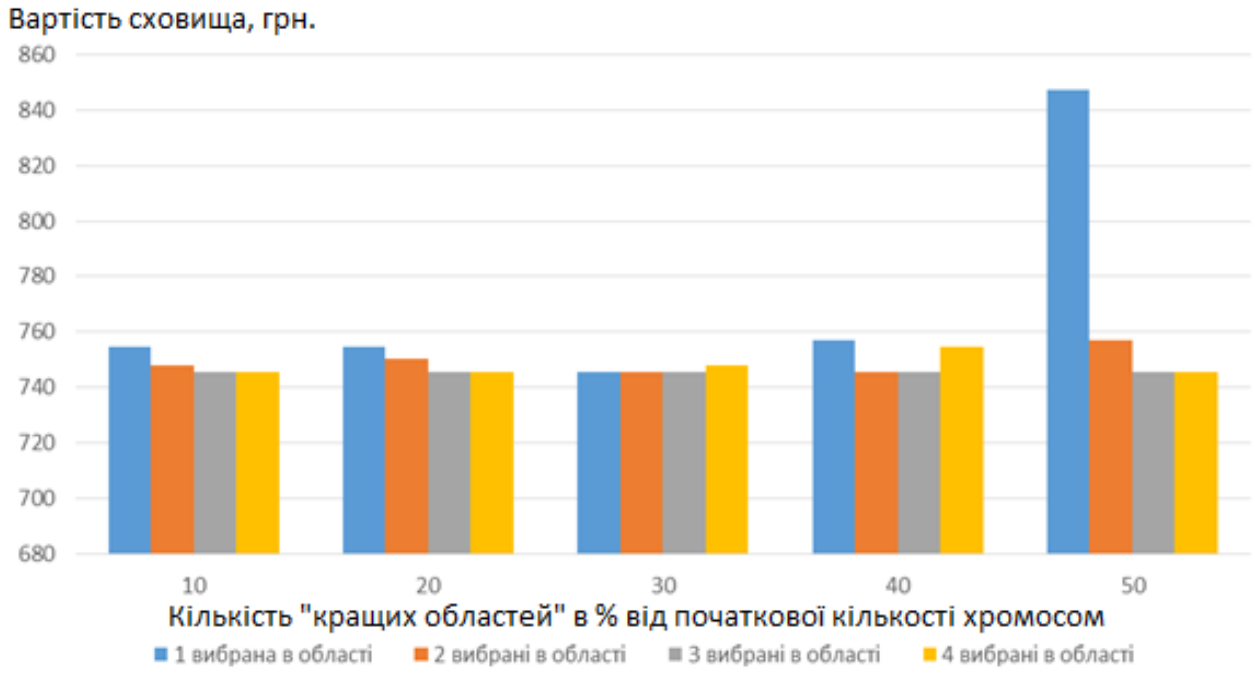


Рис.3.12. Діаграма залежності значення цільової функції від кількості «кращих областей» та вибраних хромосом у цих областях при 60 початкових хромосомах. [Розроблено автором]

Рис.3.13. Діаграма залежності значення часу виконання алгоритму від кількості «кращих областей» та вибраних хромосом у цих областях при 60 початкових хромосомах. [Розроблено автором]

Згідно цих діаграм, оптимальне значення з найменшим часом досягається при 30% «кращих областей» і одній вибраній хромосомі для кожної області.

Аналогічний експеримент для 70 початкових хромосом дав результат для цільової функції, відображений діаграмою на рис 3.14. Результат для часу наведено на рис. 3.15.

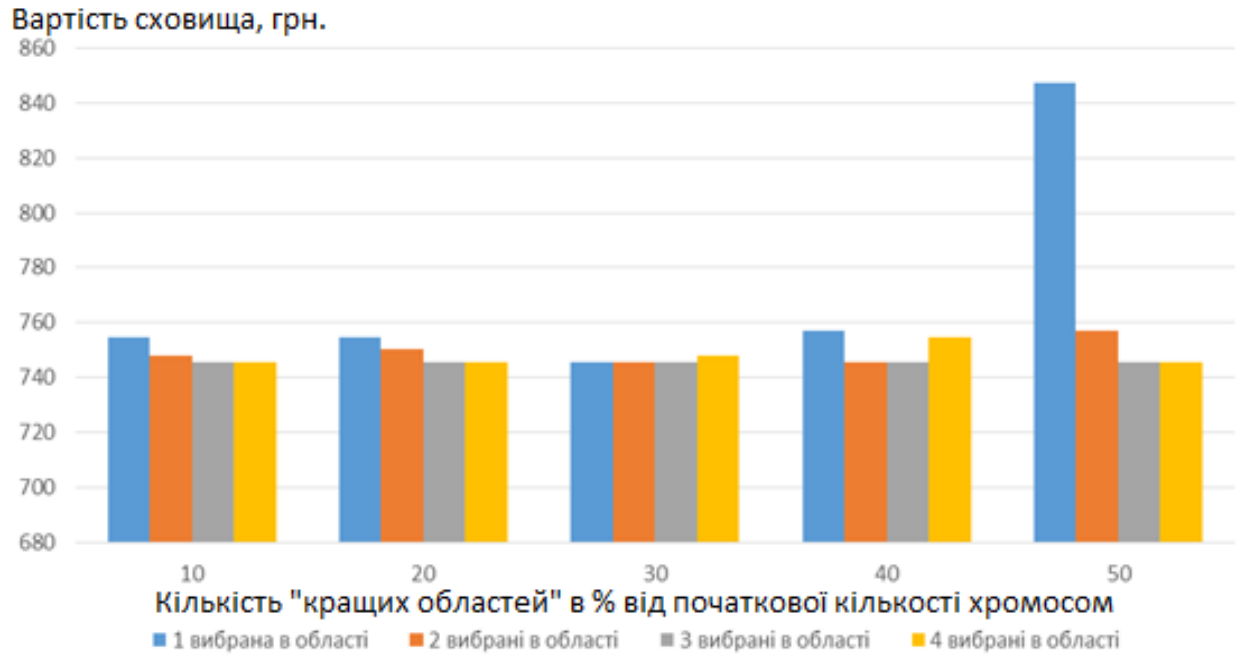


Рис.3.14. Діаграма залежності значення цільової функції від кількості «кращих областей» та вибраних хромосом у цих областях при 70 початкових хромосомах. [Розроблено автором]

Рис.3.15. Діаграма залежності часу виконання алгоритму від кількості «кращих областей» та вибраних хромосом у цих областях при 70 початкових хромосомах. [Розроблено автором]

Згідно цих діаграм, оптимальне значення з найменшим часом також досягається при 30% «кращих областей» і одній вибраній хромосомі для кожної області.

Аналогічний експеримент для 80 початкових хромосом дав наступний результат для цільової функції (рис 3.16.). Результат для часу наведено на рис. 3.17.

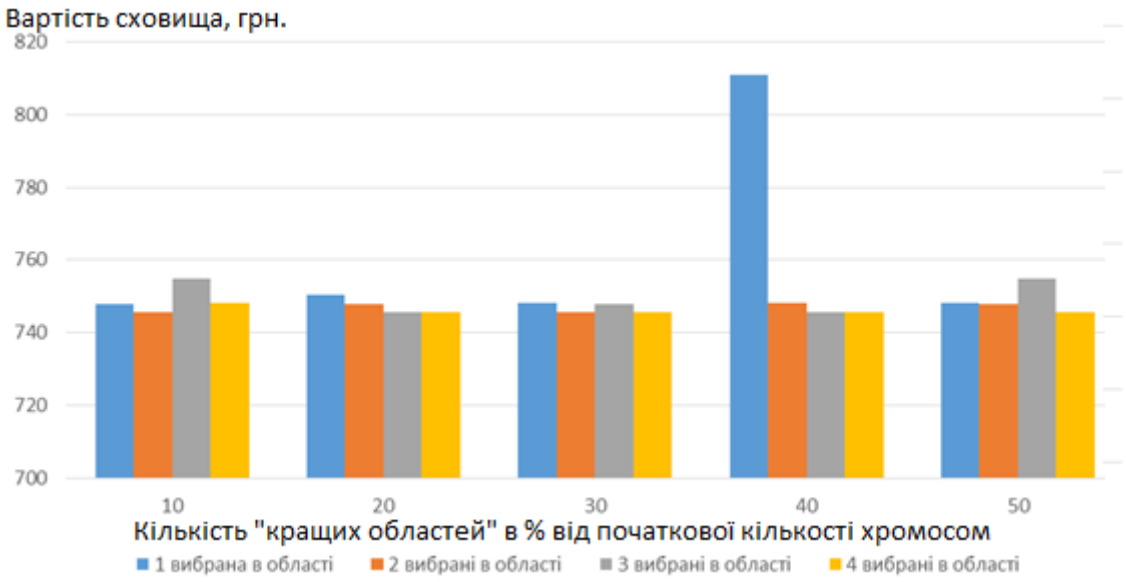


Рис.3.16. Діаграма залежності значення цільової функції від кількості «кращих областей» та вибраних хромосом у цих областях при 80 початкових хромосомах. [Розроблено автором]

Рис.3.17. Діаграма залежності часу виконання алгоритму від кількості «кращих областей» та вибраних хромосом у цих областях при 80 початкових хромосомах. [Розроблено автором]

Згідно наведених діаграм (рис. 3.16, 3.17), оптимальне значення з найменшим часом досягається при 50% «кращих областей» і одній вибраній хромосомі для кожної області.

Аналогічний експеримент для 90 початкових хромосом дав наступний результат для цільової функції (рис 3.18.). Результат для часу наведено на рис. 3.19.

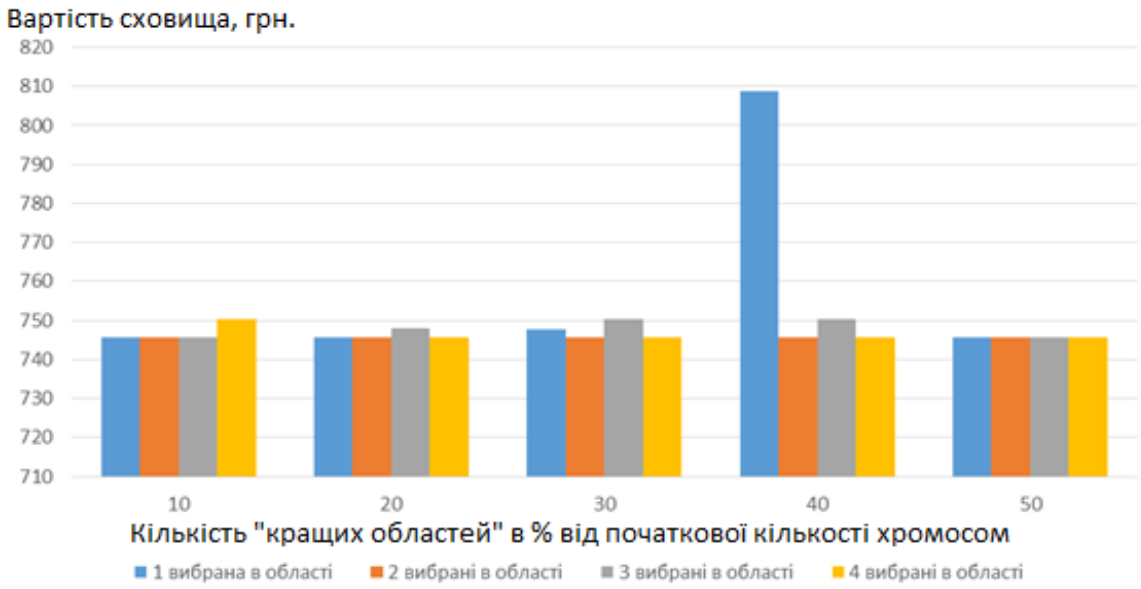


Рис.3.18. Діаграма залежності значення цільової функції від кількості «кращих областей» та вибраних хромосом у цих областях при 90 початкових хромосомах. [Розроблено автором]

Рис.3.19. Діаграма залежності значення цільової функції від кількості «кращих областей» та вибраних хромосом у цих областях при 90 початкових хромосомах. [Розроблено автором]

Згідно цих діаграм, оптимальне значення з найменшим часом досягається при 10% «кращих областей» і одній вибраній хромосомі для кожної області.

Аналогічний експеримент для 100 початкових хромосом дав наступний результат для цільової функції (рис 3.20.). Результат для часу наведено на рис. 3.21.

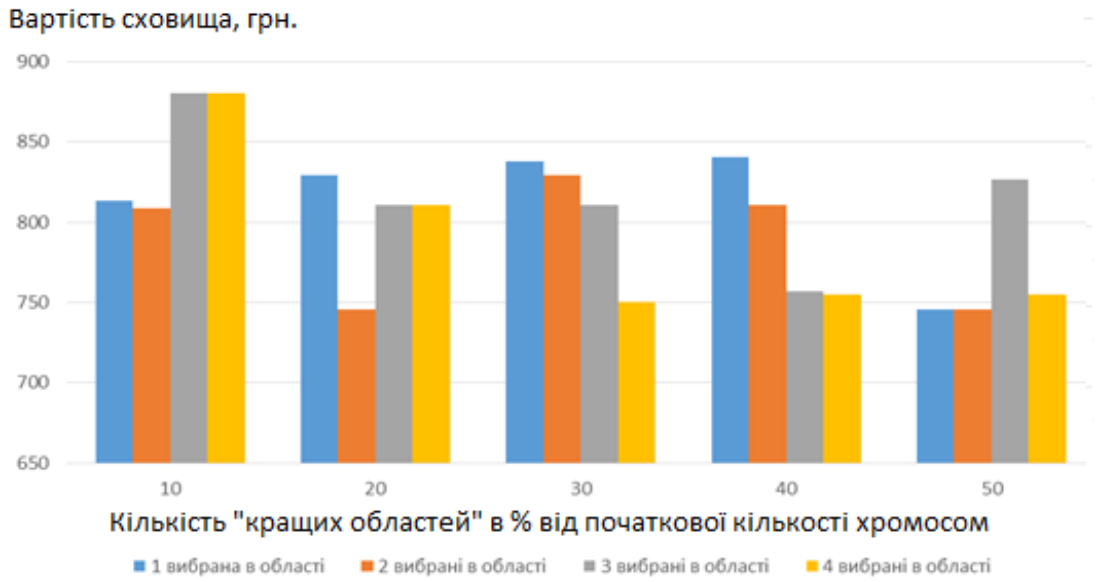


Рис.3.20. Діаграма залежності значення цільової функції від кількості «кращих областей» та вибраних хромосом у цих областях при 100 початкових хромосомах. [Розроблено автором]

Рис.3.21. Діаграма залежності часу виконання алгоритму від кількості «кращих областей» та вибраних хромосом у цих областях при 90 початкових хромосомах. [Розроблено автором]

Згідно наведених діаграм (рис. 3.20, 3.21), оптимальне значення з найменшим часом досягається при 50% «кращих областей» і двох вибраних хромосомах для кожної області.

Узагальнюючи ці результати, отримаємо, шо найкращий результат цільової функції при оптимальному часі виконання алгоритму досягається при 10 початкових хромосомах, 5 областях і при 1 хромосомі на кожну область.

При збільшенні кількості початкових хромосом алгоритм краще сходиться до оптимуму, однак це потребує більше часу. Збільшення кількості

областей та кількості вибраних хромосом також сприяє швидшій збіжності і збільшує час виконання алгоритму пропорційно добутку кількості областей і вибраних хромосом, тому доцільно вибирати кількість хромосом не більшу за початкову кількість.

При збільшенні розмірності задачі потрібно збільшувати кількість початкових хромосом, ця залежність розглянута в п.3.5.4.

3.5.3. Дослідження параметрів генетичного пошуку на час виконання алгоритму та вартість побудованого сховища

Фаза генетичного пошуку відповідає за знаходження оптимального значення ЦФ, починаючи свою роботу з початкової популяції, ближчої до оптимуму, ніж випадково вибрана.

Для ефективності алгоритму ця фаза повинна збігатися на малій кількості ітерацій, що забезпечується оптимальним вибором параметрів генетичного алгоритму – імовірностей схрещування і мутації.

Тому було проведено дослідження залежності часу виконання фази генетичного пошуку від значень її параметрів.

У даному алгоритмі схрещування виконує локальний пошук, тому було вибрано блокове схрещування, причому зі збільшенням номеру блока зростає імовірність, що цей блок буде обміняний в процесі схрещування.

Результати досліджень імовірності мутації наведені на рис 3.22.

Мутація дозволяє «виходити» з зон збіжності до локального оптимуму. Використовується багатоточкова мутація, тобто для кожної хромосоми при мутації кожен ген оцінюється окремо. Імовірність мутації росте з розміром задачі, дозволяючи таким чином більшій кількості генів бути зміненими.

Результати досліджень імовірності мутації наведені на рис 3.23-3.25.

Проведемо дослідження потрібної кількості початкових хромосом для задач різної розмірності.

На рис. 3.23 - 3.25 значення ЦФ подане у грн., час виконання алгоритму – у с.

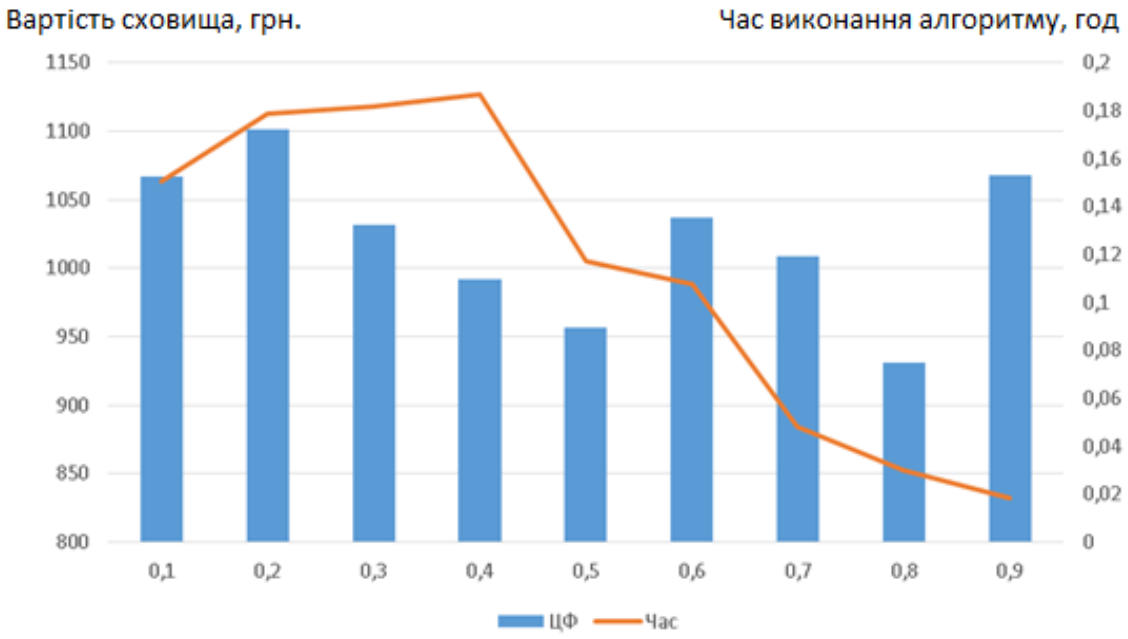


Рис.3.22. Діаграма залежності значення цільової функції та часу виконання алгоритму від імовірності схрещування. [Розроблено автором]

Рис.3.23. Діаграма залежності значення цільової функції та часу виконання алгоритму від імовірності мутації для довжини хромосоми 9. [Розроблено автором]

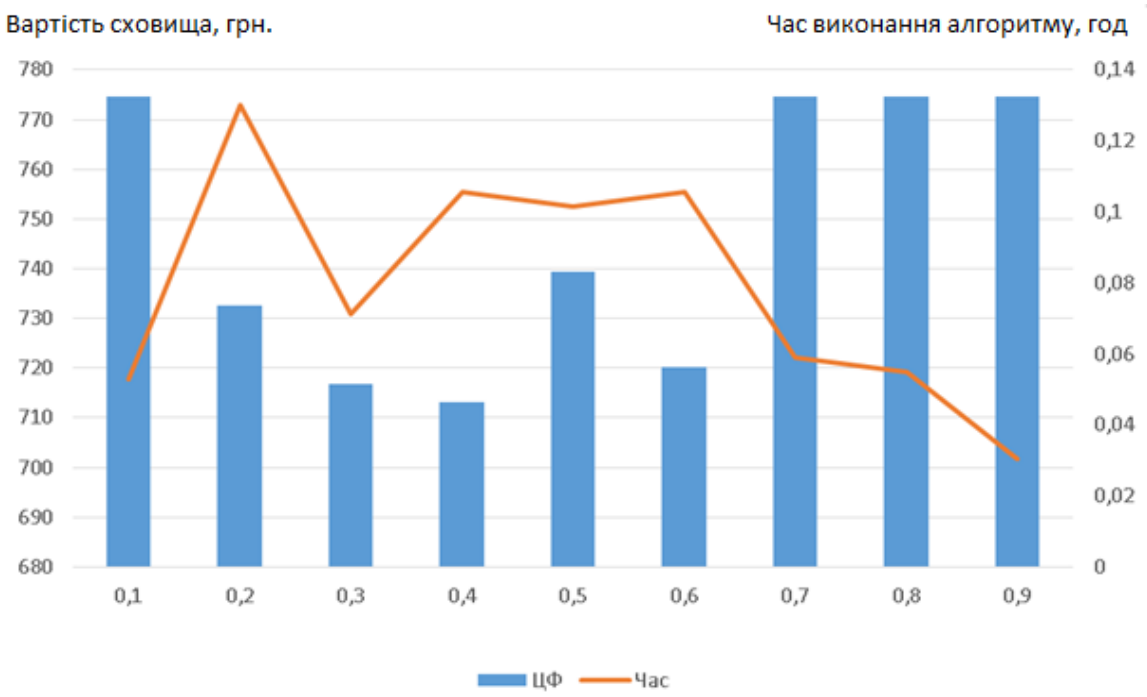


Рис.3.24. Діаграма залежності значення цільової функції та часу виконання алгоритму від імовірності мутації для довжини хромосоми 25. [Розроблено автором]

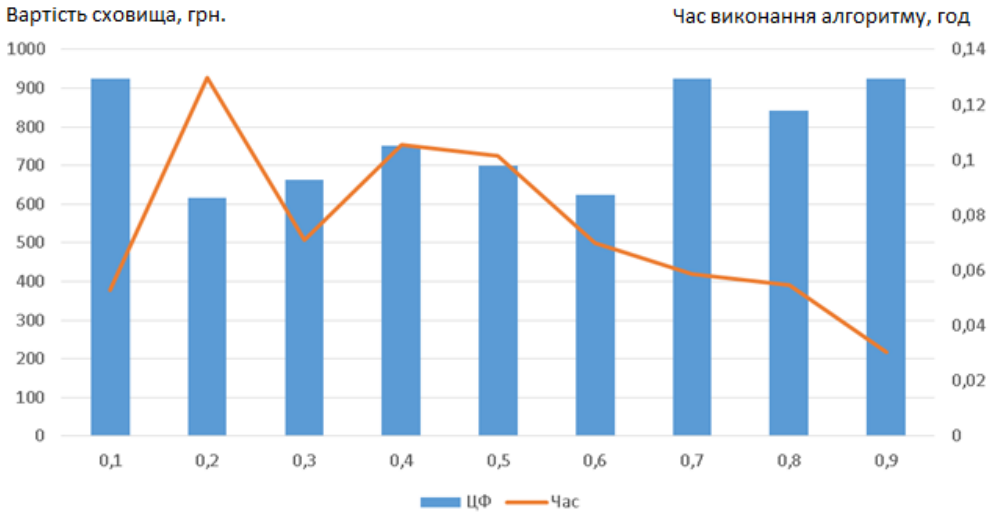


Рис.3.25. Діаграма залежності значення цільової функції та часу виконання алгоритму від імовірності мутації для довжини хромосоми 36. [Розроблено автором]

З наведених результатів (рис 3.23-3.25) бачимо, що задані параметри, а саме: імовірність схрещування двох особин 0.8 та імовірність мутації $p_m = \frac{n_A(n_N - 1)}{n_A \cdot n_N \cdot 2}$ дозволяє отримати мінімальне значення цільової функції за мінімальний час.

3.5.4. Дослідження параметрів алгоритму в залежності від розмірності задачі

Для ефективного виконання алгоритму було проведено дослідження залежності кількості початкових хромосом від розмірності задачі, прийнявши наступні усталені значення:

- кількість «кращих» областей – 50%;
- кількість «потенційних» областей – 25%;
- кількість хромосом у «кращих» областях – 1;
- кількість хромосом у «потенційних» областях – 1;
- кількість додаткових хромосом : 25%;
- імовірність схрещування : 0.8;
- імовірність мутації – 0.2.

Для рис. 3.26-3.28 значення ЦФ подане у грн., час виконання алгоритму – у мс.

Для довжини хромосоми N=9 отримано наступні результати (рис 3.26).

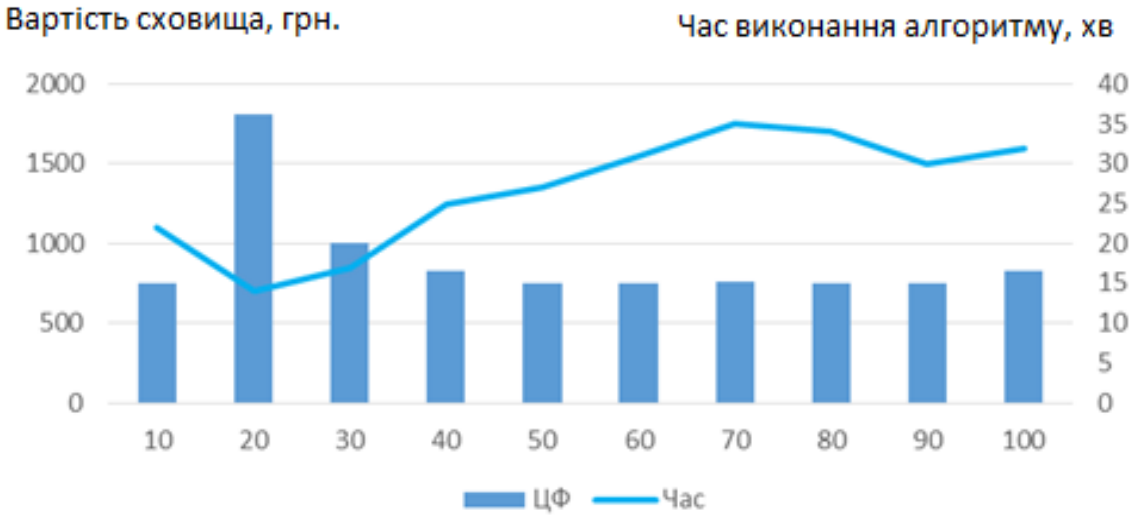


Рис.3.26. Діаграма залежності значення цільової функції та часу виконання алгоритму від розміру популяції для довжини хромосоми 9.
[Розроблено автором]

Згідно вищенаведеної діаграми, мінімум цільової функції досягається при 10 початкових хромосомах, при подальшому рості кількості хромосом росте час виконання алгоритму, однак значення ЦФ не опускається нижче значення для 10 хромосом.

Для довжини хромосоми N=25 отримано наступні результати (рис 3.27):

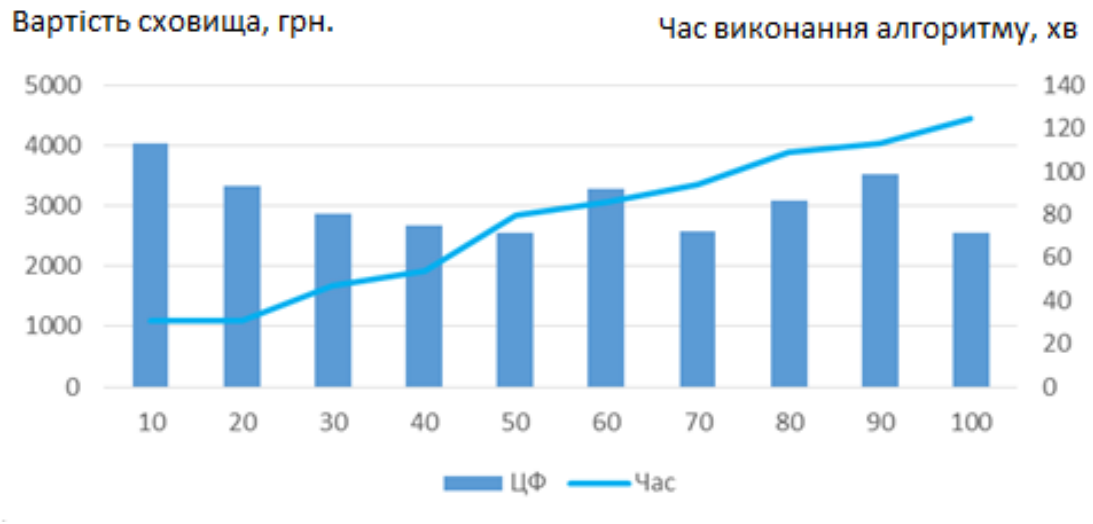


Рис.3.27. Діаграма залежності значення цільової функції та часу виконання алгоритму від розміру популяції для довжини хромосоми 25.
[Розроблено автором]

Згідно вищенаведеної діаграми, мінімум цільової функції досягається при 50 початкових хромосомах, при подальшому рості кількості хромосом спостерігаємо значення ЦФ, яке не опускається нижче значення для 10 хромосом.

Для довжини хромосоми N=100 отримано наступні результати (рис 3.28):

Рис.3.28. Діаграма залежності значення цільової функції та часу виконання алгоритму від розміру популяції для довжини хромосоми 10.
[Розроблено автором]

Згідно вищенаведеної діаграми, в силу високої кількості можливих розв’язків, мінімум цільової функції досягається при 800 початкових хромосомах, однак час виконання алгоритму є відносно великим.

Ґрунтуючись на вищенаведеному, можна ствердити, що для більшої розмірності задачі потрібен більший розмір популяції.

3.5.5. Порівняння методу побудови РСД ГТ з існуючими методами та алгоритмами

Для того, щоб перевірити якість запропонованого алгоритму, була проведена серія випробувань часу виконання алгоритму при різній розмірності задачі.

Порівняння запропонованого (модифікованого генетичного) алгоритму, що виконувався при значеннях параметрів, вказаних у попередньому пункті, проводилося з:

- бджолиним алгоритмом, з відповідними параметрами;
- генетичним алгоритмом, з відповідними параметрами;
- методом повного перебору.

Перевірка всіх алгоритмів проводилася до того моменту, коли буде знайдений оптимум (всі алгоритми збіглися до оптимуму). Результат відображений на рис. 3.29 та рис. 3.30.

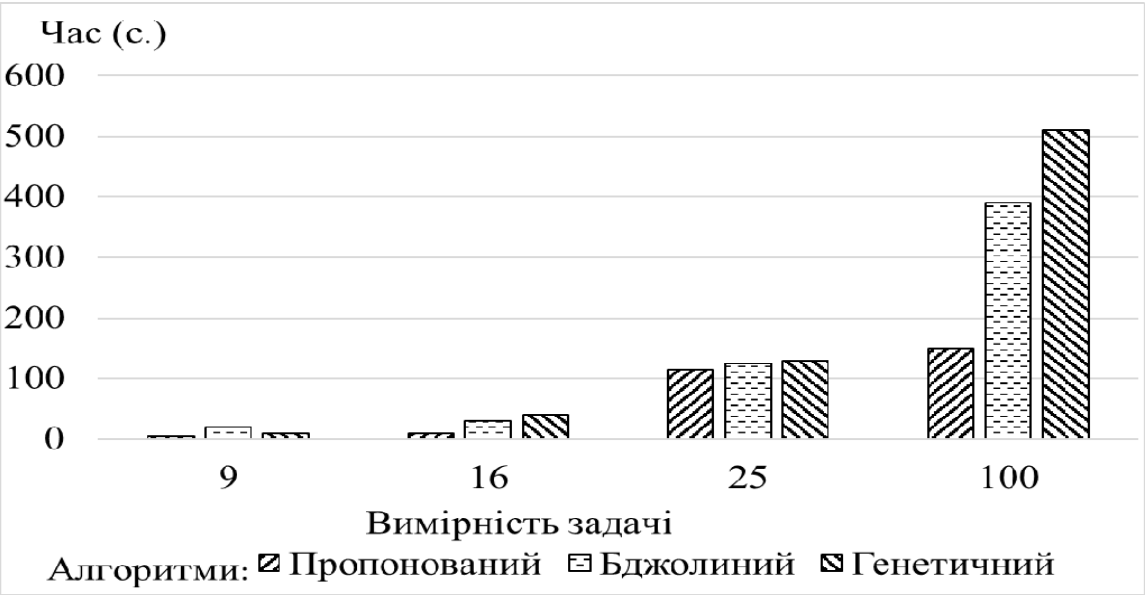


Рис.3.29. Діаграма порівняння часу виконання алгоритмів в залежності від розмірності задачі [Розроблено автором]

Час виконання методу повного перебору відображений на рис. 3.30 логарифмічно, оскільки значення мають велику різницю в порядках (від 10^2 до 10^{29}).

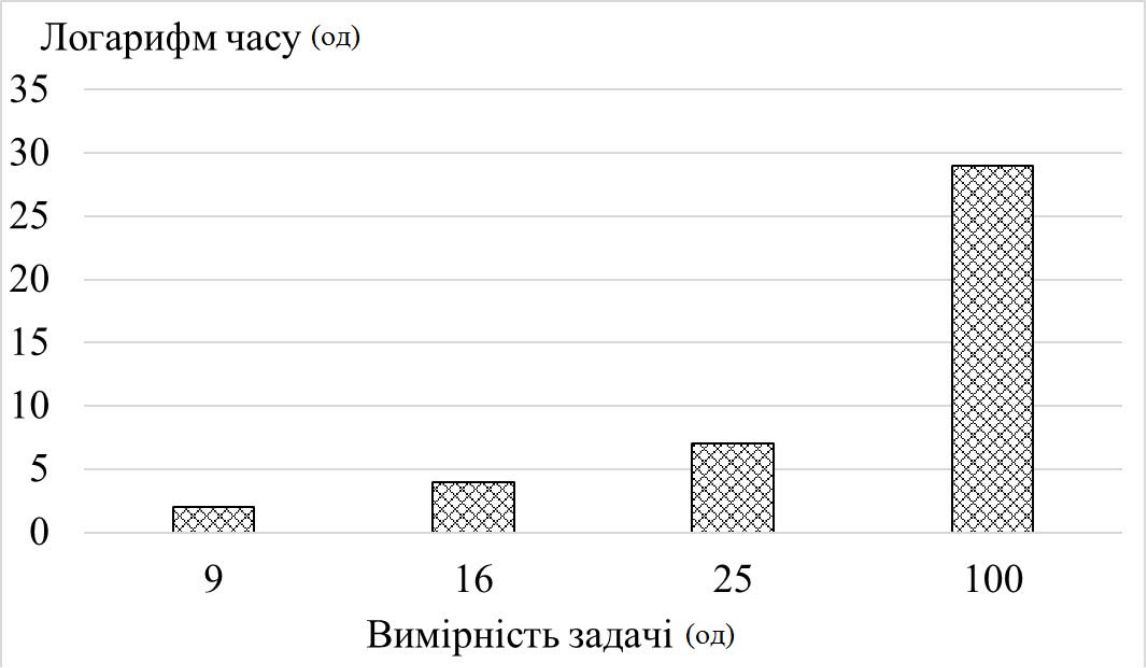


Рис.3.30. Діаграма логарифму часу виконання алгоритму повним перебором в залежності від розмірності задачі. [Розроблено автором]

На основі отриманих результатів можна зробити висновок, що пропонований алгоритм дозволяє зменшити час виконання алгоритму на 10% порівняно з часом виконання генетичного та до 50% - бджолиного, і на порядки – порівняно із часом методу повного перебору. Такий результат отримано за рахунок ближчої до оптимуму початкової популяції.

Висновки до розділу 3

У даному розділі описано метод побудови РСД ГТ на основі міжрівневих переходів за критерієм вартості побудови сховища з використанням модифікованого генетичного алгоритму, що дозволило отримати сховище мінімальної вартості за час, на 10 % менший, порівняно з часом використання генетичним алгоритмом та до 50 % меншим у порівнянні з часом використання методики бджолиного рою.

Для формування логічної моделі використовується метод спрощення формул алгебричної системи РСД ГТ за рахунок визначення перспективних множин інваріантів з наступним деталізованим вибором кращого у цих множинах, який відрізняється від існуючих наявністю правил оцінювання

інваріантів формули запиту на основі критерію вартості, що дозволяє отримати інваріант виразу запиту до сховища з мінімальною сукупною вартістю збереження та обробки даних.

Для формування фізичної моделі описується розміщення даних областей сховища та вибір маршруту реплікації цих даних. Ці процедури здійснюються на основі критеріїв вартості та модифікованого генетичного алгоритму з формуванням початкової популяції методикою бджолиного рою.

В результаті проведених досліджень для визначення параметрів генетичного алгоритму встановлено, що найкращий результат цільової функції при обмеженнях на час виконання алгоритму досягається при 10 початкових хромосомах, п’яти областях і з однією хромосомою в кожній області. Задані параметри, а саме: ймовірність схрещування двох особин $p_i = 0.8$ та ймовірність мутації $p_m = (n_N - 1)/(n_A \cdot n_N \cdot 2)$ дозволяють отримати достатньо якісне значення цільової функції за прийнятний час. Для задач більшої розмірності потрібен більший розмір популяції, тому потрібно вибирати $n_I = n_N * 10$ початкових хромосом.

Результати даного розділу описані в роботах [102..106].

РОЗДІЛ 4. ПРАКТИЧНА РЕАЛІЗАЦІЯ ІТ ПОБУДОВИ РСД ГТ

У даному розділі розглянуто архітектуру ІТ побудови РСД ГТ і особливості функціонування та реалізації даної ІТ, визначено структуру систем та описано існуючі програмні рішення, побудовані з використанням зазначеної ІТ. Описано практичні впровадження отриманих наукових результатів при побудові сховищ даних ІС для Міністерства фінансів України на основі ТСУДФ.

4.1. Архітектура ІТ побудови РСД ГТ

Одним з основних факторів, який необхідно враховувати при використанні ІТ побудови РСД ГТ, є можливість ефективного виконання різних класів запитів на даних різного рівня структурованості.

Для побудови РСД ГТ розроблена ІТ забезпечує:

- аналіз джерел даних, в тому числі структурованості даних;
- розміщення даних в носіях сховища за їх структурованістю;
- виконання запитів користувачів до носіїв сховища з виконанням необхідних додаткових операцій;
- оптимізацію сховища даних за зібраною статистикою виконання запитів.

Для взаємодії користувачів з інформаційними системами, побудованими на основі даної ІТ, її впровадження забезпечує:

- зручність отримання даних для кінцевого користувача;
- надання інформації за запитами користувачів незалежно від носія, де розміщені дані, та функцій, що він підтримує;
- надання даних в різних поданнях – табличне, “Pivot”, ієрархічне, текстове та ін.;
- розширюваність даних, метаданих та функцій сховища;
- можливість розподіленої роботи користувачів з розмежуванням доступу до даних.

Виходячи з зазначених вище вимог, розроблена ІТ побудована за змішаною трьохрівневою Web-орієнтованою архітектурою з серверами баз

даних, застосувань і клієнтами (браузери, товсті клієнти), що відображена на рис. 4.1.

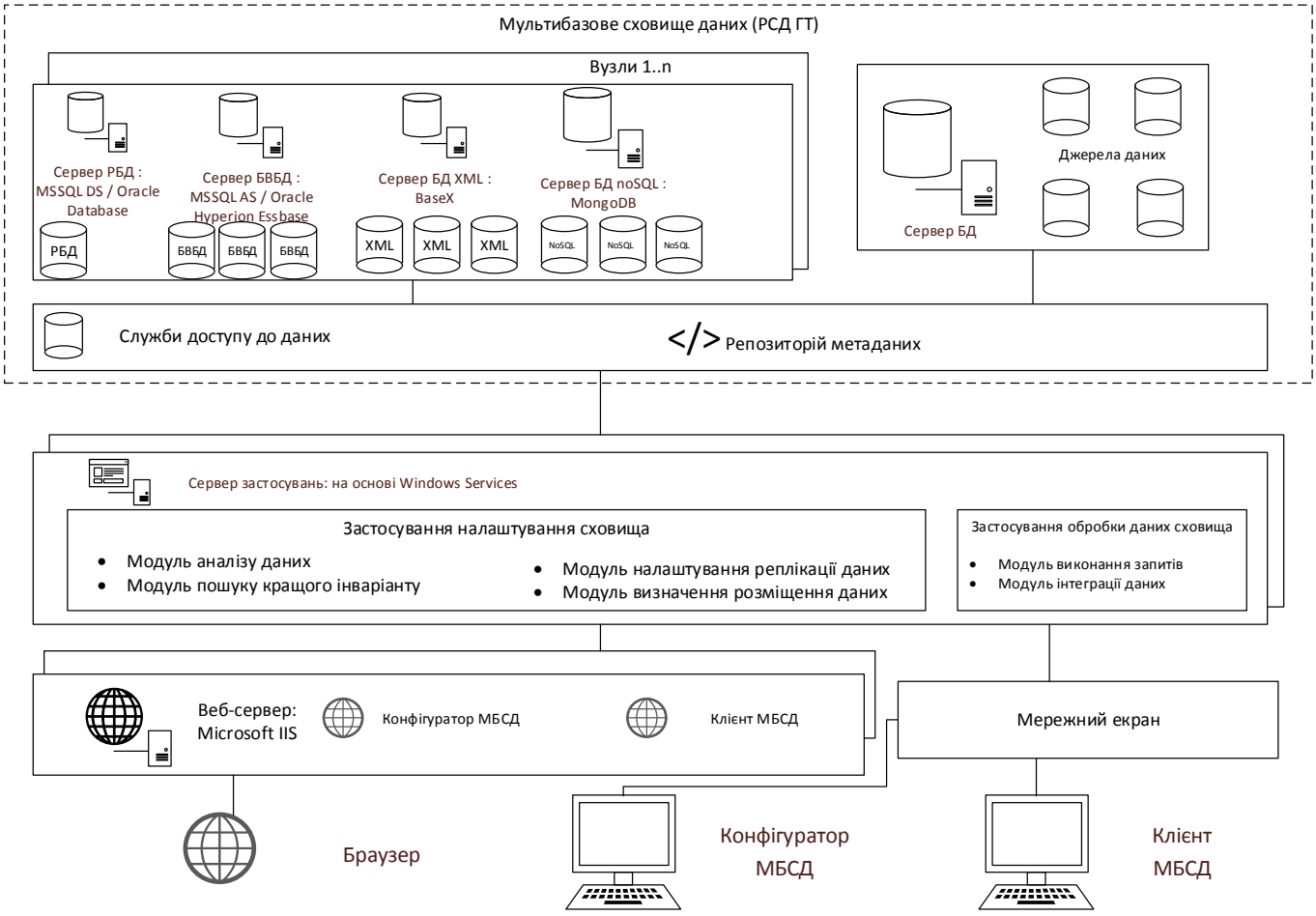


Рис. 4.1. Архітектура ІТ побудови РСД ГТ. [Розроблено автором]

Особливістю цієї архітектури є можливість користуватися функціоналом застосувань ІТ як у публічних мережах за допомогою браузера та протоколів (HTTP/S), так і в захищених мережах за допомогою товстих клієнтів та їх протоколів. В силу необхідності доступу клієнтів з різних розташувань сервери БД, застосувань та веб-сервери також розгортаються розподілено.

Відповідно до цієї архітектури, дані зберігаються в РСД ГТ (використовується МБСД), що складається з баз даних (реляційних, багатовимірних, XML, NoSQL з відповідними їм системи управління) у складі вузлів і джерел даних, що розміщуються у системах збереження під управлінням відповідних серверів БД, та служб доступу до даних для забезпечення зовнішньої взаємодії з даними сховища, які зберігають і використовують відповідний репозиторій метаданих.

Використовуючи дані цих служб, за допомогою модулів на серверах застосувань здійснюється побудова РСД ГТ та планування і збірка даних для виконання запитів.

Для побудови РСД ГТ використовується застосування налаштування сховища даних з такими модулями:

- аналізу даних, який здійснює аналіз структурованості даних і визначення областей сховища;
- пошуку кращого інваріанту, який визначає моделі для представлення даних;
- визначення розміщення даних, який розташовує дані областей сховища по його вузлах;
- налаштування реплікації даних, що конфігурує реплікацію областей сховища між його вузлами.

Для експлуатації РСД ГТ використовується застосування обробки даних сховища, що використовує модулі:

- виконання запитів отримує запити від користувачів і повертає набори даних відповідно цим запитам;
- інтеграції даних дозволяє здійснювати імпорт та експорт даних з чи в зовнішні системи.

Доступ користувачів ІТ до ресурсів програмних модулів здійснюється через веб-застосування «Конфігуратор МБСД» та «Клієнт МБСД», що розташовані на веб-серверах.

Користувач взаємодіє з цими застосуваннями через тонкий клієнт (браузер або клієнт графічного інтерфейсу), що встановлюється на комп’ютерах кінцевих користувачів.

4.2. Комплекс інструментальних засобів ІТ побудови РСД ГТ

Для реалізації інформаційної технології побудови РСД ГТ необхідно визначити засоби, які забезпечують:

- збереження даних (системи керування базами даних);
- обробку даних (сервери застосувань);

- представлення даних (веб-сервери).

Збереження даних. Розглянемо існуючі на ринку програмні рішення, а саме системи керування базами даних, що втілюють архітектури сховищ даних. З присутніх на ринку можна виділити наступні:

1. Microsoft SQL Server Analysis Services. Надає як багатовимірне (режим Multidimensional Mode) подання даних, так і реляційне (Tablular Mode).

2. У режимі Multidimensional Mode як джерела підтримуються лише бази даних. Multidimensional Mode частково втілює архітектуру віртуальної бази даних.

3. У режимі Tabular Mode як джерела підтримуються бази даних та текстові файли,Excel, PowerPivot, куби AS,RSS. Tabular Mode частково втілює архітектуру віртуальної бази даних.

4. Функція Oracle OLAP. Надає багатовимірне подання даних. Працює виключно як багатовимірна база даних, джерела не підтримуються, у якості носія виступає Oracle Database.

5. Oracle Essbase. Надає багатовимірне подання даних, будується за класичною архітектурою сховищ даних. Підтримуються джерела : бази даних , текстові файли, файли експорту, файли Excel, система Oracle BI Enterprise Edition (OBIEE). Дані зберігаються у структурах Block storage / Aggregate storage (куби даних).

6. IBM Cognos. Надає реляційне подання даних, будується за архітектурою віртуальної бази даних. Підтримуються наступні джерела : DB2 (OLAP), куби Congos, Hyperion Essbase, IBM Infosphere Warehouse, Informix, SQL Server, Analysis Services, джерела ODBC, БД Oracle, SAP BW, IBM Cognos TM1, XML.

7. IBM Cognos TM1. Надає багатовимірне подання даних, будується за класичною архітектурою сховищ даних.

8. Microstrategy Intelligence Server. Надає реляційне подання даних, будується за архітектурою віртуальної бази даних. Підтримуються джерела: реляційні бази даних (Microsoft Access, Microsoft SQL Server, MySQL, Oracle,

PostgreSQL, Sybase, Teradata), багатовимірні бази даних (Hadoop Hive,IBM Cognos TM1,SQL Server Analysys Services,Hyperion Essbase), текстові файли та інші.

9. Microstrategy Intelligence Server. Може будуватися як за класичною архітектурою сховищ даних, так і за архітектурою ВБД. Дані зберігаються в модулях Datastore (джерела даних) або InfoCube (реляційних баз даних за схемою типу «зірка»)

Проведений аналіз свідчить, що автоматичного розподілу в розглянутих рішеннях немає.

Серед цих рішень Oracle OLAP є суто багатовимірною базою даних. Інші рішення будуються за класичною архітектурою сховищ даних (Oracle Essbase, IBM Cognos TM1), за архітектурою ВБД (IBM Cognos , Microstrategy Intelligence Server) або підтримують обидві архітектури (Microsoft SQL Server Analysis Services, Microstrategy Intelligence Server)

Запропоновані підходи до побудови сховищ даних (див. розділ 1), на яких базуються існуючі рішення, передбачають врахування джерела даних в цілому на рівні схеми (баз даних), а також управління змінами. Однак вони не враховують структури джерел даних, і не описують побудову сховищ даних на базі цих даних, що відображається у практичній реалізації цих підходів.

Також на основі аналізу розглянутих підходів до побудови розподіленого зберігання даних, зокрема «великих даних» BigData, можна зробити висновок, що системи розподіленого зберігання даних не передбачають використання різних моделей даних для оптимізації.

Обробка даних. Для розробки СК МБСД було використано платформу Microsoft.Net, мову C# та середовище розробки Visual Studio. Бібліотека класів, що поставляється з .net і мова високого рівня C#, а також методологія RAD (швидкої розробки застосувань), на якій побудоване середовище розробки Visual Studio, дозволяє швидко створювати застосування, орієнтовані на бази даних. Вагомою перевагою даної платформи в розробці засобів роботи з

системами управління даних є доступність і розширюваність програмних засобів взаємодії з цими системи.

Для реалізації ІТ була вибрана 64-розрядна платформа (x64). Вона дозволяє використовувати 64-розрядні процесори, а також обсяги основного запам'ятовуючого пристрою, що перевищують 4 ГБ.

Представлення даних. Оскільки використовується платформа Microsoft .NET Framework, тому в якості веб-сервера використовується Microsoft Internet Information Services, а сервер застосувань виконаний в вигляді набору служб Windows (Windows Services). Кожне веб-застосування і застосування виконується у окремому пулі або службі відповідно, працездатність котрих підтримується вбудованими системними засобами – диспетчерами IIS та служб Windows. Кожна служба підключає і використовує відповідні модулі у відповідності до описаних методів та звернень користувача, що надходять від веб-застосувань або товстих клієнтів.

Таким чином, практична реалізація ІТ складається з наступних програмних компонент:

- СК реляційних баз даних (Microsoft SQL Server Database Services, Oracle Database, MySQL,PostgreSQL) у 64-розрядному виконанні;
- СК багатовимірних баз база даних (Microsoft SQL Server Analysis Services або Hyperion Essbase) у 64-розрядному виконанні;
- СКБД XML (BaseX);
- СКБД NoSQL (MongoDB)
- система керування мультибазовим сховищем даних (СК МБСД), яка є окремою програмою, розробленою спеціально для забезпечення функціонування сховища даних, і включає в себе файлове сховище.

4.3. Опис реалізації інформаційної технології

Для опису функціонального призначення програмних засобів ІТ побудови РСД ГТ в UML-нотації використана поведінкова діаграма прецедентів (варіантів використання, use case diagrams). Тому для графічного представлення

варіантів використання ПЗ ІТ побудови РСД ГТ наведемо відповідну UML діаграму (рис 4.2.).

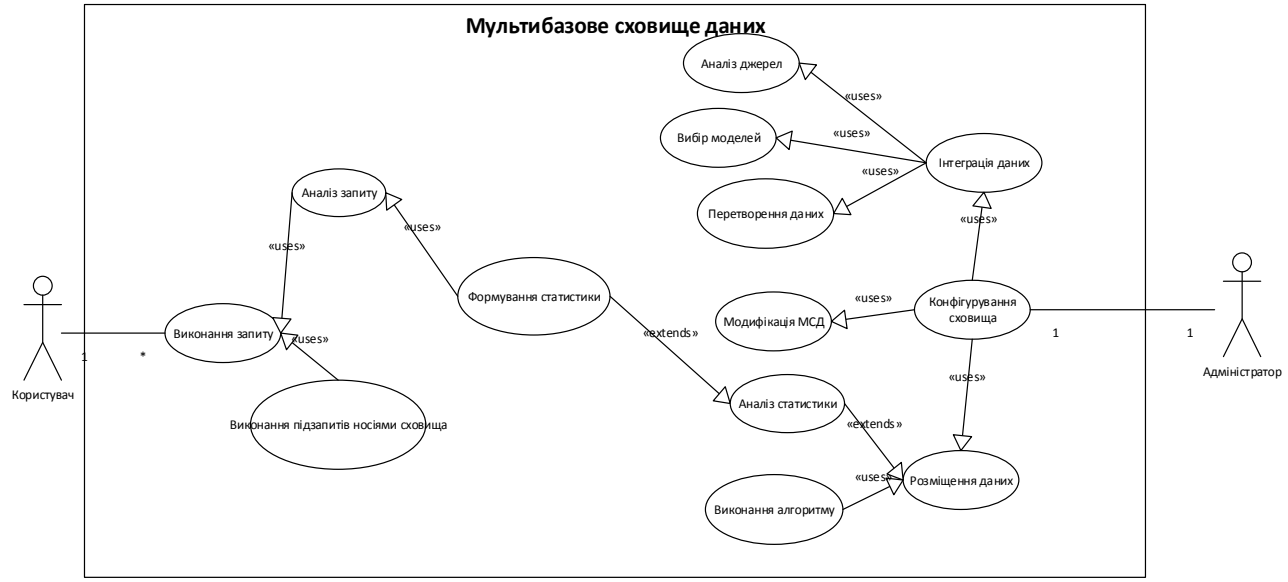


Рис.4.2. Діаграма варіантів використання ПЗ ІТ побудови РСД ГТ. [Розроблено автором]

Діаграма варіантів використання описує взаємодію між варіантами використання ПЗ та дійовими особами, та відображає функціональні вимоги до ПЗ ІТ побудови РСД ГТ з точки зору кінцевого користувача.

СК МБСД складається з наступних застосувань:

- MDWManager, графічна утиліта для проектування сховища;
- MDWManager.Web, веб-застосування для проектування сховища;
- MDWClient, «товстий» клієнт, за допомогою якого користувачі отримують результати запитів;
- MDWClient.Web, веб-застосування для опрацювання даних сховища;
- MDWService.Build, служба, яка виконує побудову сховища та аналіз даних;
- MDWService.Query, служба, яка забезпечує виконання запитів взаємодію з джерелами та носіями даних;
- MDWTestApp, застосунок для проведення досліджень.

Взаємозв'язок цих компонентів СК МБСД показаний на діаграмі компонентів СК МБСД у додатку Г.

Стани сховища і їх послідовність зміни наведена на діаграмі діяльності СК МБСД у додатку Е.

Дане програмне забезпечення (застосування і їх моделі) розроблено на платформі Microsoft .NET 4.0. Діаграма класів СК МБСД наведена в додатку Д.

Класи модуля MDWService описані в додатку Б.

Програмний код MDWService для визначення областей наведено в додатку А. Копії екранів роботи модуля MDWManager наведено в додатку В.

Для роботи модулів СК МБСД встановлені наступні мінімальні вимоги до технічних засобів (без врахування відповідних вимог до СКБД):

- сервер архітектури x64 (64-розрядна платформа) – процесор 2 ГГц (двохядерний), 8 ГБ ОЗП, 100 ГБ НЖМД, ОС MicrosoftWindowsServer 2008 R2 (x64);
- робоча станція архітектури x86 або x64 (32-розрядна або 64-розрядна платформа) - процесор 1 ГГц (одноядерний), 1 ГБ ОЗП, 10 ГБ НЖМД, ОС Microsoft Windows 7 або Microsoft Windows 7 x64.

4.4. Особливості використання ІТ побудови РСД ГТ при створенні інформаційних систем для Міністерства фінансів України

4.4.1. Задача побудови розподіленої інформаційної системи для управління бюджетними коштами

Інформаційна технологія побудови РСД ГТ використовувалася при створенні ряду систем для Міністерства фінансів України, зокрема для побудови розподіленої інформаційної системи управління бюджетними коштами.

Складання та контроль за виконанням державного та місцевих бюджетів України здійснюється на наступних рівнях [107]:

– рівень центру (МФ України, ДК України, ДФІ України);
– рівень регіону (області), управління Державного казначейства (УДК);
– рівень району, відділення Державного казначейства (ВКД).

Центральний рівень забезпечує збір, обробку, зберігання даних, що застосовуються на всіх рівнях бюджетного процесу. На центральному рівні зосереджено серверне обладнання, центральне комутаційне обладнання, пристрої зберігання даних.

На регіональному та районному рівні передбачається збір, зберігання та обробка даних лише стосовно даного регіону чи району.

Крім того, на регіональному рівні забезпечується функціонування транспортної мережі, агрегування каналів районного рівня та забезпечення інтерфейсу підключення фінансових установ регіонального рівня до відомчої транспортної мережі. На районному рівні - забезпечення точки доступу бюджетних установ місцевого рівня до відомчої мережі та маршрутизація підключень користувачів до веб – порталу системи управління державними фінансами (СУДФ).

До децентралізації всі дані в ручному режимі завантажувалися в центральне сховище даних на реляційній БД. Після децентралізації фінансів виникла необхідність розподіленого збереження даних - дані повинні завантажуватися як у центральне сховище даних, так і на відповідний рівень. Таким чином, виникає задача ефективної обробки оперативних, первинних та аналітичних даних, з врахуванням взаємозв’язків між цими даними, як на центральному, так і на місцевому рівнях.

Види даних, що зберігаються і обробляються в управлінні державними фінансами [107]:

- оперативні дані включають всі показники, що впливають на бюджетний процес, використовуються для прийняття рішень, одержуються в режимі реального часу і не потребують перевірок цілісності даних – наприклад, інформація про банківські транзакції, курси валют чи біржеві показники; ці дані можуть мати різну структуру, зберігаються розподілено (по всій Україні), обсяги даних ~ сотні гігабайт на рік;

- первинні дані є результатом прийняття рішення на основі оперативних даних і мають стандартну жорстку (чітко визначену) структуру, зберігаються розподілено (по всій Україні), обсяги даних ~ гігабайти на рік;

- дані для аналізу отримуються з первинних даних шляхом представлення в деякому вигляді, зокрема агрегації, обсяги даних ~ гігабайти на рік;

- метадані – інформація про числові показники, яка використовується у бюджетному процесі і має деревовидну структуру (чітко задану або з додатковими атрибутами, властивими окремим елементам даних), обсяги даних ~ десятки МБ на рік;

- супровідні дані (документи, навчальні матеріали, наукові видання тощо) в неструктурованому вигляді, у окремих застосуваннях. В окремих випадках проходять процедуру аналізу і попадають в одну з попередніх категорій даних, обсяги даних ~ до гігабайт на рік.

При таких різноманітних властивостях та видах даних для їх зберігання та ефективної обробки, як правило, на логічному рівні використовуються різні бази даних, зокрема реляційні, багатовимірні, бази даних NOSQL та NoSQL, а також файлове сховище.

На фізичному ж рівні для збереження даних необхідно для кожного з вузлів визначити, які бази даних на ньому будуть виконуватися для максимальної загальної ефективності системи, тобто для забезпечення необхідного рівня швидкодії і не перевищення відведених на це лімітів сумарних затрат на систему.

Для реалізації інформаційних систем, що вирішують поставлену вище практичну задачу, пропонується використати МБСД на основі ІТ побудови РСД ГТ, оскільки вона дозволяє розподілено зберігати та ефективно обробляти різні типи даних.

На базі ІТ побудови РСД ГТ була розроблена так звана типова система управління державними фінансами (ТСУДФ), що складається з наступних рівнів.

Рівень зберігання даних ТСУДФ (рис. 4.3.) розміщується на серверах баз даних і містить МБСД (системи керування реляційними, багатовимірними базами даних, БД NOSQL, БД noSQL, файловим сховищем), яке через служби доступу отримує запити і у відповідності до них надає дані чи змінює дані сховища, а також оптимізує свою роботу на основі методу побудови РСД ГТ.

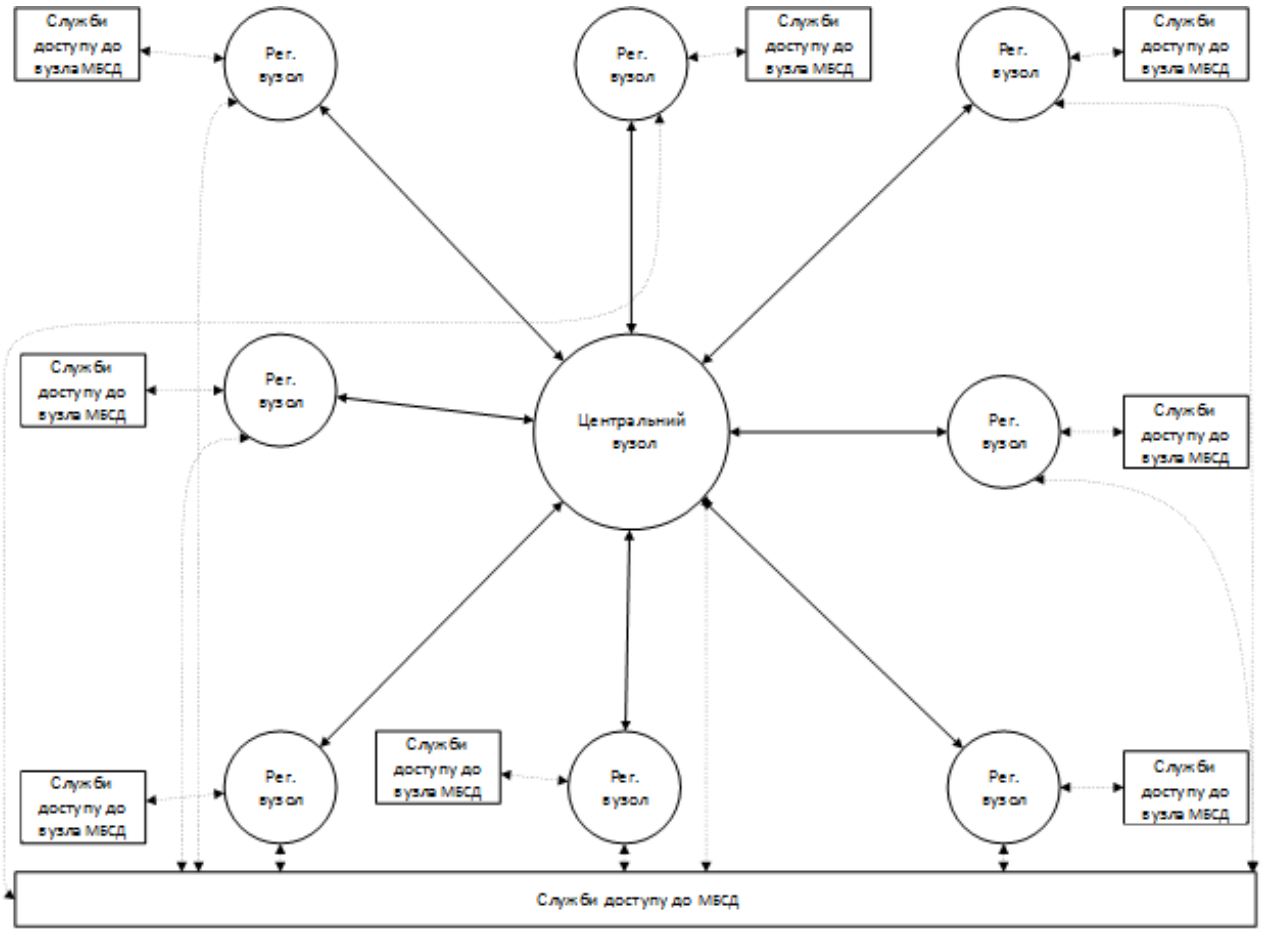


Рис. 4.3. Структура рівнів зберігання даних ТСУДФ. [Розроблено автором]

2. Рівень застосувань ТСУДФ (рис 4.4.) містить модулі, які виконуються на серверах застосувань і поділяються на:

2.1. Модулі надання даних, які призначені для взаємодії з даними у сховищі, зокрема модулі експорту та імпорту даних, аналізу даних та формування звітності. Ці модулі розташовуються на серверах, що мають низький час доступу по мережі (network latency) до серверів, на яких розгорнуті служби доступу МБСД.

2.2. Модулі прикладних застосунків, які обробляють дані у відповідності до бізнес-процесів управління державними фінансами і розташовуються на серверах, розміщених як центрально, у випадку використання служб віддалено, так і локально, у випадку коли застосування оперує даними певного регіону.

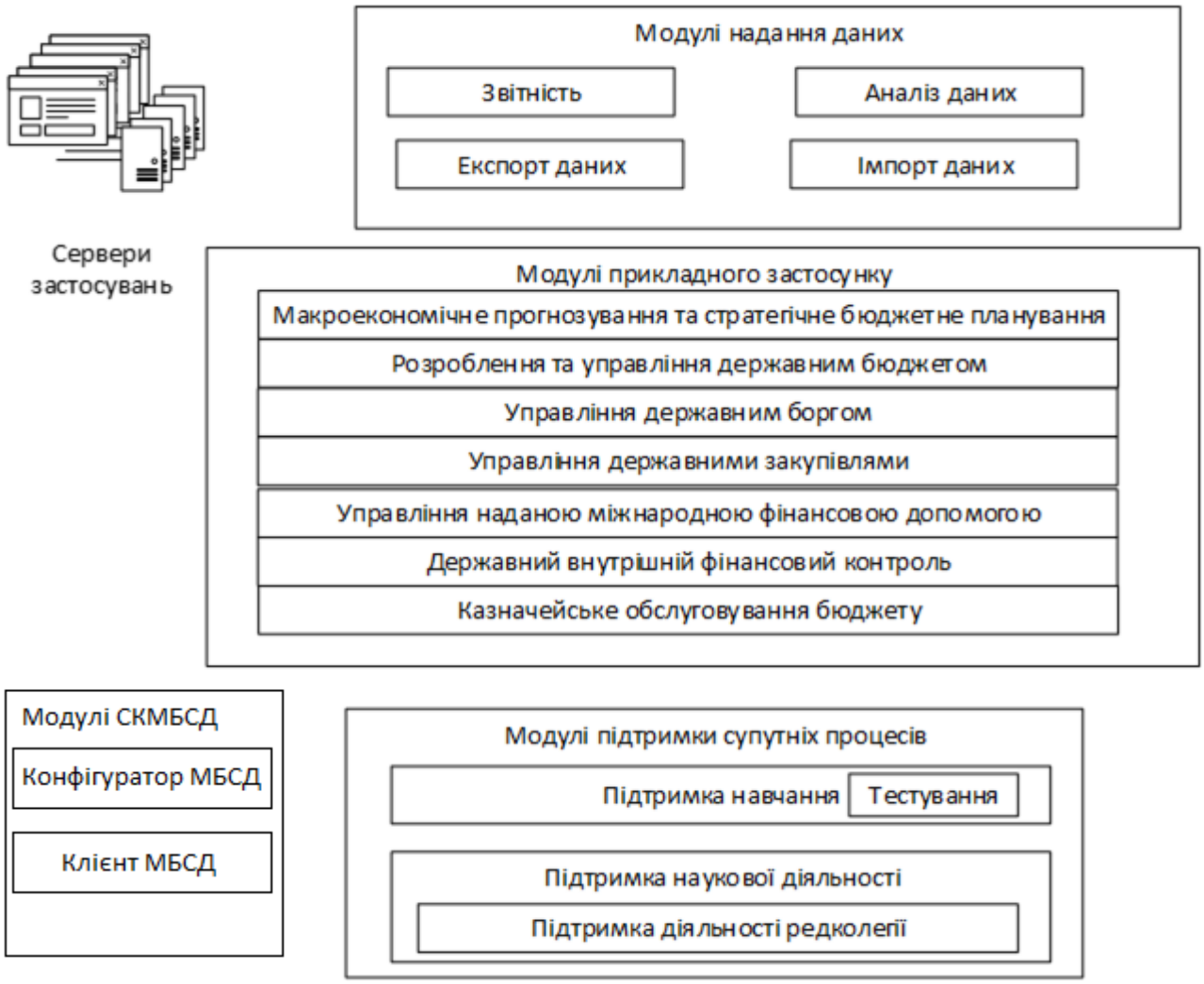


Рис. 4.4. Структура рівня застосувань ТСУДФ. [Розроблено автором]

2.3. Модулі підтримки супутніх процесів, які мають низький рівень критичності і можуть розташовуватися як на центральних, так і на локальних серверах.

2.4. Модулі СК МБСД, які виконують обслуговування сховища (див. архітектуру ІТ, п.4.1)

3. Інтерфейсний рівень ТСУДФ, що включає веб-сервери, сервери CMS, файлові сервери, сервери захищеного доступу (рис. 4.5.). Ці модулі дозволяють клієнтам (іншим застосуванням та кінцевим користувачам) працювати з застосуваннями управління фінансами. Модулі-служби розміщуються як на центральних, так і на локальних серверах, в той час як застосунки-клієнти є мобільними відносно цих служб і можуть використовуватися лише при наявності мережного підключення до служб доступу (мережі Інтернет).

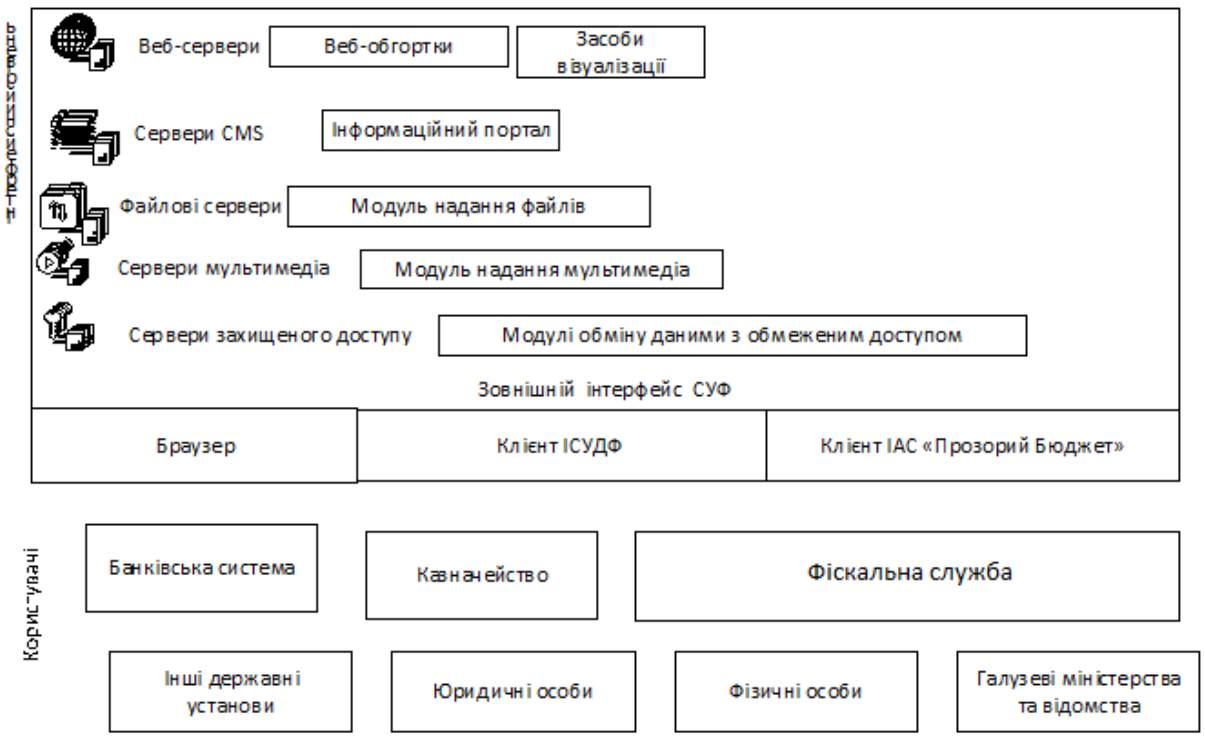


Рис. 4.5. Структура інтерфейсного рівня доступу до даних ТСУДФ. [Розроблено автором]

На основі МБСД, модулів рівня застосувань та модулів інтерфейсного рівня будується прикладне програмне забезпечення.

4.4.2. Інтегрована система управління державними фінансами

На основі архітектури ТСУФ будується інтегрована система управління державними фінансами (ІСУДФ), яка призначена для забезпечення ефективного, результативного і цільового використання бюджетних коштів учасниками бюджетного процесу відповідно до їх повноважень, що спрямовані на досягнення цілей, завдань і конкретних результатів своєї діяльності [107].

ІСУДФ не є загальнодоступною системою, тому на інтерфейсному рівні використовується модуль обміну даними з обмеженим доступом, а для доступу до самих застосувань використовуються відповідні веб-обгортки, а клієнт ІСУДФ призначений строго для внутрішнього використання.

Функціональність ІСУДФ підтримує наступні центральні процеси зі сторони (за рахунок відповідних модулів):

1. Міністерства фінансів України:

- макроекономічне прогнозування та стратегічне бюджетне планування;
- розроблення та управління державним бюджетом;
- управління державним боргом;
- управління державними закупівлями;
- управління міжнародною фінансовою допомогою;
- державний внутрішній фінансовий контроль;
- казначейське обслуговування бюджету.

2. Державного казначейства України:
- ведення єдиного реєстру бюджетних установ та організацій;
- реєстрація і доведення планових показників;
- формування пропозицій на відкриття асигнувань і розподіл відкритих асигнувань;
- управління ресурсами;
- головна книга бухобліку;
- бухоблік бюджетних установ;
- управління матеріальними активами та запасами;
- контроль і облік зобов'язаннями/ кредиторська-дебіторська заборгованість.

Для сховища ІСУДФ формулюються наступні вимоги до його ефективності:

- час запису інформації типової екранної форми в базу даних не повинен перевищувати 5 секунд;
- час видачі інформації типового запиту із бази даних не повинен перевищувати 5 секунд;
- ППЗ повинно забезпечувати одночасну роботу із системою не менш, ніж 500 осіб;
- у моменти пікового завантаження ППЗ повинно забезпечувати обробку не менш, ніж 30000 транзакцій за годину.

На рис. 4.6. представлено інформаційні потоки функціональності ІСУДФ.

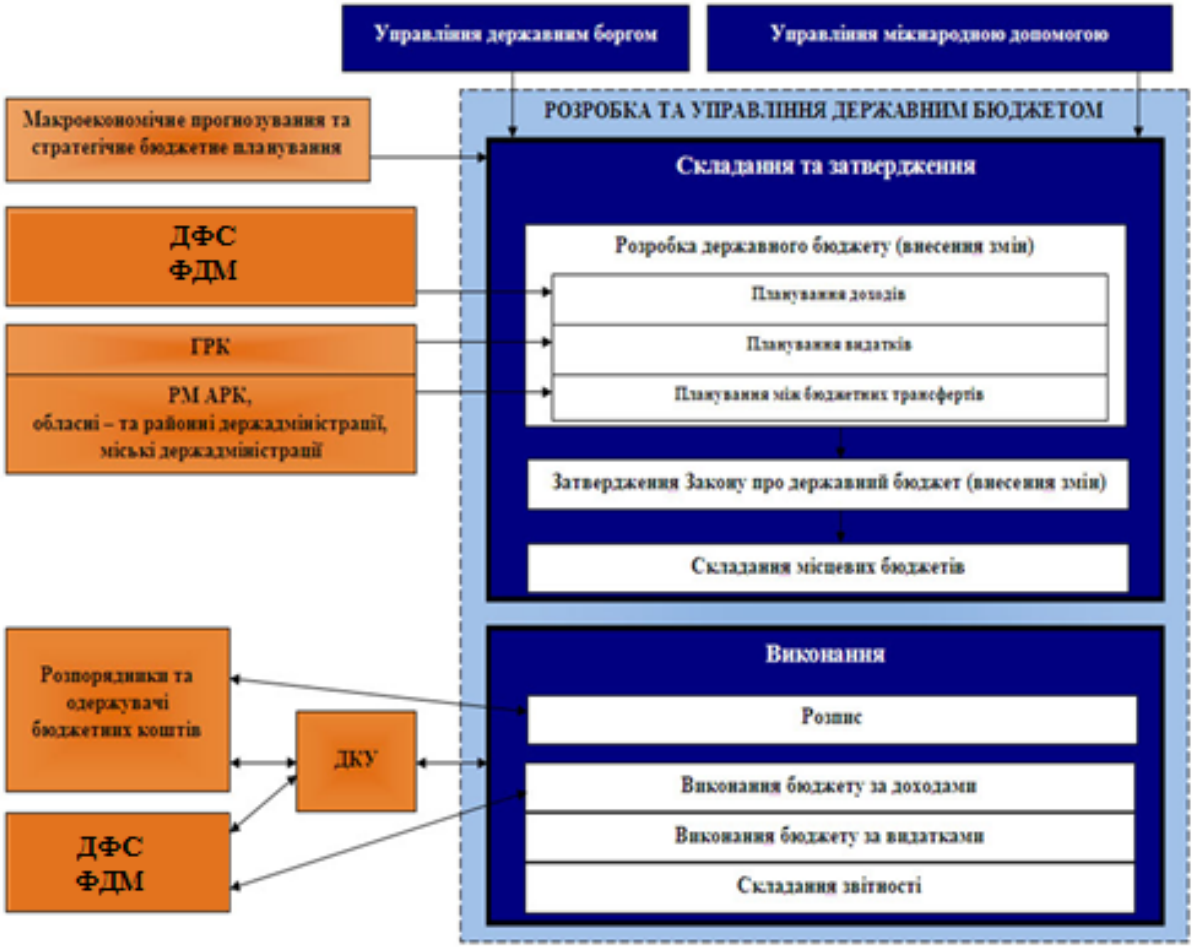


Рис. 4.6. Інформаційні потоки для управління бюджетними коштами. [Розроблено автором]

Основними фазами бюджетного процесу є :

- складання проекту бюджету;
- формування розпису бюджету;
- моніторинг виконання бюджету.

Оскільки, ці фази управління державними фінансами, як правило, використовують запити до даних різного типу, то для автоматизованої оптимізації запитів використано МБСД на 27 вузлах, де була інстальована БД Oracle Database та Oracle Essbase. У результаті застосування МБСД було зменшено сумарну вартість роботи ІСУДФ (рис.4.7.) за рахунок більш оптимального розподілу ресурсів, які використовуються, шляхом зменшення кількості фізичних вузлів.

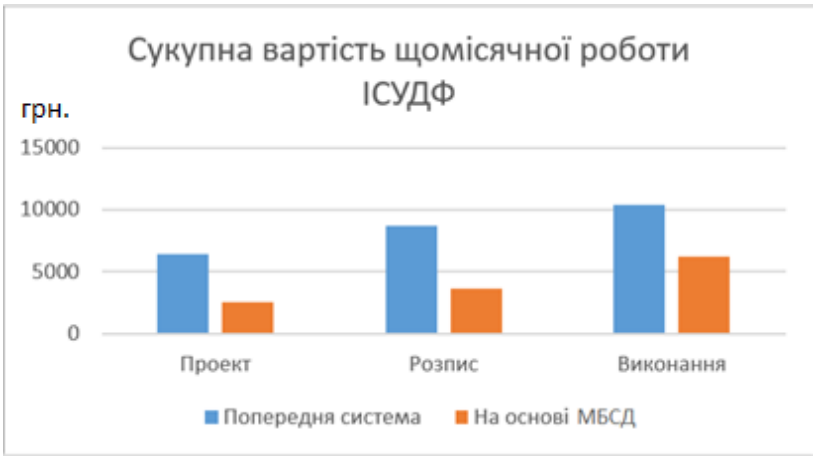


Рис. 4.7. Діаграма сукупної вартості витрат на підтримку роботи ІСУДФ на місяць (у грн.). [Розроблено автором]

4.4.3. Інформаційно-аналітична система «Прозорий бюджет»

Метою створення інформаційно-аналітичної системи «Прозорий бюджет» (ІАС ПБ) є забезпечення прозорості та публічності державних коштів за допомогою розміщення публічної інформації про результати діяльності органів фінансової сфери.

Основними завданнями ІАС ПБ є:

- підвищення поінформованості громадян та бізнесу про бюджетний процес і роботу державних органів влади;
- забезпечення можливості кожному члену суспільства бачити на які цілі витрачаються податки, що ними сплачуються;
- створення умов для прозорого діалогу влади із громадськістю у ході бюджетного процесу;
- забезпечення відкритості політики уряду для світового співтовариства, що підвищує інвестиційну привабливість країни та її регіонів і сприяє росту інвестицій в економіку України;
- підвищення прозорості розрахунків у сфері соціального забезпечення, формування, вирахування та розподілу соціальних виплат і пільг;
- надання можливості моніторингу з боку громадськості за цільовим витрачанням бюджетних коштів як інструменту боротьби з корупцією;
- скорочення часу обслуговування заявок і питань громадян;
- моделювання та прогнозування ситуацій у фінансовій сфері;

- перевірка ефективності прийняття рішень;
- автоматизація вибору оптимального рішення;
- надання можливостей гнучкої візуалізації.

Приклад функціонування ІАС ПБ представлений на рис. 4.8.

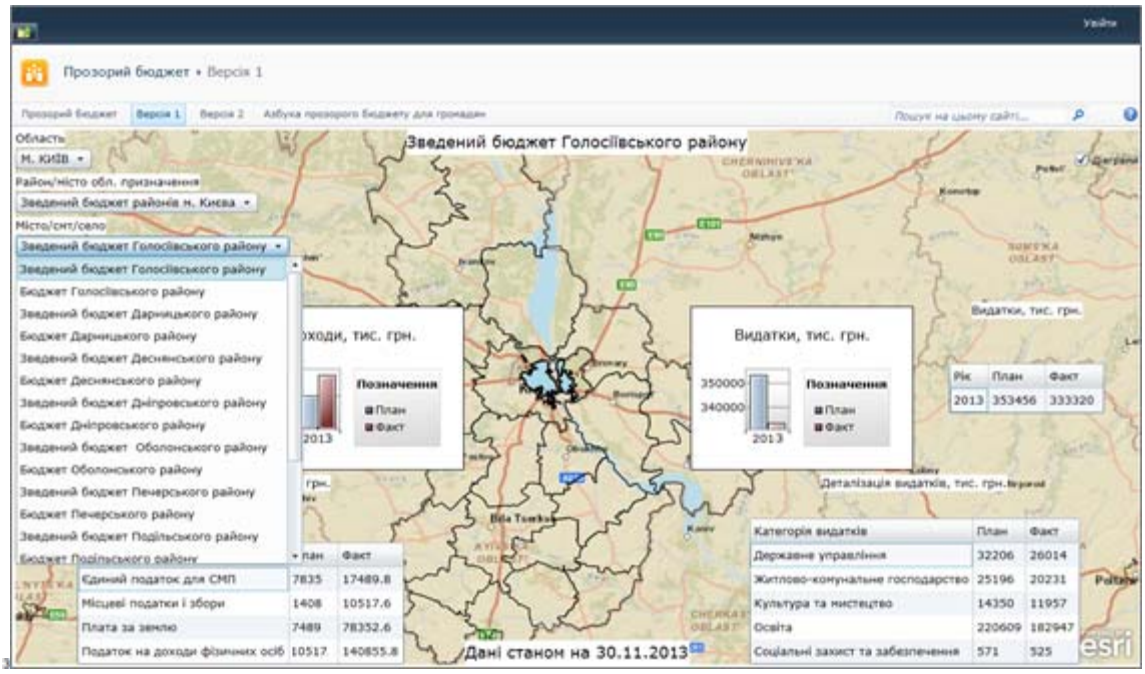


Рис. 4.8. Візуалізація даних ІАС «Прозорий бюджет». [Розроблено автором]

ІАС ПБ використовує модулі аналізу даних рівня застосувань та візуалізації даних інтерфейсного рівня, а для доступу – браузер і клієнтський браузерний модуль.

Для ІАС ПБ на рис. 4.9 представлена технічна архітектура, на рис. 4.10 – функціональна архітектура, а на рис. 4.11 – структурна схема бази даних.

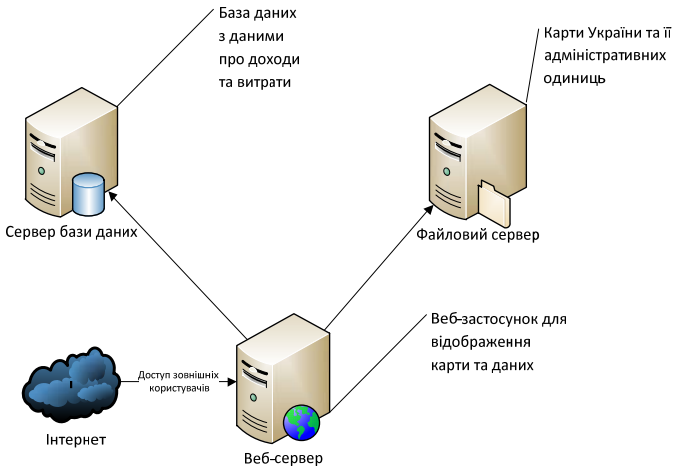


Рис. 4.9. Технічна архітектура ІАС «Прозорий бюджет». [Розроблено автором]

ІАС ПБ побудована за класичною трирівневою архітектурою з використанням додаткового файлового сервера для зберігання карт України та її адміністративних одиниць.

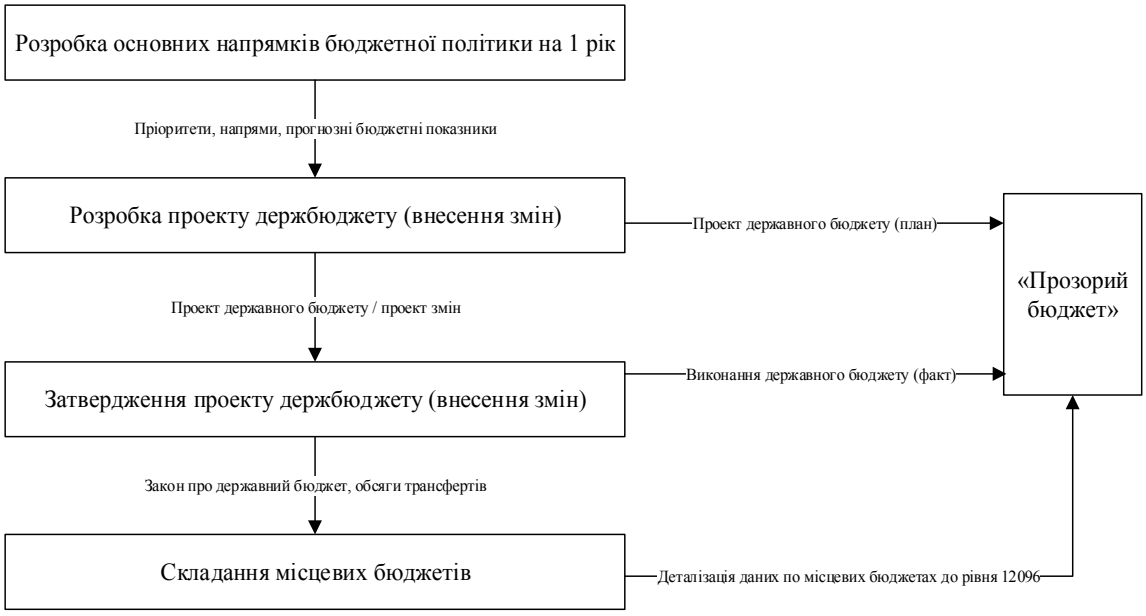


Рис.4.10 Функціональна архітектура ІАС «Прозорий бюджет»ю [Розроблено автором]

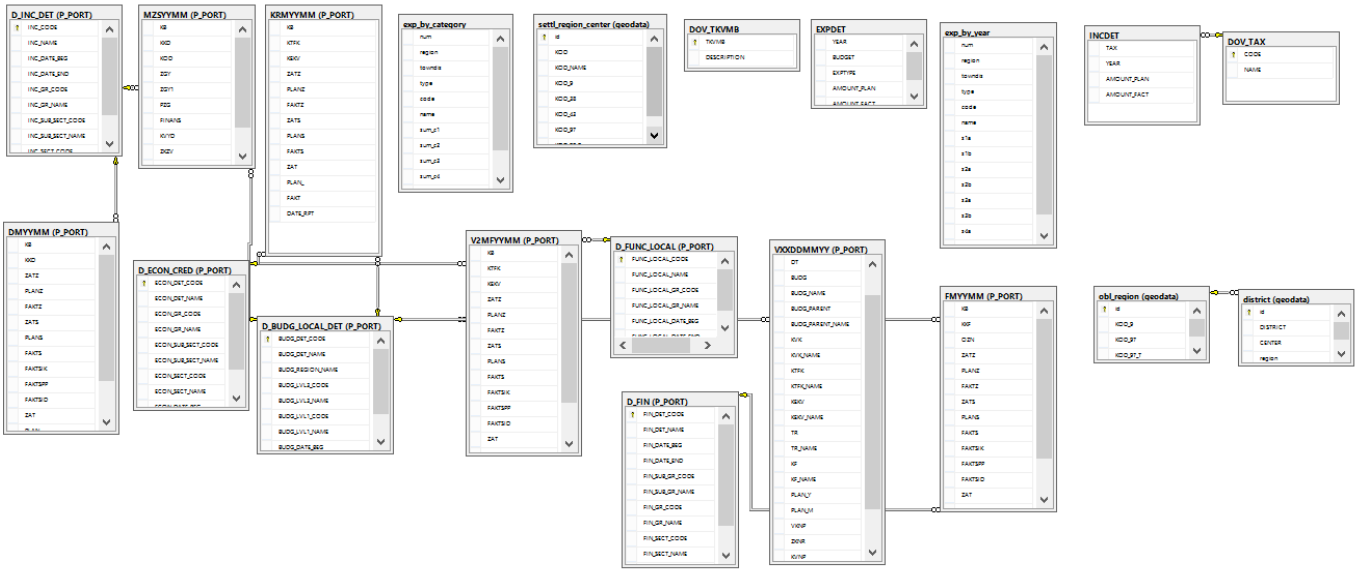


Рис. 4.11. Схема моделі бази даних для ІАС «Прозорий бюджет». [Розроблено автором]

Для забезпечення функціонування веб-застосування ІАС ПБ було побудоване сховище даних, що задовільняє вимогам Бюджетного кодексу України [107], містить довідники бюджетної класифікації – єдиного систематизованого згрупування доходів, видатків, кредитування, фінансування

бюджету, боргу відповідно до законодавства України та міжнародних стандартів, які визначають чисельні дані в розрізі

- 12 096 бюджетів усіх рівнів;
- план-факту державного та місцевих бюджетів;
- діапазону дат – 5 років (два минулі роки, поточний рік, два майбутні роки);
- категорії доходів місцевих бюджетів: єдиний податок, місцеві податки та збори, плата за землю, податок на доходи фізичних осіб;
- категорії видатків місцевих бюджетів: державне управління, житлово-комунальне господарство, освіта, охорона здоров'я, правоохоронна діяльність, соціальний захист.

Дані ІАС ПБ представлені 4-ма рівнями :

– загальноукраїнський (1-й);
– регіональний (2-й);
– районний (3-й);
– місцевий (4-й).

На діаграмі 4.12. наведено порівняння середнього часу виконання запиту отримання даних відповідних рівнів при застосуванні багатовимірних БД та МБСД.

Рис. 4.12. Діаграма середнього часу виконання запитів до ІАС «Прозорий бюджет» різних рівнів агрегування даних (у мс). [Розроблено автором]

Приріст швидкодії виконання запитів пояснюється оптимізацією сукупних запитів до різних БД, а саме: багатовимірної, реляційної і БД XML, в яких збережені фінансові дані, довідники і описові текстові дані.

4.4.4. Інформаційна система «Віртуальний університет»

Запропонована ІТ побудови РСД ГТ була використана при розробці інформаційної системи «Віртуальний університет» Міністерства фінансів України (ІС ВУ), метою якої є навчання без відриву від виробництва учасників бюджетного процесу щодо своїх повноважень, пов'язаних з формуванням та використанням бюджетних коштів, здійсненням контролю за дотриманням бюджетного законодавства, які спрямовані на досягнення цілей, завдань і конкретних результатів своєї діяльності та забезпечення ефективного, результативного і цільового використання бюджетних коштів на основі сучасних інформаційних технологій і кращого світового досвіду.

Основні функції ІС ВУ:

- навчання учасників бюджетного процесу відповідно до їх повноважень для забезпечення ефективного, результативного і цільового використання бюджетних коштів усіх рівнів;
- управління знаннями: системна консолідація досвіду учасників бюджетного процесу і його поширення;
- функції єдиного центру корпоративної культури, «сховища» цінностей учасників бюджетного процесу;
- функції центру інновацій у сфері управління державними фінансами.

Практична реалізація даних функцій забезпечується ІС ВУ за допомогою використання модулів застосувань: підтримки навчання, тестування, і модулів інтерфейсного рівня: інформаційного порталу, надання файлів, надання мультимедіа.

Модуль підтримки навчання – забезпечує формування навчальних груп, будує розклад занять, дозволяє переглядати матеріали навчальних курсів за групою та/або викладачем

Модуль тестування – забезпечує незалежне оцінювання та сертифікацію слухачів віртуального університету після опанування ними курсів навчальних програм на принципах прозорості, об'єктивності, рівності, справедливості, доступності й зрозумілості та реалізується з використанням платформи Microsoft SharePoint.

Модулі інформаційного порталу – дозволяє публікувати інформаційні матеріали у форматі статей та інтерактивних дошок та надає базову платформу для взаємодії з іншими модулями

Модуль надання файлів – забезпечує можливість завантаження матеріалів навчальних курсів.

Модуль надання мультимедіа - забезпечує можливість потокового перегляду матеріалів навчальних курсів.

Головна сторінка порталу Віртуального університету(ПВУ) Міністерства фінансів України представлена на рис. 4.13.

Рис. 4.13. Головна сторінка Порталу Віртуального університету МФУ. [Розроблено автором]

Сховище даних для Порталу та центру тестування є єдиним, схема БД якого представлена на рис. 4.14.

Варто зазначити, що навантаження на веб-застосування Порталу визначається статистикою кількості слухачів, які одночасно навчаються у ВУ,

(див. Додаток Ж) і накладає певні вимоги до СД, а саме підтримку декількох тисяч одночасних підключень до Порталу з збереженням ізоляції кожної сесії.

При застосуванні ІТ побудови РСД ГТ для ІА ВУ стало можливим оптимізувати навантаження виконання запитів отримання даних в режимі реального часу як для отримання навчальних матеріалів, так і для проходження слухачами тестів, що дозволило підвищити пропускну спроможність сховища даних, а відтак і кількість одночасних сесій для навчання та тестування з 200 до 250 (на сервер) без потреби виділення додаткових ресурсів.

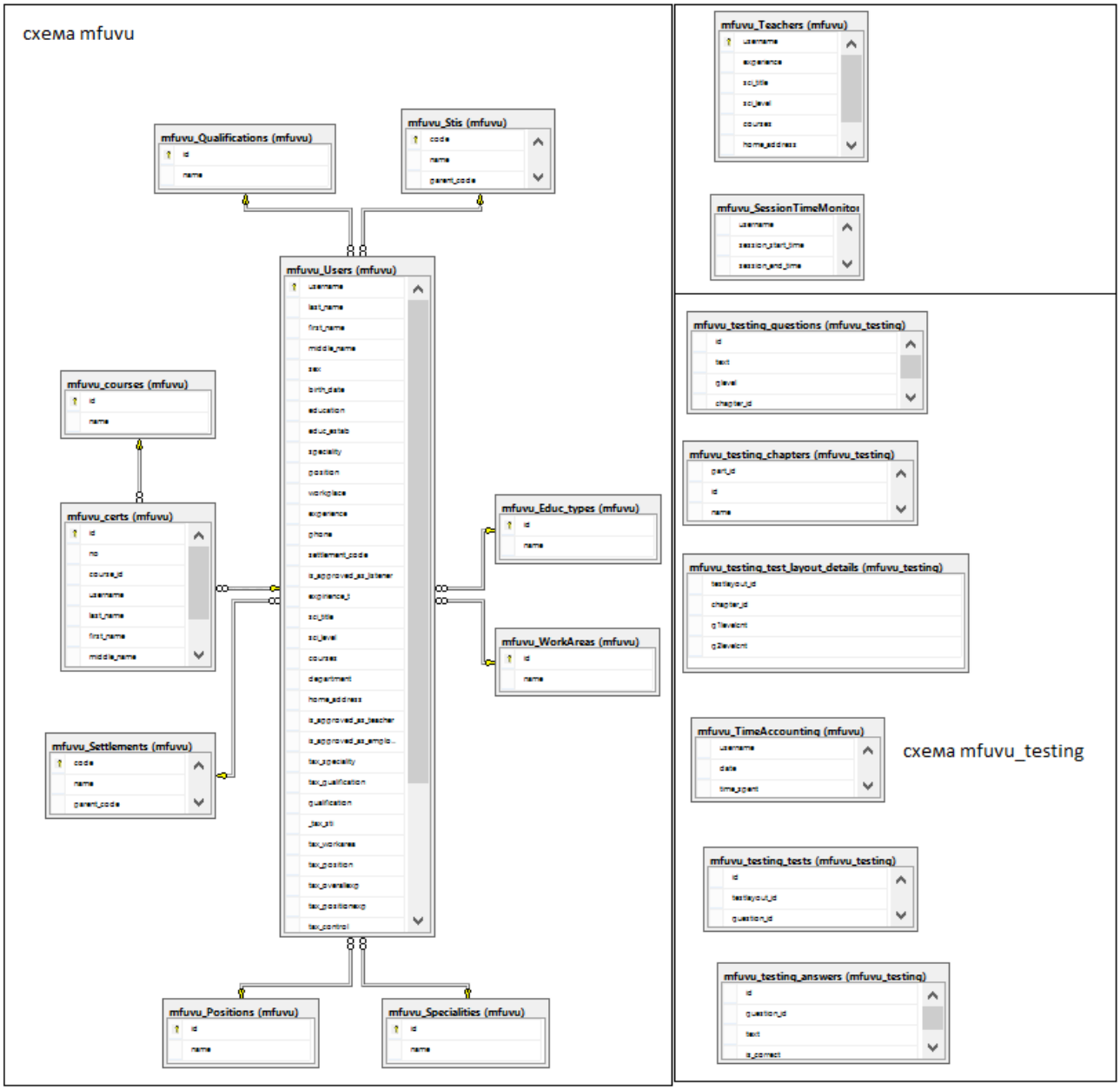


Рис. 4.14. Схема бази даних для «Віртуального Університету» та Центру оцінювання та сертифікації компетентності працівників МФУ. [Розроблено автором]

4.4.5. Портал Інституту післядипломної освіти «Академії фінансового управління»

Портал Інституту післядипломної освіти «Академії фінансового управління» (ПІПО АФУ) Міністерства фінансів України забезпечує створення єдиного інформаційного простору для підвищення компетентності учасників бюджетного процесу, фахівців фінансової сфери.

ПІПО АФУ використовує модулі: підтримки навчання, тестування рівня застосувань, інформаційного порталу, надання файлів, надання мультимедіа інтерфейсного рівня, які аналогічні модулям ІС ВУ (див. п. 4.3.4.).

Головна сторінка Порталу Інституту післядипломної освіти «Академії фінансового управління» представлена на рис. 4.15.

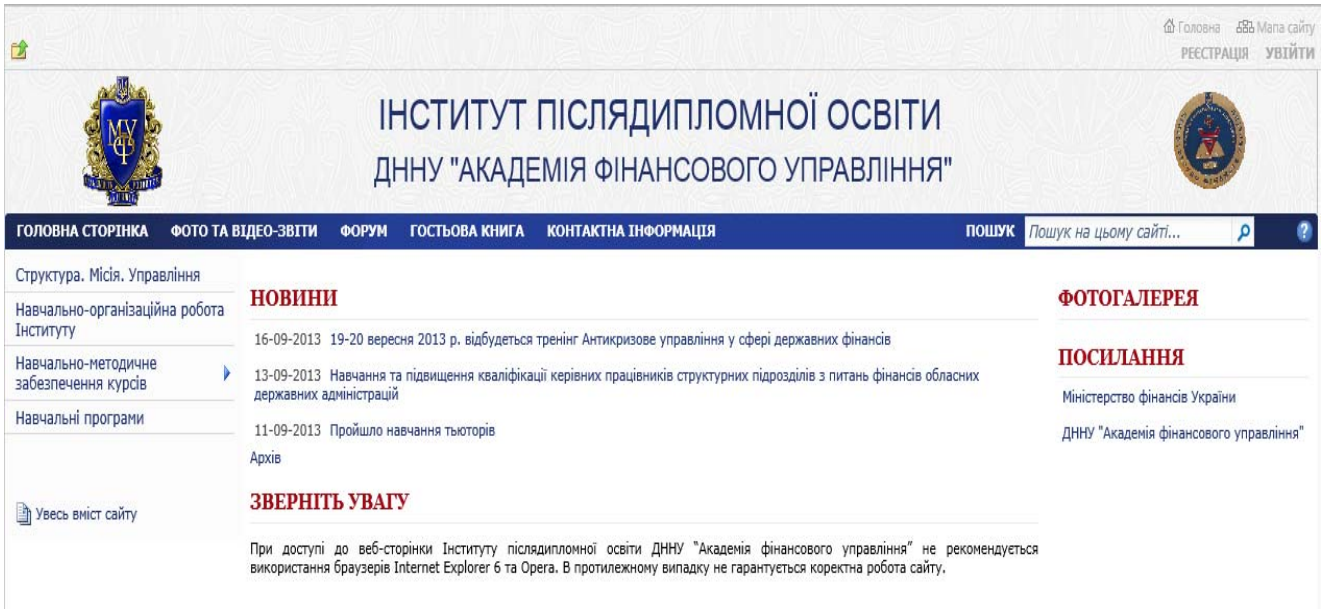


Рис. 4.15. Головна сторінка Порталу Інституту післядипломної освіти АФУ. [Розроблено автором]

На основі методу побудови МБСД була побудована схема для даного застосування, у якій дані про слухачів та викладачів Інституту були збережені в реляційній базі даних, вміст навчальних модулів – в базі даних NOSQL, а навчальні матеріали, що є неструктурованими даними, розміщено в файловому сховищі.

Це дозволило підвищити кількість одночасних сесій тестування з 200 до 250 (на сервер) без потреби виділення додаткових ресурсів.

4.4.6. Портал журналу «Фінанси України»

Портал журналу «Фінанси України» (ПЖ ФУ), виконаний як застосування Порталу періодичного наукового видання, забезпечує створення єдиного інформаційного простору для наукової підтримки прийняття рішень учасниками управління державними фінансами.

Портал журналу «Фінанси України»:

- представляє у світовому інформаційному просторі опубліковані в журналі «Фінанси України» наукові статті, інформацію про авторів статей, а також інформацію щодо умов та порядку опублікування статей;
- автоматизує процес подання авторами статей в редакційну колегію журналу «Фінанси України»;
- частково автоматизує роботу членів редакційної колегії та рецензентів журналу «Фінанси України»;
- веде різноманітну статистику по публікаціям журналу «Фінанси України».

Головна сторінка Порталу журналу «Фінанси України» представлена на рис. 4.16.

Рис. 4.16. Головна сторінка Порталу журналу «Фінанси України». [Розроблено автором]

Для функціонування ПЖ ФУ використовуються модулі підтримки наукової діяльності, підтримки діяльності редколегії рівня застосувань та модуль інформаційного порталу інтерфейсного рівня.

Модуль підтримки наукової діяльності надає доступ до даних Державного та місцевих бюджетів країни в обсязі, необхідному для проведення наукових досліджень в галузі державних фінансів.

Модуль підтримки діяльності редколегії – забезпечує процеси прийняття, редагування, рецензування статей, їх облік та формування номерів журналів.

Схема структури бази даних Порталу журналу «Фінанси України» представлена на рис. 4.17.

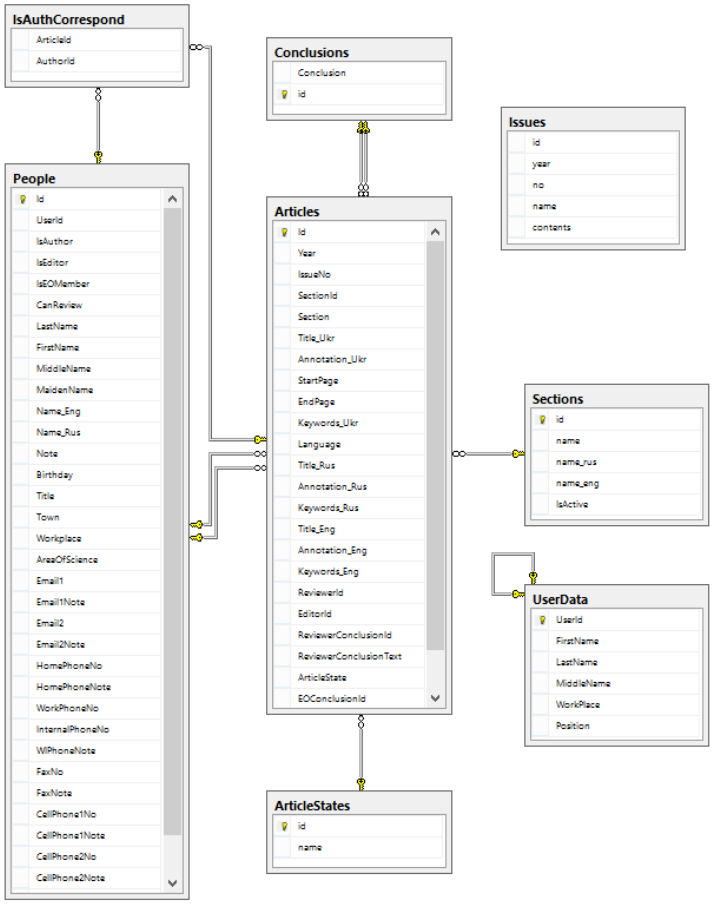


Рис. 4.17. Структурна схема бази даних Порталу журналу «Фінанси України». [Розроблено автором]

На основі методу побудови МБСД була побудована схема для даного застосування, згідно якої дані про редколегію, редакторів, рецензентів, авторів та статей були збережені в реляційній базі даних, окремі змінні їх властивості та розпізнані статті – в базі даних NOSQL, файли статей, що не можна якісно

розпізнати, які є неструктурованими даними, розміщено в файловому сховищі. Інфраструктура мультибазових сховищ даних надає уніфікований доступ до всіх статей.

Крім статей та метаданих журналу, що знаходяться в публічного доступі, на Порталі розміщуються засоби підтримки роботи з фінансовими моделями, що дозволяють отримати значення відповідних фінансових показників. Дані для цих фінансових моделей отримуються як з наборів, які надає користувач, так і з статей і матеріалів журналу. У останньому випадку використовують можливості визначення та аналізу даних сховища, які дозволяють оптимізувати збереження та обробку різнорідних даних.

Таким чином, впровадження запропонованої ІТ надати можливість ефективного опрацювання даних фінансових моделей, що мають гетерогенний характер.

Дані для порталу розміщуються в єдиному МБСД, яке також зберігає і обробляє дані порталів «Віртуальний Університет» та «Інститут післядипломної освіти», тому додаткових витрат на збереження та обробку даних не потрібно.

4.5 Ефективність використання ІТ побудови РСД ГТ

Ефективність використання інформаційної технології проектування розподілених сховищ даних визначається:

- сукупною вартістю збереження та обробки даних у побудованому сховищі;
- часом побудови сховища даних;
- швидкодією запитів (відсутністю запитів, які її запізнюються);
- кількістю паралельних реплікованих даних.

Для оцінки цих показників був проведений набір експериментів над тестовими даними. У додатку З наведена структура детермінованих схем БД з деталізацією за областями сховища даних в обсязі, достатньому для розуміння подальших даних. Приведені відомості про джерела даних, їх області з кількістю таблиць більше 1, таблиці цих областей даних. Також вказана

структурованість даних і розміщення у сховищі (порядковий номер області даних і тип БД).

Під час цього набору експериментів виконано серію запитів до них - 320 запитів, з них 165 запізнилися. На діаграмі (рис. 4.18) наведений розподіл кількості запитів і кількості запитів, що запізнилися, по областях. Запитами, які запізнилися, вважаються ті, швидкодія яких менша за 15000 1/мс.

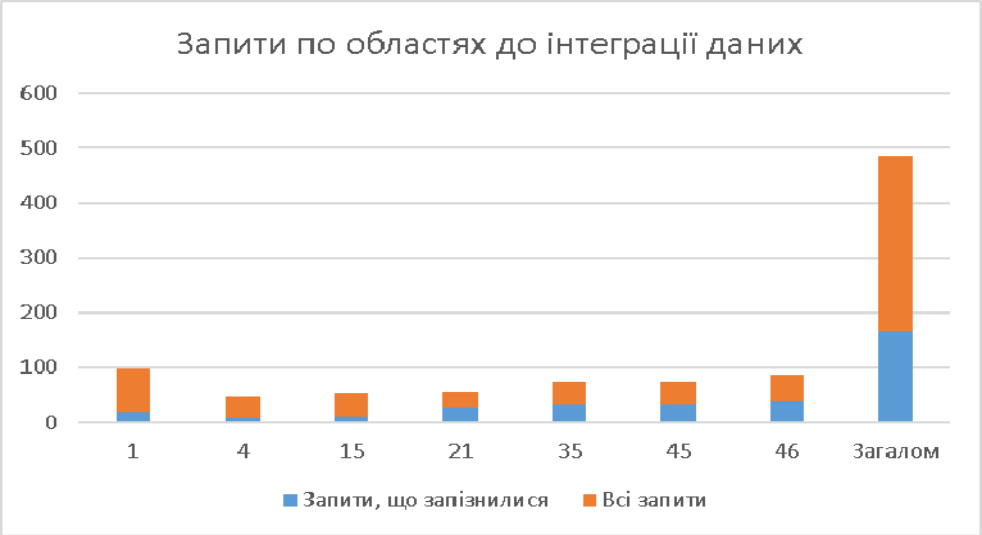


Рис. 4.18. Діаграма загальної кількості запитів по областях і кількості запитів по областях, що запізнилися. [Розроблено автором]

Діаграма мінімальної, середньої та максимальної швидкодії виконання запитів до джерел даних показана на рис. 4.19. На діаграмі відображені показники мінімальної, середньої та максимальної швидкодії виконання запитів у кожній області даних. Мінімальна швидкодія за сховищем складає 0,96 1/мс, середня - 554543,8278 1/мс, 19533753,64 1/мс.

Розмістимо дані у сховищі без використання логічного та фізичного розподілу даних. Розміщення областей відображено в додатку К.

Виконаємо ці ж запити у сховищі з наступною конфігурацією розгортання, що відповідає варіанту розміщення 1111 для центрального вузла. Тоді отримаємо наступні показники швидкодії виконання запитів до сховища даних по областях (рис 4.20).

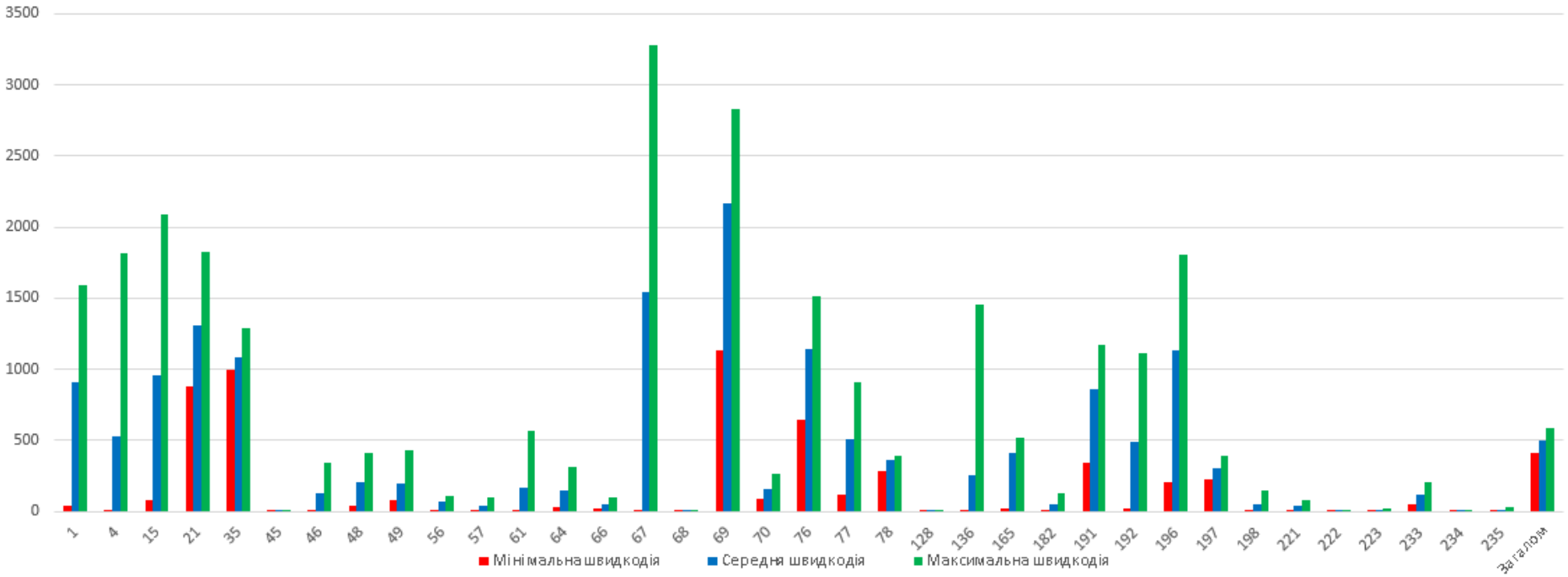


Рис. 4.19. Показники швидкодії виконання запитів до джерел даних по областях (вісь абсцис – номери областей, вісь ординат – швидкодія виконання запитів, елементів даних за мілісекунду, 1/мс). [Розроблено автором]

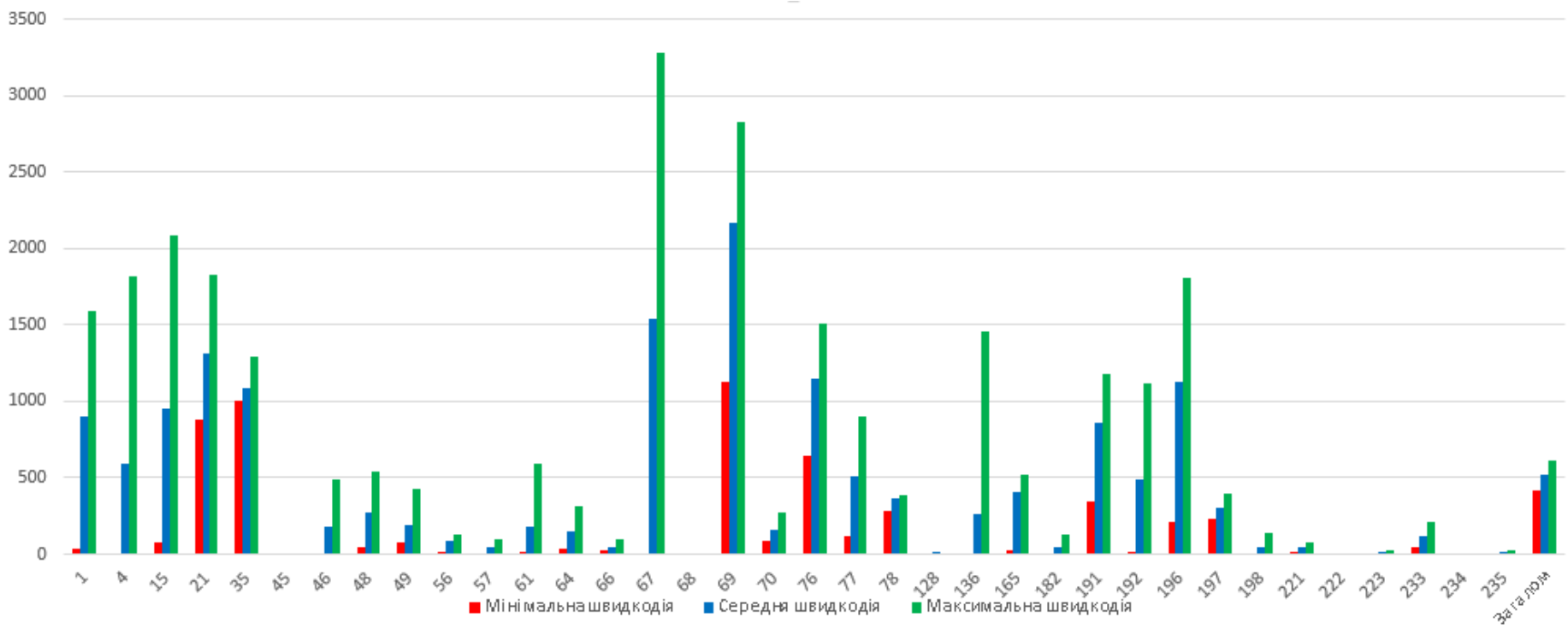


Рис. 4.20. Показники швидкодії виконання запитів до сховища даних по областях (при використанні лише центрального вузла). [Розроблено автором]

Розрахуємо за цими формулами мінімальна швидкодія виконання запитів складає 1,54 1/мс, середня - 580267 1/мс, максимальна – 23041474,7 1/мс.

Наведемо співвідношення кількості запитів, що запізнилися в джерелах і у сховищі (рис 4.21.).

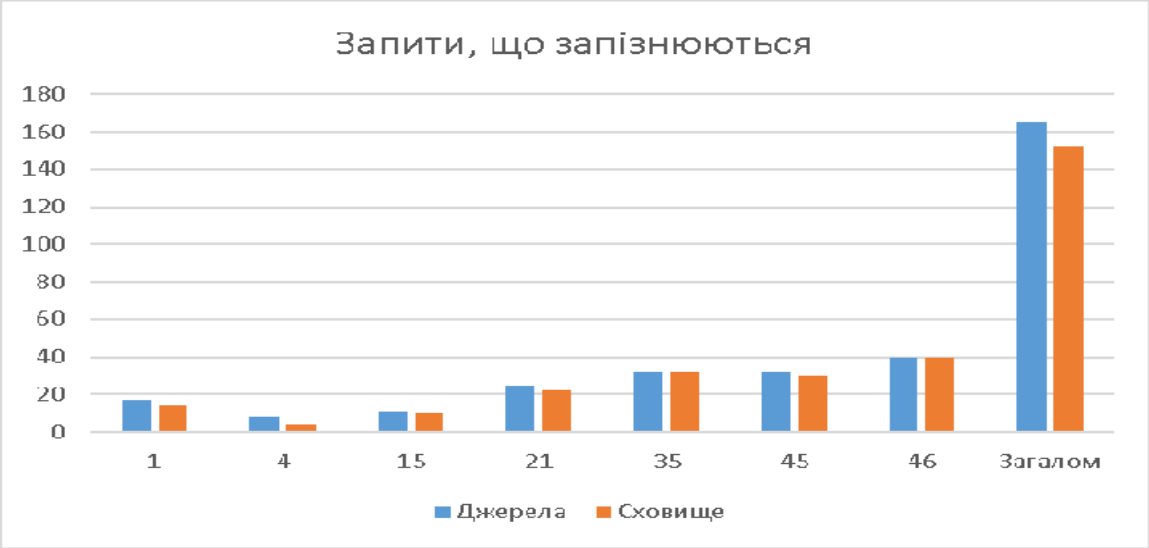


Рис. 4.21. Кількість запитів, що запізнюються, у джерелах даних, і у сховищі (при розміщенні у одному центральному вузлі). [Розроблено автором]

У результаті виконання 3353 запитів до сховища з них: 2360 запитів виконалося вчасно, 993 запізнилися. Отримаємо наступні показники роботи сховища:

- сукупна вартість збереження та обробки даних у побудованому сховищі складає 484000 грн.;
- у додаткових вузлах 20..25 (додатковим вузлом тут називаються вузол, який не має визначених носіїв і відповідних областей, однак може стати регіональним шляхом реплікації однієї чи декількох областей сховища і виконувати запити до цих даних) реплікуються наступні області (табл.4.1.).

Таблиця 4.1.

Розміщення даних областей у додаткових вузлах. [Розроблено автором]

№ областей	№ вузла	Тип вузла
1;11;13..19;21..29;31..33	21	Регіональний
33..39;41..42;44..49;51..53	22	Регіональний
53..59;61..69	23	Регіональний
69;71..78;202..204;206..209;211..213	24	Регіональний
213..229	25	Регіональний

Після виконання логічного та фізичного розподілу отримуємо наступні значення показників:

- з виконаних 3353 запитів: всі виконалося вчасно, 0 запізнилися;
- сукупна вартість збереження та обробки даних у побудованому сховищі складає 357000 грн.;
- дані кожної області реплікуються щонайменше у двох вузлах;
- загальний час без модифікації генетичного алгоритму складає 109720.3 с, з використанням фази попереднього пошуку – 53058.47 с. (зменшення до 50%).

Дані експериментальні результати підтверджують виконання поставлених вимог до інформайційної технології.

Висновки по розділу 4

У даному розділі розглянуто ІТ побудови РСД ГТ на базі комплексу математичних моделей і методів аналізу та оцінювання процесів опрацювання даних у сховищах зазначеного типу і оптимізації їх структури, що дозволяє створювати ІС, у яких ефективно обробляються гетерогенні дані, що зберігаються розподілено.

ІТ побудови РСД ГТ було використано у застосуваннях при створенні інформаційних та інформаційно-аналітичних систем Міністерства фінансів України, для чого розроблено архітектуру ТСУДФ, яка дозволяє використати переваги МБСД, зокрема ефективну обробку розподілених даних різних видів. При цьому бази даних, що використовуються, можуть бути різними в межах визначених класів (реляційні та багатовимірні БД, БД XML та NoSQL), що розширює можливості застосування МБСД.

Досягнуті при впровадженні зменшення вартості роботи сховищ даних на 30% і скорочення часу виконання запитів на 33% за рахунок оптимізації їх виконання між різними базами даних і фізичними вузлами разом з результатами експериментальних досліджень підтвердили практичну значимість розробленої інформаційної технології і дають підстави вважати, що мету дослідження досягнуто.

Результати даного розділу опубліковані в роботах [108..115]

ВИСНОВКИ

У результаті виконання дисертаційної роботи вирішено науково-практичне завдання створення інформаційної технології побудови РСД ГТ та одержані результати, що мають істотні переваги перед існуючими результатами:

1. На основі проведеного аналізу існуючих підходів і методів побудови розподілених сховищ даних встановлено, що задача побудови РСД ГТ може бути вирішена за допомогою методів та моделей логічного та фізичного розподілу даних у сховищі з врахуванням властивостей джерел даних і статистики виконання запитів до сховища.

2. Розроблено новий метод побудови РСД ГТ з мінімальною сукупною вартістю збереження та обробки даних, що здійснює логічний і фізичний розподіл даних та відрізняється від існуючих тим, що враховує інформацію про дані сховища та запити до цих даних за допомогою методу спрощення формул алгебричної системи СД та модифікованого генетичного алгоритму.

3. Для формування метаопису сховища даних та оптимізації запитів до нього вперше запропоновано формальну алгебричну систему сховищ даних гібридного типу, яка відрізняється від існуючих набором основ (сутності, елементи даних РБД, БВБД, БД XML і NoSQL) та операцій (новими є операції перетворення основ між собою, виділення структурованої і слабко-структурованої частини відношення, з’єднання сутностей, побудови зв’язної множини), дослідження властивостей яких дозволило оперувати даними, представленими різними моделями, та будувати для них РСД ГТ.

4. Для логічного розподілу даних удосконалено метод спрощення формул алгебричної системи РСД ГТ у сховищі з визначенням перспективних множин інваріантів з наступним деталізованим вибором кращого у цих множинах, який відрізняється від існуючих наявністю правил оцінки інваріантів формули запиту, що дозволяє отримати множинний інваріант запитів до сховища з мінімальною сукупною вартістю збереження та обробки даних. Для фізичного розподілу даних у сховищі розроблено модифікований

генетичний алгоритм з фазою формування початкової популяції, яка базуються на методики бджолиного рою, що дозволяє зменшити витрати часу на знаходження конфігурацій розгортання і реплікації та здійснити фізичний розподіл даних у сховищі на 10 % порівняно з генетичним алгоритмом та на 50% - порівняно з бджолиним.

5. Розроблено інформаційну технологію побудови РСД ГТ на базі комплексу математичних моделей і методів аналізу та процесів опрацювання даних у сховищах зазначеного типу з оптимізацією їх структури, що дозволяє створювати інформаційні системи з РСД ГТ, причому час проектування і реалізації останніх скорочується у 2-3 рази, а вартість підтримки зменшується на 50%.

6. Впровадження створеної ІТ при розробці низки інформаційних та інформаційно-аналітичних систем для Міністерства фінансів України дозволило отримати суттєвий ефект для:

– системи управління державними фінансами Міністерства фінансів України – знизити до 25-30 % вартість системи збереження та обробки даних;

– інформаційної системи «Віртуальний університет» Міністерства фінансів України та порталу Інституту післядипломної освіти ДННУ «Академія фінансового управління» - підвищити пропускну спроможність сховища даних, а відтак і кількість одночасних сесій для навчання та тестування з 200 до 250 на сервер без потреби виділення додаткових ресурсів;

– інформаційно-аналітичної системи «Прозорий бюджет» Міністерства фінансів України - підвищити на 30-33 % швидкодію виконання запитів і у порівнянні з варіантом без використання запропонованої ІТ;

– порталу журналу «Фінанси України» - надати можливість ефективного опрацювання даних фінансових моделей, що мають гетерогенний характер.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Debbarma N. Analysis of Data Quality and Performance Issues in Data Warehousing and Business Intelligence / N. Debbarma, G. Nath, H. Das // International Journal of Computer Applications – 2013 – Volume 79, No 15 – P.20-26.

2. Kimpel J. F. Critical Success Factors for Data Warehousing : a classic answer to a modern question / J. F. Kimpel // Issues in Information Systems – 2013 Volume 14, Issue 1 —P.376-384.

3. Chan Joseph O. Optimizing Data Warehousing Strategies / Joseph O. Chan // Communications of the IIMA – 2005 – Vol. 5: Iss. 1, Article 1. – P.1-14.

4. Jarke M. Architecture and Quality in Data Warehouses / M Jarke, M.A. Jeusfeld, C. Quix, P. Vassiliadis // Seminal Contributions to Information Systems Engineering – 2013 – P.183-189.

5. Yuan Y.. The Yin and Yang of Processing Data Warehousing Queries on GPU Devices / Y. Yuan, R. Lee, X. Zhang // Proceedings of the VLDB Endowment– 2013 – Vol. 6, No. 10 – P.817-828.

6. Jarke M. Fundamentals of data warehouses / M. Jarke, M. Lenzerini, Y. Vassiliou, P. Vassiliadis // Springer-Verlag Berlin, Heidelberg New York, 2013 – 224 p.

7. Nayem R. Building Data Warehouses Using Automation / Nayem R. // International Journal of Intelligent Information Technologies (IJIIT) – 2015 – Volume 11, Issue 2 –P. 1-22.

8. Song L. Study on Building and Applying Technology of Multi-level Data Warehouse/ L Song, Y Zhan, Y Li // International Conference on Advances in Mechanical Engineering and Industrial Informatics (AMEII 2015) – 2015 – P. 591-596.

9. Gosain A. Architecture Based Materialized View Evolution: A Review / A.Gosain, S.Sabharwal, R.Gupta // Procedia Computer Science – Volume 48, 2015, – P. 256–262.

10. Blanco C. An architecture for automatically developing secure OLAP applications from models / Blanco C., de Guzmán I.G.R, Fernández-Medina E. - Information and Software Technology - Volume 59, March 2015, – P. 1–16.

11. Gupta A. Amazon Redshift and the Case for Simpler Data Warehouses/ A. Gupta, D. Agarwal, D. Tan, J. Kulesza, R. Pathak, S. Stefani, V. Srinivasan // SIGMOD '15 Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data – 2015 – P. 1917-1923

12. Cuzzocrea A. Data Warehousing and OLAP over Big Data: Current Challenges and Future Research Directions / A. Cuzzocrea, L. Bellatreche, I.-Y. Song // DOLAP '13 Proceedings of the sixteenth international workshop on Data warehousing and OLAP – 2013 – ACM New York, NY, USA – P. 67-70.

13. Boulmakoul A.. MongoDB-Hadoop Distributed and Scalable Framework for Spatio-Temporal Hazardous Materials Data Warehousing / A. Boulmakoul, L. Karim, M. H. Laarabi, R. Sacile, E. Garbolino // Proceedings of the 7th International Congress on Environmental Modelling and Software, June 15-19, San Diego, California, USA., San Diego, California, USA.; 06/2014. – P. 2255-2262

14. Gupta A. Mesa: Geo-Replicated, Near Real-Time, Scalable Data Warehousing / A. Gupta, F. Yang, J. Govig, A. Kirsch, K. Chan et al. // Proceedings of the VLDB Endowment –2014 – Vol. 7, No. 12 — P.1259-1270.

15. Sharma R. Grid Cloud Computing and Data Warehousing / R. Sharma, G. Sharma// International Journal of Advance Research in Computer Science and Management Studies – 2015 – Volume 3, Issue 4, April.

16. Tian Y. Joins for Hybrid Warehouses: Exploiting Massive Parallelism in Hadoop and Enterprise Data Warehouses / Y. Tian, T. Zou, F. Özcan, R. Goncalves, H. Pirahesh // 18th International Conference on Extending Database Technology (EDBT), March 23- 27, 2015 – P.373-384.

17. Viswanathan G. User-Centric Spatial Data Warehousing: A Survey of Requirements & Approaches / G. Viswanathan, M. Schneider // Int. J. Data Mining, Modelling and Management – 2014 – Vol. 6, No. 4, – P.1-21.

18. Aji A. A High Performance Spatial Data Warehousing System over MapReduce Proceedings VLDB Endowment. / A. Ablimit , F. Wang, H. Vo, R. Lee, Q. Liu, X. Zhang, and J. Saltz // 2013 – Hadoop-GIS: 6(11) – p.1009-1020.

19. Aji A. Demonstration of Hadoop-GIS: A Spatial Data Warehousing System Over MapReduce / A. Aji, X. Sun, H. Vo, Q. Liu ,R. Lee, X. Zhang, J. Saltz , F. Wang // SIGSPATIAL'13, Nov. 05–08 2013, Orlando, FL, USA – P. 528-531 .

20. Triplet T. A review of genomic data warehousing systems / Briefings in Bioinformatics / T. Triplet, G. Butler // Advance Access – 2013 – № 1 – P. 1-13.

21. Neogi S. G.. MoDa-A Data Warehouse for Multi-"Omics" Data / S. G. Neogi, P. Vasilis, M. Krestyaninova, M. Kapushesky, I. Emam, A. Brazma, U. Sarkans // Data Mining Genomics Proteomics – 2013 – Vol. 4 Issue 5 – p.1-6.

22. Quiroz Andres A Robust and Extensible Tool for Data Integration Using Data Type Models / Andres Quiroz, Eric Huang, Luca Ceriani // Proceedings of the Twenty-Seventh Conference on Innovative Applications of Artificial Intelligence, 2015 – P. 3993-3998.

23. Cal A. Data Integration under Integrity Constraints / A. Cal, D. Calvanese, G. De Giacomo, M. Lenzerini// CAISE 2002, LNCS 2348 – 2002 – P. 262-279.

24. Jeffery S.R. Arnold: Declarative Crowd-Machine Data Integration / S.R. Jeffery, L. Sun, M. DeLand, N. Pendar , R. Barber, A. Galdi // 6th Biennial Conference on Innovative Data Systems Research (CIDR '13) January 6-9, 2013, Asilomar, California, USA – P. 262-279.

25. Gupta A. Amazon Redshift and the Case for Simpler Data Warehouses/ A. Gupta, D. Agarwal, D. Tan, J. Kulesza, R. Pathak, S. Stefani, V. Srinivasan // SIGMOD '15 Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data – 2015 – P. 1917-1923.

26. Cuzzocrea A. Data warehousing and OLAP over big data: current challenges and future research directions / L. Bellatreche, A. Cuzzocrea, Y. Song // DOLAP '13 Proceedings of the sixteenth international workshop on Data warehousing and OLAP - ACM New York, NY, USA – 2013 – P.67-70.

27. Aligona J. A collaborative filtering approach for recommending OLAP sessions / Julien Aligona, Enrico Gallinuccib, Matteo Golfarelli,Patrick Marcela,Stefano Rizzib, Golfarelli M. // Decision Support Systems – 2015 – Volume 69, January 2015 – Pages 20-30

28. Watson H.J. Tutorial: Big data analytics: Concepts, technologies, and applications / Communications of the Association for Information Systems – 2014 – Volume 34, Article 65, April 2014 – P. 1247-1268.

29. Eckerson W. Making Data Governance Work: Tales from the Trenches / W. Eckerson, D.Power // Technics Pubns Llc, 2015 – 275 p.

30. Stöhr T. Multi-Dimensional Database Allocation for Parallel Data Warehouses / T. Stöhr, H. Martens, E. Rahm // Proceedings of the 26th International Conference on Very Large Databases, Cairo, Egypt, 2000 – P.273-284

31. Lauer T. Exploring graphics processing units as parallel coprocessors for online aggregation / T. Lauer, A. Datta, Z. Khadikov, C. Anselm // DOLAP '10 Proceedings of the ACM 13th international workshop on Data warehousing and OLAP – P. 77-84.

32. Costa J.P. Data Warehouse Processing Scale-Up for Massive Concurrent Queries with SPIN / J.P. Costa, P. Furtado // Transactions on Large-Scale Data- and Knowledge-Centered Systems XVII, Volume 8970 of the series Lecture Notes in Computer Science – 2015 – P. 1-23

33. Wu X. Optimizing XML queries: Bitmapped materialized views vs. indexes/ X. Wu, D. Theodoratos, W.H. Wang, T. Sellis // Information Systems - Volume 38, Issue 6, September 2013 – P. 863–884

34. Zhang C. An evolutionary approach to materialized views selection in a data warehouse environment / C. Zhang, X. Yao, J. Yang // IEEE Transactions on Systems, Man and Cybernetics – 2001. — P. 282-294.

35. Gupta H. Selection of views to materialize in a data warehouse // Knowledge and Data Engineering, IEEE Transactions on (Volume:17 , Issue: 1) – 2005 – P. 24-43

36. Horng J.T. Applying evolutionary algorithms to materialized view selection in a data warehouse / J.T. Horng, Y.J. Chang, , B.J. Liu // Soft Computing - August 2003, Volume 7, Issue 8, P. 574-581

37. Horng J.T. A mechanism for view consistency in a data warehousing system / J.T. Horng, C.-W. Chen // Journal of Systems and Software – 2001 – Volume 56, Issue 1, 1 February 2001 – P. 23-37.

38. Inmon W.H. Corporate information factory / W. H. Inmon, C. Imhoff, R. Sousa // John Wiley & Sons, 2002 – 400 p.

39. Inmon W.H. Corporate Information Factory Components [Електронний ресурс] / W. H. Inmon. Режим доступу: http://www.inmoncif.com/view/26

40. Kimball R. The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouse / New York : John Wiley & Sons, 2000 – 374 p.

41. Kimball R..The data warehouse toolkit: the complete guide to dimensional modeling / Ralph Kimball // Wiley- 2002 – 436 p.

42. Hackney. D. Architectures and Approaches for Successful Data Warehouses [Електронний ресурс] / D. Hackney. Режим доступу : http://www.egltd.com/presents/ArchitecturesApproaches.pdf

43. Пашнєв А. А. Формування логічної структури розподіленого гомогенного сховища даних / А. А. Пашнєв, О. В. Курко, О. В. Мігура // Системи обробки інформації. – 2006. – Вип. 2. – С. 107-109.

44. Кучук Г.А. Формалізація предметної області багатовимірних баз даних / Г. А. Кучук // Системи обробки інформації. – 2001. – № 1. – С. 110-114.

45. Кучук Г.А. Математическая модель функционирования специализированного программного комплекса в среде гетерогенной мультисервисной сети / Г. А. Кучук, А. В. Петров, Р. В. Королёв // Системи обробки інформації. – 2012. – № 3. – С. 191-198.

46. Кучук Г.А. Метод синтезу логічної структури мережевої бази даних / Г. А. Кучук // Системи обробки інформації. – 2001. – № 2. – С. 32-37.

47. Приймак Д.В., Акимишин О.І. Шляхи побудови конфігурованих сховищ даних на основі платформи Hadoop. [Електронний ресурс] / Д.В. Приймак, О.І. Акимишин // Режим доступу : http://eom.lp.edu.ua/seminar/spr/pryjmak.doc

48. Шаховська Н. Б. Сховища та простори даних. Монографія / Шаховська Н. Б., Пасічник В. В.// Львів: Видавництво Львівської політехніки, 2009. – 244 с.

49. Організація просторів даних у складних інформаційних системах :дис. ... д-ра техн. наук : 05.13.06 / Шаховська Наталія Богданівна ; Нац. ун-т «Львів. політехніка». — Л., 2012. — 452 арк. — Бібліогр.: арк. 351-387.

50. Томашевський В.М. Математична модель задачі проектування гібридних сховищ даних з врахуванням структур джерел даних / В.М. Томашевський, А.Ю. Яцишин // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка : зб. наук. пр. – 2011. – № 53. – С. 54–61.

51. Томашевський В.М. Особливості проектування гібридних сховищ даних з врахуванням джерел даних / В.М. Томашевський, А.Ю. Яцишин // Інформаційні системи та мережі: збірник наукових праць. – 2011. – № 715. – С. 246–254.

52. Date C.J. Databases, Types, and the Relational Model: The Third Manifesto (Third Edition) / C.J Date, H. Darwen // Mass Reading : Addison-Webley, 2006 – 572 p.

53. Eckerson W. Four Ways to Build a Data Warehouse [Електронний ресурс] / W. Eckerson // Режим доступу : http://www.tdwi.org/research/display.aspx?ID=6699

54. Franconi E. The GMD Data Model and Algebra for Multidimensional Information. / E. Franconi, A. Kamble // In Proc. 5th International Conference on Data Warehousing and Knowledge Discovery. – 2003. – P.55-65.

55. Franconi E. A data warehouse conceptual data model for multidimensional aggregation: a preliminary report / E. Franconi, U. Sattler // Journal of the Italian Association for Artificial Intelligence. – 1999. – P.9-21.

56. Фищенко В.К. Использование вейвлет-преобразования для оптимального оценивания тренда случайного процесса / В.К. Фищенко, Е. Л. Кулешов // «Автометрия». – 2003. – Т. 39, № 1. – С. 103 - 113.

57. Федоров А. Хранилище данных / Федоров А., Елманова Н. // Компьютер пресс. – 2001. – №5. – С.137-145.

58. Черняк Л. Хранилища и карты данных [Електронний ресурс] // Открытые системы. – 2005. – №47 – Режим доступу : http://www.osp.ru/cw/2005/47/373696/.

59. Gray J. Data cube: a relational aggregation operator generalizing group-by, cross-tabs and subtotals. / J. Gray, A. Bosworth, A. Layman, H. Pirahesh // In Proc. of ICDE-96. – 1996. – P.121-143.

60. Cabibbo L. A logical approach to multidimensional databases. / L. Cabibbo, R. Torlone // In Proc. of EDBT-98. – 1998. – P.23-29.

61. Agrawal R. Modeling multidimensional databases. / R. Agrawal, A. Gupta, S. Sarawagi // In Proc. of ICDE-97. – 1997. – P. 56-72.

62. Vassiliadis P. Modeling and optimisationissues for multidimensional databases. / P. Vassiliadis, S. Skiadopoulos // In Proc. of CAiSE-2000. – 2000. – P. 482-497.

63. Jagadish H.V. What can hierarchies do for data warehouses? / H.V. Jagadish, V.S. Lakshmanan, D. Srivastava // In Proc. 25th International Conference on Very Large Databases (VLDB). – 1999. – P.530-541.

64. Golfarelli M. Data Warehouse Design : Modern Principles and Methodologies. / Golfarelli M., Rizzi S. // McGrawHill, 2009 – 480 p.

65. Watson H.J. Data Warehouse Architectures: Factors in the Selection Decision and the Success of the Architectures [Електронний ресурс] / H.J. Watson, T. Ariyachandra // Режим доступу : http://www.terry.uga.edu/~hwatson/DW_Architecture.

66. Golfarelli M. Designing the Data Warehouse: Key Steps and Crucial Issues / M. Golfarelli, S. Rizzi // Journal of Computer Science and Information Management – 1999 – Vol. 2, No. 1 – P. 1-14.

67. Tsois A. MAC: Conceptual Data Modeling for OLAP / A. Tsois, N. Karayannidis, T. Sellis // Proceedings of the Intl. Workshop on DMDW 2001, Interlaken, Switzerland, June 4, 2001. – P. 5-11.

68. Carsten Sapia Extending the E/R Model for the Multidimensional Paradigm. / Carsten Sapia, Markus Blaschka, Gabriele Höfling, Barbara Dinter.// Advances in Database Technologies Lecture Notes in Computer Science Volume 1552, 1999 – P. 105-116.

69. Nectaria Tryfona. starER: A Conceptual Model for Data Warehouse Design / Nectaria Tryfona, Frank Busborg, and Jens G. Borch Christiansen // DOLAP '99 Proceedings of the 2nd ACM international workshop on Data warehousing and OLAP – 1999 – P. 3-8.

70. ISO/IEC/IEEE 24765:2010. [Електронний ресурс] // Режим доступу : http://www.egltd.com/presents/ArchitecturesApproaches.pdf

71. Павлов О.А. Інформаційні технології та алгоритмізація в управлінні / О.А. Павлов, С.Ф. Теленик // К.: Техніка, 2002. – 344 с.

72. Phipps Cassandra. Automating Data Warehouse Conceptual Schema Design and Evaluation / P. Cassandra // 4th International Workshop on Design and Management of Data Warehouses. – 2002 – P.1-10.

73. Luján-Mora S. A UML profile for multidimensional modeling in data warehouses. / S. Luján-Mora, J. Trujillo, I.Y. Song // Data & Knowledge Engineering, – 2006 – 59(3) – P. 725–769.

74. Abello A. A multidimensional conceptual model extending UML / A. Abello , J. Samos, F. Saltor // Information Systems – 2006 – 31(6) – P. 541–567.

75. Golfarelli M. The DFM: A conceptual model for data warehouse. / M. Golfarelli // Encyclopedia of data warehousing and mining (2nd ed.) – 2008 – P. 638-640.

76. Golfarelli M. Conceptual design of data warehouses from E/R schemes / M. Golfarelli , D. Maio , S. Rizzi // Proceedings of the Thirty-First Hawaii International Conference on System Sciences –1998 – vol.7 – P. 334 – 343.

77. Vrdoljak B. Automating conceptual design of web warehouses. / B. Vrdoljak, M. Banek, S. Rizzi // Proceedings of the 7th International Conference on Telecommunications (ConTEL 2003) – 2003 – vol.2 – P. 535 – 542.

78. Trujillo J. Designing Data Warehouses with OO Conceptual Models. / J. Trujillo, M. Palomar , J. Gomez, I.-Y. Song // Computer – 2001 – Volume:34, Issue: 12 – P.66-75.

79. Lin W.-Y. A Genetic Selection Algorithm for OLAP Data Cubes / W.-Y. Lin,I-C. Kuo // Knowledge and information systems – 2004 – vol. 6 –Volume 6, Issue 1 – P. 83-102.

80. Peralta V. On the Applicability of Rules to Automate Data Warehouse Logical Design / V. Peralta, A. Illarze, R. Ruggia // In Proceedings of the Decision Systems Engineering Workshop 2003, Klagenfurt, Austria – 2003 – P. 317-328.

81. Ouaret Z. Towards the Automation of Data Warehouse Logical Design: a Rule-Based Approach / Z. Ouaret, O. Boussaid, R. Chalal // 2014 Ninth International Conference on Digital Information Management (ICDIM) – 2014 – P. 27-32.

82. O.Boussaid. X-Warehousing: An XML-Based Approach for Warehousing Complex Data / O. Boussaid, R. Ben Messaoud, R. Choquet, S. Anthoard // ADBIS'06 Proceedings of the 10th East European conference on Advances in Databases and Information Systems, Springer-Verlag Berlin, Heidelberg, 2006 – P. 39-54.

83. Yu J. X. Materialized view selection as constrained evolutionary optimization / J. X. Yu, X. Yao, C.-H. Choi, and G. Gou // In IEEE Transactions on Systems, Man and Cybernetics – Part C, volume 33 – 2003. – P. 458-467.

84. Zhang C. Genetic algorithm for materialized view selection in data warehouse environments / C. Zhang // Lecture Notes in Computer Science – 2002 – Volume 1676 – P. 116-125

85. Zhang C. Evolving Materialized views in a Data Warehouse / C. Zhang, X. Yao, J. Yang // Evolutionary Computation – 1999 – Volume 2 – P. 823-829.

86. Lawrence M. Multiobjective genetic algorithms for materialized view selection in OLAP data / Proceedings of the 8th annual conference on Genetic and evolutionary computation – 2002 – P. 699-706.

87. Elhoussaine Z. Complete Algorithm for fragmentation in Data warehouse / Z. Elhoussaine, D. Aboutajdine, E. Q. Abderrahim // 7th WSEAS Int. Conf. on ARTIFICIAL INTELLIGENCE, KNOWLEDGE ENGINEERING and DATA BASES (AIKED'08) – 2008 – P. 537-540.

88. Velinov G. Framework for Generalization and Improvement of Relational Data / G. Velinov , D. Gligoroski, M. Kon Popovska // IJCSNS International Journal of Computer Science and Network Security – 2008 – VOL.8 No.3. – P. 32-40.

89. Stöhr T. Multi-Dimensional Database Allocation for Parallel Data Warehouses / T. Stöhr, H. Märtens, E. Rahm // Proceedings of the 26th International Conference on Very Large Data Bases, Morgan Kaufmann Publishers Inc. San Francisco, CA, USA, 2000 – P. 273-284.

90. Datta A. A Case for Parallelism in Data Warehousing and OLAP / A. Datta , B. Moon , H. Thomas // The 9th International Workshop On Database And Expert Systems Applications (Dexa98) – 1998 – P. 226-231.

91. Furtado P.. Experimental Evidence on Partitioning in Parallel Data Warehouses / P. Furtado // Proceedings of the 7th ACM international workshop on Data warehousing and OLAP, New York, NY, USA, 2004, — P. 23-30.

92. Gupta A. Virtual database technology / Gupta A. , Harinarayan V. , Rajaraman A. // Proceedings of 14th International Conference on Data Engineering, 1998 – P. 297-301.

93. Using HDP for Hadoop Platform-as-a-Service [Електронний ресурс] // Режим доступу: http://hortonworks.com/blog/using-hdp-hadoop-platform-service/

94. Apache Hadoop Tutorial. Module 4: MapReduce[Електронний ресурс] / Режим доступу : https://developer.yahoo.com/hadoop/tutorial/module4.html

95. Yuan Y. VDB-MR: MapReduce-based distributed data integration using virtual database. / Y. Yuan, Y. Wu, , X. Feng, J. Li, G. Yang, W. Zheng // Future Generation Computer Systems – 2010 – Volume 26, Issue 8 – P. 1418–1425.

96. Яцишин А.Ю. Підходи та алгоритми проектування гібридних сховищ даних / А.Ю. Яцишин // Інформаційні системи та мережі : збірник наукових праць. – 2010. – № 673. – С. 205–213.

97. Яцишин А.Ю. Застосування генетичного алгоритму для проектування гібридних сховищ даних / А.Ю. Яцишин // Вісник Національного університету "Львівська політехніка" : Інформаційні системи та мережі. – 2010. – № 689. – С. 262–270.

98. Яцишин А.Ю. Побудова мультибазових сховищ даних на основі структурованості даних та запитів / А.Ю. Яцишин // Східно-Європейський журнал передових технологій. – 2015. – № 1/2(73). – С. 11–17.

99. Яцишин А.Ю. Проектування узагальненого гібридного сховища даних / Наука і вища освіта : тези доповідей учасників XVIII Міжнар. наук. конф. студентів і молодих учених (Запоріжжя, 22–23 квітня 2010 р.) : у 4-х т. / Класичний приватний університет. – Запоріжжя : Вид-во КПУ. – 2010. – Т. 3. – С. 248–249.

100. Яцишин А.Ю. Особливості інтеграції даних при проектуванні гібридних сховищ даних / А.Ю. Яцишин // Збірник наукових праць VI Міжнародної науково-практичної конференції "Наука і соціальні проблеми суспільства: інформатизація та інформаційні технології. – Харків : ХНУРЕ, 2011. – С. 405–406.

101. Яцишин А.Ю. Побудова математичної моделі задачі проектування гібридних сховищ даних з врахуванням структур джерел даних / А.Ю. Яцишин // «Сучасні комп'ютерні інформаційні технології» : матеріали всеукраїнської школи-семінару молодих вчених та студентів ACIT'2011. – Тернопіль : Економічна думка, 2011. – С. 142–144.

102. Яцишин А.Ю. Проектування гібридних сховищ даних як задача оптимізації / А.Ю. Яцишин // Сучасні інформаційні системи та технології :

матеріали І науково-практичної конференції (Суми, 15–18 травня 2012 р.) / редкол.: А.С. Довбиш, О.А. Борисенко, І.В. Баранова. – Суми : Сумський державний університет, 2012. – С. 39–40.

103. Яцишин А.Ю. Проектування мультибазових сховищ даних на основі двохфазного алгоритму / А.Ю. Яцишин // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка : зб. наук. пр. – 2012. – № 55. – С. 125–130.

104. Яцишин А.Ю. Проектування гібридних сховищ даних з врахуванням структурованості даних / А.Ю. Яцишин // Управління розвитком складних систем. – 2012. – № 9. – С. 59–65.

105. Яцишин А.Ю. Врахування джерел даних у проектуванні сховища даних для системи «Прозорий бюджет» Міністерства фінансів України / А.Ю. Яцишин // Збірник доповідей учасників XIII всеукраїнської науково-практичної конференції "Інноваційний потенціал української науки – XXI сторіччя" (Запоріжжя, 31 жовтня – 09 листопада 2011 р.). – Запоріжжя : Видавництво ПГА, 2011. – С. 131–134.

106. Яцишин А.Ю. Врахування структурованості даних при проектуванні гібридних сховищ даних / А.Ю. Яцишин // Сучасні комп'ютерні інформаційні технології : матеріали ІІ всеукраїнської школи-семінару молодих вчених і студентів АСІТ’2012. – Тернопіль : ТНЕУ, 2012. – С. 185–186.

107. Яцишин А.Ю. Проектування мультибазових сховищ даних на основі аналізу структурованості даних / А.Ю. Яцишин // Інтелектуальні системи прийняття рішень і проблеми обчислювального інтелекту : матеріали міжнародної наукової конференції. – Херсон : ХНТУ, 2012. – С. 244–245.

108. Бюджетний кодекс України від 08.07.10 № 2456-VI (Редакція станом на 01.01.2016) /[Електронний ресурс] – Режим доступу: http://zakon3.rada.gov.ua/laws/show/2456-17.

109. Яцишин А.Ю. Використання генетичного алгоритму з використанням адаптивності за генами для проектування мультибазового сховища даних / А.Ю. Яцишин // Інтелектуальні системи прийняття рішень і

проблеми обчислювального інтелекту : матеріали міжнародної наукової конференції. – Херсон : ХНТУ, 2014. – С. 349–351.

110. Яцишин А.Ю. Експериментальні дослідження двохфазного алгоритму проектування мультибазових сховищ даних / А.Ю. Яцишин // Сучасні комп'ютерні інформаційні технології : матеріали IV всеукраїнської школи-семінару молодих вчених і студентів АСІТ'2014. – Тернопіль : ТНЕУ, 2014. – С. 208–209.

111. Yatsyshyn A. Ensuring data warehouse quality of virtual university of ministry of finance of Ukraine / A. Yatsyshyn // Матеріали VIII міжнародної конференції "Стратегія якості в промисловості та освіті" (Варна, Болгарія, 8–15 червня 2012 р.) : у 3-х т. – Т. І. – Дніпетровськ – Варна, 2012. – С. 486–489.

112. Яцишин Ю.В. Роль віртуального університету у забезпеченні прозорості бюджетного процесу / Ю.В. Яцишин, А.Ю. Яцишин // Державний бюджет і бюджетна стратегія в умовах економічних реформ : у 4-х т. / ДННУ «Акад. фін. управління» ; за заг. ред. М.Я. Азарова. – К., 2011. – Т. 2. – С. 878–902 (ISBN 978-966-2380-24-8).

113. Яцишин Ю.В. Соціальна технологія «Прозорий бюджет» як інновація / Ю.В. Яцишин, А.Ю. Яцишин // Державний бюджет і бюджетна стратегія в умовах економічних реформ : у 4-х т. / ДННУ «Акад. фін. управління» ; за заг. ред. М.Я. Азарова. – К., 2011. – Т. 4. – С. 327–381 (ISBN 978-966-2380-26-2).

114. Яцишин А.Ю. Побудова сховища даних для Віртуального університету Міністерства фінансів України / А.Ю. Яцишин // Наука. Розвиток. Прогрес. – 2011. – С. 37–38.

115. Яцишин А.Ю. Особливості застосування генетичних алгоритмів при проектуванні гібридних сховищ даних з врахуванням джерел даних / А.Ю. Яцишин // Збірник доповідей учасників XV всеукраїнської науково-практичної конференції "Інноваційний потенціал української науки – XXI сторіччя" (Запоріжжя, 01 жовтня – 07 березня 2012 р.). – Запоріжжя : Видавництво ПГА, 2011. – С. 118–121.

116. Яцишин А.Ю. Проектування гібридного сховища даних з врахуванням структур джерел даних для системи електронного урядування / А.Ю. Яцишин // Інтелектуальні системи прийняття рішень і проблеми обчислювального інтелекту : матеріали міжнародної наукової конференції. – Херсон : ХНТУ, 2011.

ДОДАТКИ

Додаток А - програмний код модуля MDWService.

Додаток Б - таблиця опису класів модуля MDWService.

Додаток В - копії екранів роботи модуля MDWManager.

Додаток Г - діаграма компонентів СК МБСД.

Додаток Д - діаграма класів СК МБСД.

Додаток Е - діаграма діяльності СК МБСД.

Додаток Ж – діаграми статистики реєстрації та навчання слухачів у Віртуальному університеті Міністерства фінансів України.

Додаток З - перелік таблиць даних і розміщення їх у видах носіїв сховища.

Додаток К - розміщення областей сховища по його вузлах.

Додаток Л – акти про впровадження результатів наукових досліджень

Додаток М – свідоцтва про реєстрацію авторського права на твір

Додаток А. Програмний код модуля MDWService.

Процедура формування попередньої популяції модифікованого генетичного алгоритму з супровідними функціями до неї.

```
public void GetACs(ArrayList SortedACs,int bestcnt,int potentcnt,out
ArrayList BestACs,out ArrayList PotentACs)
        {
           BestACs = new ArrayList(bestcnt);
           for (int i = 0; i < bestcnt; i++)
              try
              {
                 BestACs.Add(SortedACs[i]);
              }
              catch
              { }
           PotentACs = new ArrayList(potentcnt);
           for (int i = 0; i < potentcnt; i++)
              try
              {
                 PotentACs.Add(SortedACs[bestcnt+i]);
              }
              catch
              { }
        }

        public ArrayList GetNearPoints(ArrayList ACs,int maxdist,int pointnum)
        {
        //    Console.WriteLine("Entered GetNearPoints");
           ArrayList NPs = new ArrayList(ACs.Count * pointnum);
           for (int i = 0; i < ACs.Count;i++)
              for (int j=0;j<pointnum;j++)
                 NPs.Add(GetNearPoint((Genome)ACs[i],maxdist));
         //   Console.WriteLine("Left GetNearPoints");
           return NPs;
        }
        public Genome GetNearPoint(Genome g,int maxdist)
        {
           //Console.WriteLine("Entered GetNearPoint");
```

```
    int dist = m_ManagedRandom.Next(1, maxdist);
    //int dist = maxdist;
    Genome g2 = new Genome(g.Length);
    /*   for (int i = g.Length - 1; i > g.Length - 1 - dist; i--)
           g2.m_genes[i] = 1 - g.m_genes[i];
       for (int i = g.Length - 1 - dist; i >= 0; i--)
           g2.m_genes[i] = g.m_genes[i];*/
    List<int> poss = new List<int>();
    for (int i = 0; i < dist; i++)
    {
     //  Console.WriteLine(i);
       int j = m_ManagedRandom.Next(2, g.m_genes.Length - 1);
       /*while (poss.Contains(j))
           j = m_ManagedRandom.Next(2, g.m_genes.Length - 1);*/
       poss.Add(j);
    }


    for (int i = 0; i < g.Length ; i++)
       g2.m_genes[i] =  poss.Contains(i) ? 1 - g.m_genes[i] : g.m_genes[i];
   // Console.WriteLine("Left GetNearPoint");
    return g2;
}

public ArrayList GetNewACs(Chromosome Markup,int count)
{
    ArrayList m_thisGeneration = new ArrayList();
    for (int i = 0; i < count - 1; i++)
    {
       Genome g = new Genome(Markup, true);
       m_thisGeneration.Add(g);
    }
    return m_thisGeneration;
}


  ArrayList Best; ArrayList Potent;
  GetACs(m_thisGeneration, bestcnt, potcnt, out Best, out Potent);
     ShowGeneration("Best area centers", Best,abcenabled,gaenabled);
```

```
                ShowGeneration("Potent area centers", Potent, abcenabled, gaenabled);
            ArrayList InBest = GetNearPoints(Best, bestdist, bestnew);
                ShowGeneration("In best areas", InBest, abcenabled, gaenabled);
            ArrayList InPotent = GetNearPoints(Potent, potdist, potnew);
              ShowGeneration("In potent areas", InPotent, abcenabled, gaenabled);
            ArrayList NewPoints = GetNewACs(MarkupChromosome, scatter);
              ShowGeneration("Scatter search", NewPoints, abcenabled, gaenabled);
            Genome g = new Genome(MarkupChromosome, true);
            int gj = 0;
            m_thisGeneration = new ArrayList();
            m_thisGeneration.AddRange(Best);
            m_thisGeneration.AddRange(InBest);
            m_thisGeneration.AddRange(InPotent);
            m_thisGeneration.AddRange(Potent);
            m_thisGeneration.AddRange(NewPoints);
            // Rearrange
            CheckPopulationForFeasibleness();
            CheckPopulationForDuplication();
            RankPopulationByGenes();
            GenomeGenesComparer.sign = RankPopulationByFitness(starter, 0,
m_generationSize, out time2, MarkupChromosome) == 0 ? 1 : -1;
                if (m_thisGeneration.Count > m_populationSize)
                    m_thisGeneration.RemoveRange(m_populationSize,
m_thisGeneration.Count - m_populationSize);
                ShowGeneration("ABC, cegc=" + cegc + "/" + regc + " ; T=" + time2,
m_thisGeneration,abcenabled,gaenabled);
            //   Console.ReadLine();
            time_abc += time2;
                if (Math.Abs(Math.Abs(fit) -
Math.Abs(((Genome)m_thisGeneration[0]).Fitness)) < 1)
```

```
                cegc++;
            else cegc = 0;
        fit = ((Genome)m_thisGeneration[0]).Fitness;
    }
    time += time_abc;
    Console.WriteLine(string.Format("Winner by ABC : {0}, time spent:{1}",
fit, time_abc));
    initGeneration.AddRange(m_thisGeneration);
                                    #endregion
```

Додаток Б. Таблиця опису класів модуля MDWService
[Розроблено автором]

Клас	Батьківський клас	Опис класу
Класи, які підтримують структуру сховища		
LADataStore	- (object)	Носій даних (НД)
LADataWarehouse	LADataStore	СД
LAMDW	LADataWarehouse	МБСД
LAОбласть	- (object)	Область МБСД
LABaseTable	- (object)	Базова таблиця
LATable	LABaseTable	Таблиця МБСД
LADatastoreTable	LABaseTable	Таблиця носія даних
LAView	- (object)	Подання МБСД
LADataStoreView	- (object)	Подання носія даних
LAColumn	- (object)	Стовпчик (таблиці)
LAAttribute	- (object)	Атрибут МБСД
Expression	- (object)	Вираз
TableExpression	- (object)	Вираз над таблицями
ValuesExpression	- (object)	Вираз над значеннями
Класи, що підтримують обчислення виразів		
Condition	Expression	Умова вибору
FilterCondition	Condition	Умова фільтрування
JoinCondition	Condition	Умова з’єднання
BoolCondition	Condition	Умова-операція
LADataSource	- (object)	Джерело даних [таблиці]
LADirectSource	LADataSource	Пряме джерело
LASplitSource	LADataSource	Джерело над розділеними даними
LACalculatedSource	LADataSource	Джерело з розрахованих даних
LAComplexSource	LADataSource	Складне джерело

Operator	- (object)	Оператор
TableOperator	Operator	Оператор над таблицями
ConditionOperator	Operator	Оператор-умова
ValueOperator	Operator	Оператор над значеннями
Класи, що підтримують запити до сховища		
Query	- (object)	Запит
QueryClasses	- (object)	Класи запитів
QueryInfo	- (object)	Інформація про запит
QueryStatistics	- (object)	Статистика виконання запитів
Класи, що підтримують проектування сховища		
DatabaseWorker	- (object)	Клас для роботи з БД
CubeWorker	- (object)	Клас для роботи з кубами даних
Класи, що підтримують оптимізацію сховища		
GeneticAlgorithm	- (object)	Генетичний алгоритм
HDWAlgorithm	GeneticAlgorithm	Пропонований алгоритм
AlgorithmStatus	- (object)	Стан алгоритму
Genome	- (object)	Геном
Chromosome	- (object)	Хромосома
Класи, що підтримують носії даних		
LADataStore	- (object)	Носій даних (НД)
LAMSSQLDS	LADataStore	НД типу Microsoft SQL Server Database Services
LAMSSQLAS	LADataStore	НД типу Microsoft SQL Server Analysis Services
LANXDB	LADataStore	НД типу BaseX
LAMongoDB	LADataStore	НД типу MongoDB

Додаток В. Копії екранів роботи модуля MDWManager

Після запуску модуля MDWManager отримуємо таке вікно, де ми можемо бачити джерела, носії даних сховища та його структуру. Також можна проводити всі операції, необхідні для проектування, а саме – визначення джерел та носіїв, аналіз сховища, що будується та інтеграція даних.

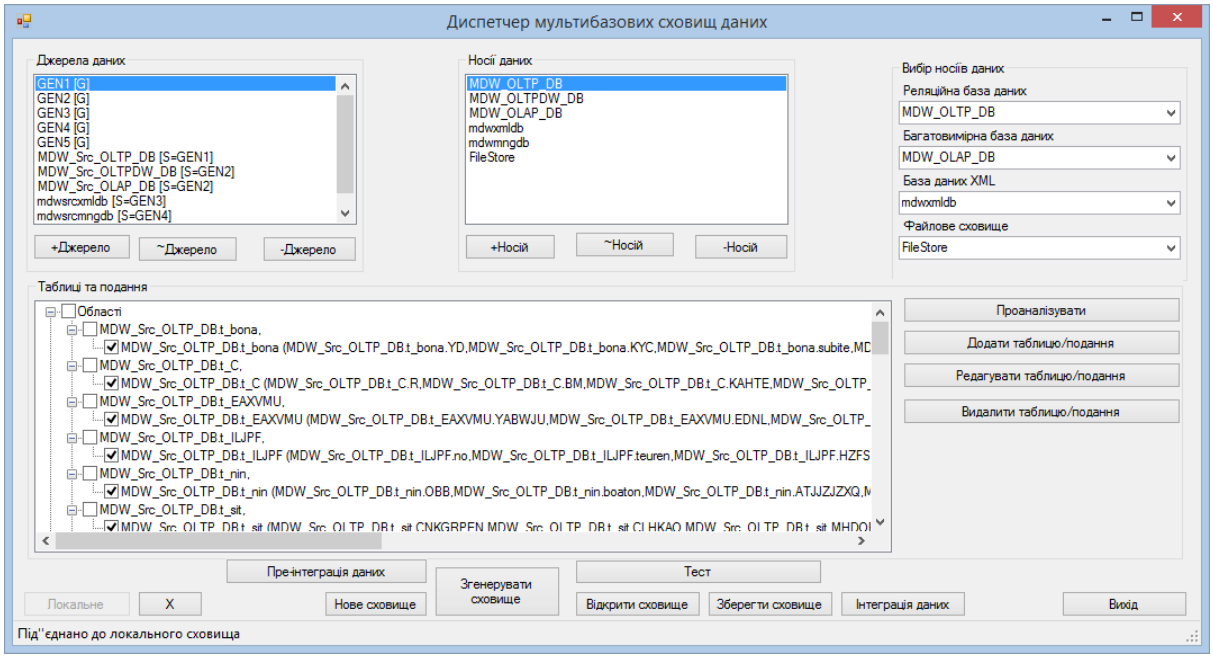


Рис. 5.1. Знімок екрана основного вікна.

Для визначення джерел та носіїв даних передбачено наступну екранну форму:

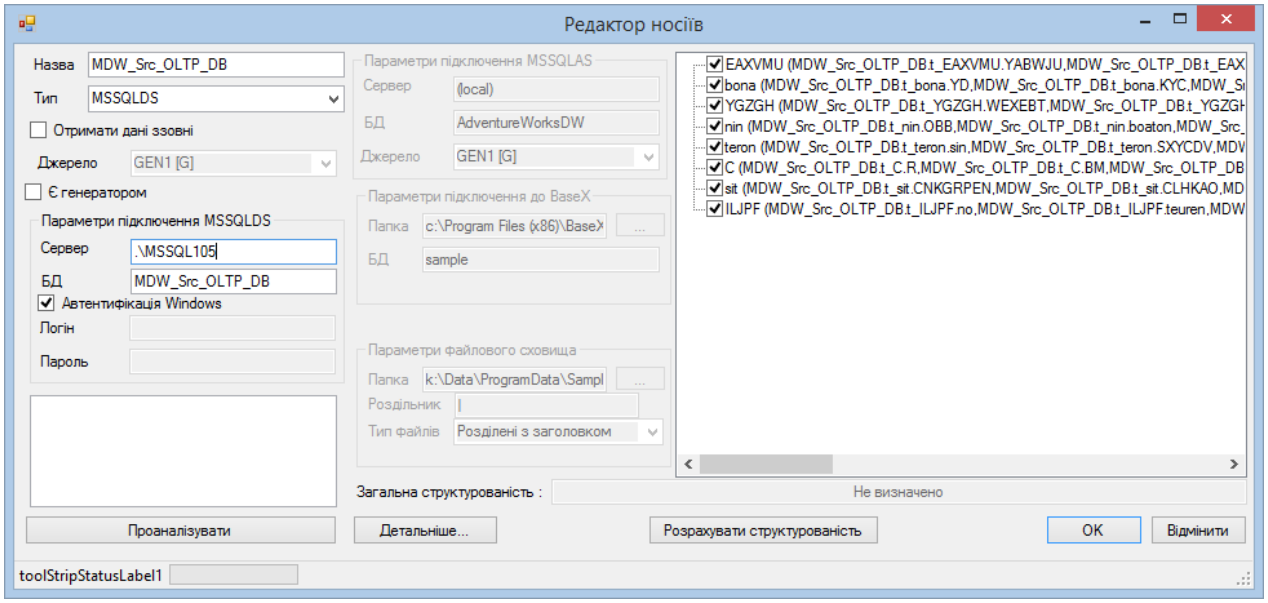


Рис. 5.2. Знімок редактора носіїв даних.

Для зміни сутностей даних передбачено наступний редактор

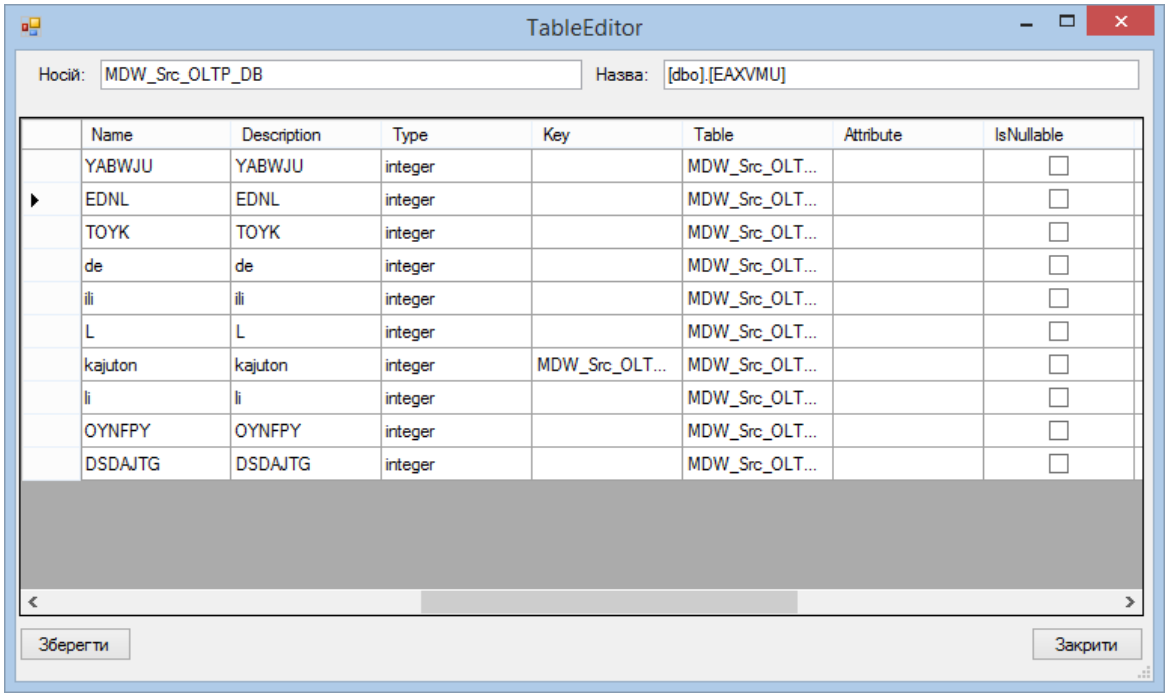
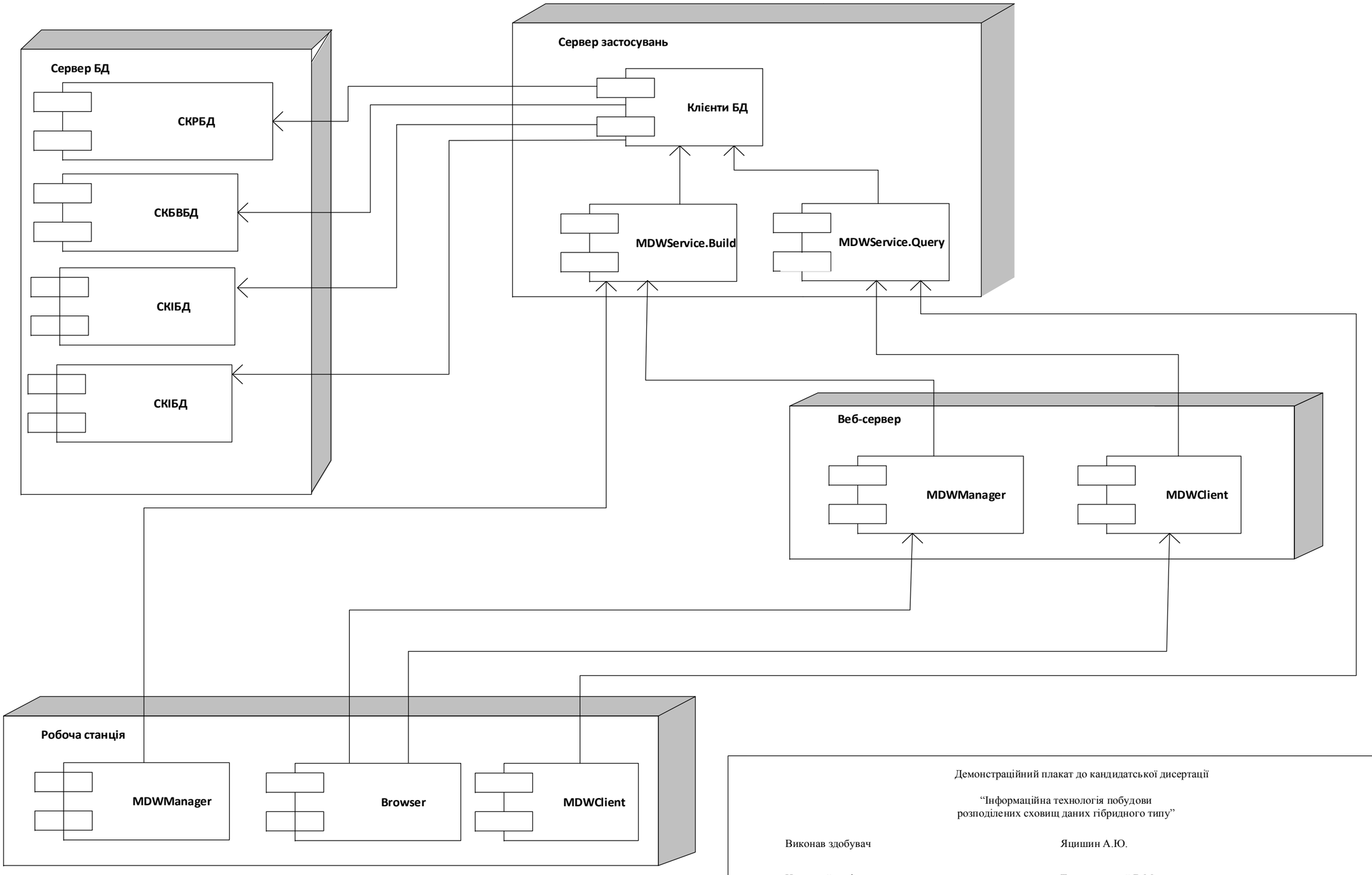


Рис. 5.3. Знімок екрана роботи з сутностями.

Додаток Г.

ДІАГРАМА КОМПОНЕНТІВ СКМБСД



Демонстраційний плакат до кандидатської дисертації

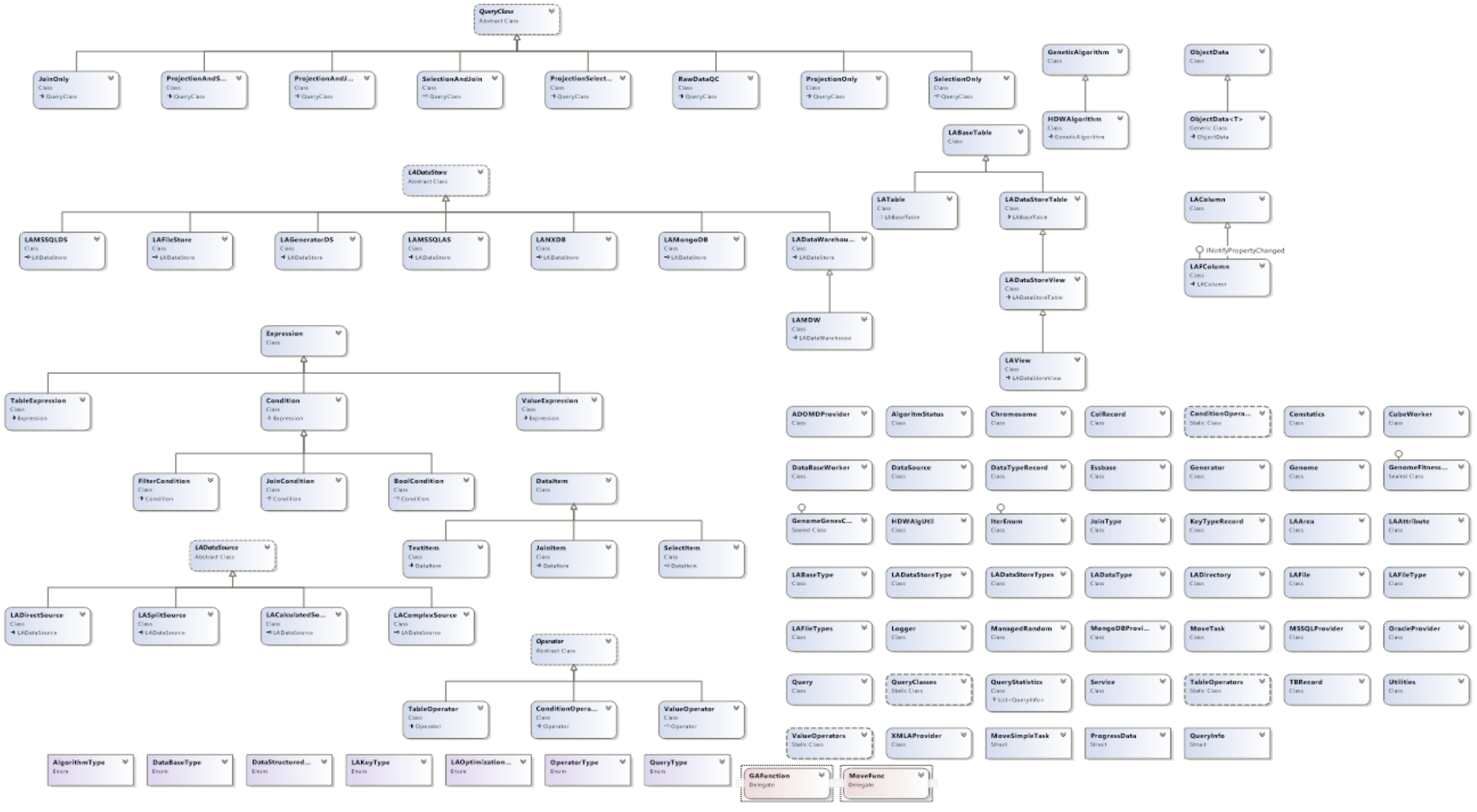
“Інформаційна технологія побудови
розподілених сховищ даних гібридного типу”

Виконав здобувач Яцишин А.Ю.

Науковий керівник Томашевський В.М.

Додаток Д.

ДІАГРАМА КЛАСІВ СКМБСД



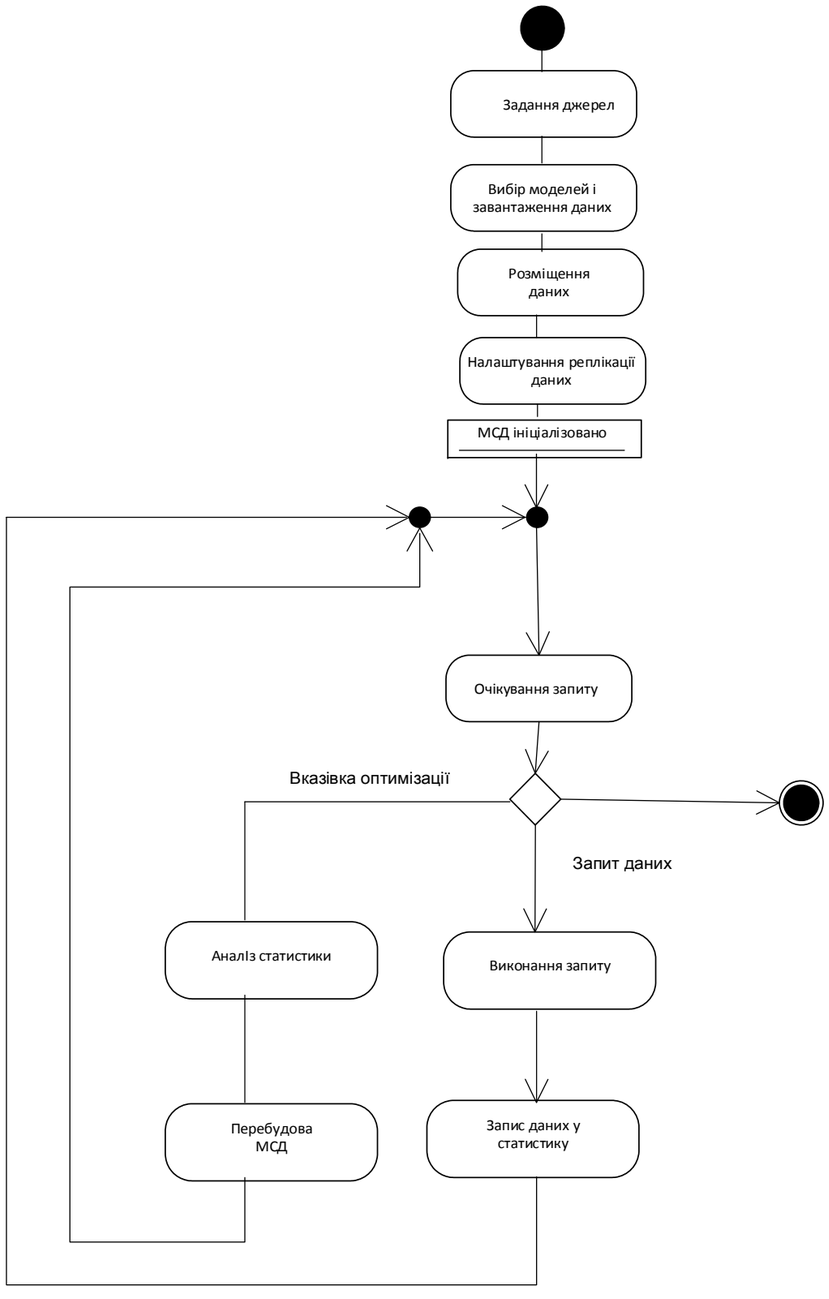
Демонстраційний плакат до кандидатської дисертації

“Інформаційна технологія побудови
розподілених сховищ даних гібридного типу”

Виконав здобувач	Яцишин А.Ю.
Науковий керівник	Томашевський В.М.

Додаток Е.

ДІАГРАМА ДІЯЛЬНОСТІ СХОВИЩА ДАНИХ



Демонстраційний плакат до кандидатської дисертації

“Інформаційна технологія побудови
розподілених сховищ даних гібридного типу”

Виконав здобувач Яцишин А.Ю.

Науковий керівник Томашевський В.М.

Додаток Ж. Діаграми статистики реєстрації та навчання слухачів у Віртуальному університеті Міністерства фінансів України.

Рис. 5.4. Діаграма реєстрації та навчання користувачів у Віртуальному університеті Міністерства фінансів України (кількість користачів, рік). [Розроблено автором]

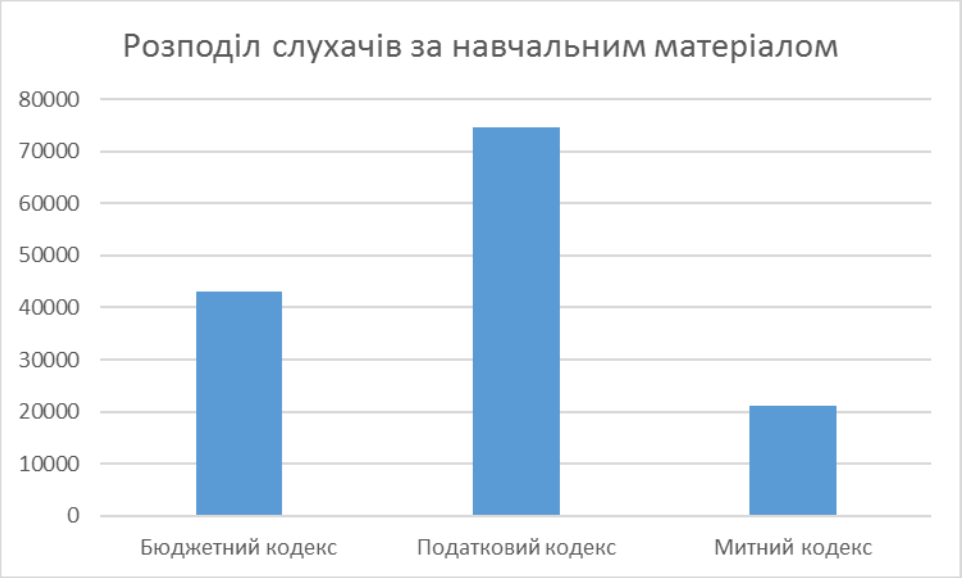


Рис. 5.5. Діаграма розподілу слухачів Віртуального університету Міністерства фінансів України за навчальним матеріалом (кількість слухачів). [Розроблено автором]

Додаток З. Перелік таблиць даних і розміщення їх у видах носіїв сховища. [Розроблено автором]

№ обл.	Джерело/область/таблиця	Структурова-ність	Розміщення
Джерело даних MFUVUDB		**0.154224**	**-**
128	**Область даних**	**1**	**RDB**
128	MFUVUDB.t_aspnet_Roles	0	RDB
128	MFUVUDB.t_aspnet_Applications	0	RDB
136	**Область даних**	**0.697441**	**RDB**
136	MFUVUDB.t_mfuvu_certs	0.857143	RDB,NOSQL
136	MFUVUDB.t_mfuvu_courses	1	RDB
136	MFUVUDB.t_mfuvu_Users	0.525039	RDB,NOSQL
136	MFUVUDB.t_mfuvu_Educ_types	1	RDB
136	MFUVUDB.t_mfuvu_Settlements	1	RDB
136	MFUVUDB.t_mfuvu_Specialities	1	RDB
136	MFUVUDB.t_mfuvu_Qualifications	1	RDB
136	MFUVUDB.t_mfuvu_WorkAreas	1	RDB
136	MFUVUDB.t_mfuvu_Positions	1	RDB
136	MFUVUDB.t_mfuvu_Stis	1	RDB
165	MFUVUDB.t_vw_aspnet_MembershipUsers	0.261539	RDB,NOSQL
182	MFUVUDB.t_vw_mfuvu_list_sti_correspond	0.666942	RDB,NOSQL
191	MFUVUDB.t_vw_mfuvu_Users	0.401167	RDB,NOSQL
Джерело даних TransBudgDB		**0.628414**	**-**
192	**Область даних**	0.941605	RDB
192	TransBudgDB.t_VXXDDMMYY	0.961365	RDB
192	TransBudgDB.t_D_BUDG_LOCAL_DET	0.888913	RDB
193	TransBudgDB.t_D_ECON_CRED	0.9	RDB,NOSQL
194	TransBudgDB.t_D_FIN	0.916667	RDB,NOSQL
196	**Область даних**	0.681536	RDB
196	TransBudgDB.t_DMYYMM	0.533333	RDB,NOSQL
196	TransBudgDB.t_D_INC_DET	0.903839	RDB
197	**Область даних**	1	RDB
197	TransBudgDB.t_district	1	RDB
197	TransBudgDB.t_obl_region	1	RDB
198	**Область даних**	1	RDB
198	TransBudgDB.t_INCDET	1	RDB
198	TransBudgDB.t_DOV_TAX	1	RDB
221	TransBudgDB.t_VW_EXPENSES_DET	0.5	RDB,NOSQL
222	TransBudgDB.t_VW_EXPENSES_LVL1	0.5	RDB,NOSQL
223	TransBudgDB.t_VW_EXPENSES_LVL3	0.5	RDB,NOSQL
233	TransBudgDB.t_VW_INCOME_DET	0.5	RDB,NOSQL

234	TransBudgDB.t_VW_INCOME_LVL1	0.5	RDB,NOSQL
235	TransBudgDB.t_VW_INCOME_LVL3	0.5	RDB,NOSQL
Область даних FUPortalDB		**0.325218**	**-**
66	**Область даних**	0.875	RDB
66	FUPortalDB.t_Users	1	RDB
66	FUPortalDB.t_Applications	0.666667	RDB,NOSQL
67	**Область даних**	0.399563	RDB
67	FUPortalDB.t_IsAuthCorrespond	1	RDB
67	FUPortalDB.t_Articles	0.375858	RDB,NOSQL
67	FUPortalDB.t_Sections	1	RDB
67	FUPortalDB.t_People	0.227109	RDB,NOSQL
67	FUPortalDB.t_Conclusions	1	RDB
67	FUPortalDB.t_ArticleStates	1	RDB
68	FUPortalDB.t_aspnet_Applications	0.666667	RDB,NOSQL
69	FUPortalDB.t_Authors	0.281627	RDB,NOSQL
70	FUPortalDB.t_Issues	0.605556	RDB,NOSQL
75	**Область даних**	1	RDB
75	FUPortalDB.t_UsersOpenAuthAccounts	-1	RDB
75	FUPortalDB.t_UsersOpenAuthData	-1	RDB
76	FUPortalDB.t_vw_Art_Auth	0.385276	RDB,NOSQL
77	FUPortalDB.t_vw_Art_Auth_short	0.886225	RDB,NOSQL
78	FUPortalDB.t_vw_ArticlesExtended	0.386946	RDB,NOSQL
Область даних AdventureWorks		**0.959203**	**-**
1	**Область даних**	1	RDB
1	AdventureWorks.t_SalesTaxRate	1	RDB
1	AdventureWorks.t_StateProvince	1	RDB
1	AdventureWorks.t_CountryRegion	1	RDB
1	AdventureWorks.t_SalesTerritory	0	RDB
4	**Область даних**	0.822177	RDB
4	AdventureWorks.t_WorkOrder	0.901004	RDB
4	AdventureWorks.t_Product	0.716587	RDB,NOSQL
4	AdventureWorks.t_UnitMeasure	1	RDB
4	AdventureWorks.t_ProductSubcategory	1	RDB
4	AdventureWorks.t_ProductCategory	1	RDB
4	AdventureWorks.t_ProductModel	0.686198	RDB,NOSQL
4	AdventureWorks.t_ScrapReason	1	RDB
12	**Область даних**	1	RDB
12	AdventureWorks.t_CurrencyRate	1	RDB
12	AdventureWorks.t_Currency	1	RDB
15	**Область даних**	0.978086	RDB
15	AdventureWorks.t_PurchaseOrderHeader	1	RDB
15	AdventureWorks.t_Employee	1	RDB

15	AdventureWorks.t_Vendor	0.882212	RDB
15	AdventureWorks.t_ShipMethod	1	RDB
21	**Область даних**	1	RDB
21	AdventureWorks.t_WorkOrderRouting	1	RDB
21	AdventureWorks.t_Location	1	RDB
35	**Область даних**	0.96681	RDB
35	AdventureWorks.t_SalesOrderDetail	0.954741	RDB
35	AdventureWorks.t_SpecialOfferProduct	1	RDB
45	AdventureWorks.t_vAdditionalContactInfo	0.535294	RDB,NOSQL
46	AdventureWorks.t_vEmployee	0.778927	RDB,NOSQL
48	AdventureWorks.t_vEmployeeDepartmentHistory	0.728194	RDB,NOSQL
49	AdventureWorks.t_vIndividualCustomer	0.811049	RDB,NOSQL
56	AdventureWorks.t_vProductModelInstructions	0.884108	RDB,NOSQL
57	AdventureWorks.t_vSalesPerson	0.842246	RDB,NOSQL
61	AdventureWorks.t_vStoreWithContacts	0.88568	RDB,NOSQL
64	AdventureWorks.t_vVendorWithContacts	0.879274	RDB,NOSQL

Додаток К. Розміщення областей сховища по його вузлах

[Розроблено автором]

№ областей	№ вузла	Тип вузла
1..235	1	Центральний
1..10	2	Регіональний
12..20	3	Регіональний
21..30	4	Регіональний
31..40	5	Регіональний
41..50	6	Регіональний
51..60	7	Регіональний
61..70	8	Регіональний
71..80	9	Регіональний
81..90	10	Регіональний
91..100	11	Регіональний
101..110	12	Регіональний
111..120	13	Регіональний
121..130	14	Регіональний
131..140	15	Регіональний
141..150	16	Регіональний
151..170	17	Регіональний
171..190	18	Регіональний
191..210	19	Регіональний
211..234	20	Регіональний
-	21	Додатковий*
-	22	Додатковий*
-	23	Додатковий*
-	24	Додатковий*
-	25	Додатковий*

Додаток Л. Акти про впровадження результатів наукових досліджень

ЗАТВЕРДЖУЮ **Заступник Міністра фінансів України - керівник апарату**

Ю.І. Шевченко

«27» грудня **2011р.**

АКТ

про впровадження результатів Науково-дослідної роботи на тему: «Модернізація управління державними фінансами в контексті адміністративної реформи» (Згідно договору 15010-02/82 від 20 травня 2011р., державний реєстраційний № РК 0111U007012, науковий керівник к.т.н., доцент А.О. Білощицький).

Комісія у складі:

Голова – Директор департаменту забезпечення діяльності Міністра (патронатна служба) Міністерства фінансів України О.С. Яроцький.

Члени комісії Романов І.В;

Онищенко М.П.

- цим Актом засвідчує, що результати Науково-дослідної роботи на тему: «Модернізація управління державними фінансами в контексті адміністративної реформи» (Згідно договору 15010-02/82 від 20 травня 2011р., державний реєстраційний № 0111U007012, науковий керівник к.т.н., доцент А.О. Білощицький) використані співробітниками департаменту забезпечення діяльності Міністра (патронатна служба) Міністерства фінансів України, для визначення теоретико-методологічних та правових засад реалізації адміністративної реформи у Міністерстві фінансів України; встановлення необхідності і шляху удосконалення організаційної структури Міністерства фінансів в ході адміністративної реформи системи в Україні, розкриття проблеми та шляхи оптимізації організаційної структури Міністерства фінансів. Дослідження проблеми управління реформуванням організаційних структур в Україні та шляхи його удосконалення. Встановлення особливості формування та виконання функцій структурними підрозділами Міністерства фінансів в сучасних умовах. Здійснення функціонально-рольового моделювання організаційної структури, проведення ділової гри з працівниками Міністерства фінансів, узагальнення її результатів, розробки відповідних аналітичних матеріалів та рекомендацій. Розкриття економіко-правових аспектів реформування організаційної структури Міністерства фінансів. Визначення розривів та дублювання управлінських функцій у центральному апараті Міністерства фінансів України, надання науково обґрунтованих рекомендацій до їх усунення, підготовки навчально-методичних матеріалів до проведення ділових ігор з функціонально-рольового моделювання.

Голова комісії Директор департаменту забезпечення діяльності Міністра (патронатна служба) Міністерства фінансів України _____ **Яроцький О.С.**
(підпис)

Члени комісії

заступник директора департаменту _____ / Романов І.В.
(посада) (підпис) (прізвище та ініціали)

помічник Міністра _____ Онищенко М.П.
(посада) (підпис) (прізвище та ініціали)

"27" грудня 2011р.

Додаток А

" ЗАТВЕРДЖУЮ "

Проректор з наукової роботи та
міжнародних зв`язків Київського
національного університету
будівництва і архітектури,
д.т.н., професор

Плоский В. О.
« 30 » 11 2015р.

АКТ

про впровадження результатів наукових досліджень дисертаційної роботи Яцишина Андрія Юрійовича на тему «Інформаційна технологія побудови розподілених сховищ даних гібридного типу» за спеціальністю 05.13.06 – «Інформаційні технології» при виконанні науково-дослідної роботи на тему: «Модернізація управління державними фінансами в контексті адміністративної реформи» (згідно договору 15010-02/82 від 20 травня 2011 р, державний реєстраційний № 0111U007012, науковий керівник к.т.н., доцент А.О.Білощицький)

Комісія у складі :

Голова комісії - *завідувач кафедри управління проектами, доктор технічних наук, професор Бушуєв С.Д.*

Члени комісії: - *завідувач кафедри інформаційних технологій, доктор технічних наук, професор Білощицький А.О.,*

- *завідувач кафедри інформаційних технологій проектування та прикладної математики, доктор технічних наук, професор Міхайленко В.М.*

встановила, що при виконанні науково-дослідної роботи на тему: «Модернізація управління державними фінансами в контексті адміністративної реформи» (згідно договору 15010-02/82 від 20 травня 2011 р, державний реєстраційний №0111U007012, науковий керівник к.т.н., доцент А.О. Білощицький) при розробленні інформаційної системи «Віртуальний університет Міністерства фінансів України» було застосовано інформаційну технологію побудови розподілених сховищ даних гібридного типу.

При виконанні цієї роботи були використані наступні наукові та практичні результати дисертаційної роботи А.Ю.Яцишина:

1. на основі проведеного аналізу існуючих підходів та методів до побудови моделей сховищ даних для розподілених сховищ даних гібридного типу було розроблено:

– метод побудови з мінімальною сукупною вартістю збереження та обробки даних,

– математичну модель опису різних рівнів;

– методику фізичного та логічного розподілу даних.

2. досліджено вплив параметрів модифікованого генетичного алгоритму на його ефективність і збіжність та визначено значення параметрів фаз формування початкової популяції та генетичного пошуку;

3. розроблено інформаційну технологію побудови розподілених сховищ даних гібридного типу на базі комплексу математичних моделей і методів аналізу та оцінювання процесів опрацювання даних у сховищах зазначеного типу і оптимізації їх структури;

що дозволило підвищити швидкодію виконання запитів та знизити сукупну вартість системи збереження даних.

Рішення комісії:

Результати дисертаційних досліджень А.Ю.Яцишина використані при розробці інформаційної системи «Віртуальний університет Міністерства фінансів України», а також використовуються у навчальному процесі Київського національного університету будівництва і архітектури з 2012 року.

Голова комісії:

Зав. каф. управління проектами — *Бушуєв С.Д.*
(посада) (підпис) (прізвище, ініціали)

Члени комісії:

Зав. каф. інформаційних технологій — *Білощицький А.О.*
(посада) (підпис) (прізвище, ініціали)

Зав. каф. інформаційних технологій проектування та прикладної математики — *Міхайленко В.М.*
(посада) (підпис) (прізвище, ініціали)

ЗАТВЕРДЖУЮ

Заступник Міністра фінансів України
- керівник апарату

_____ **Ю.І. Шевченко**

«27» грудня **2011р.**

АКТ

про впровадження результатів науково-дослідної роботи на тему: «Моделювання та прогнозування розвитку фінансової системи України» (Згідно договору 15010-02/83 від 20 травня 2011 р, державний реєстраційний №0111U00701, науковий керівник к.т.н., доцент А.О. Білощицький).

Комісія у складі:

Голова – Директор департаменту забезпечення діяльності Міністра (патронатна служба) Міністерства фінансів України О.С. Яроцький.

Члени комісії Романов І.В;
Онищенко М.П.

цим Актом засвідчує, що результати Науково-дослідної роботи на тему «Моделювання та прогнозування розвитку фінансової системи України» (Згідно договору 15010-02/83 від 20 травня 2011 р, державний реєстраційний №0111U007014, науковий керівник к.т.н., доцент А.О. Білощицький) використані співробітниками департаменту забезпечення діяльності Міністра (патронатна служба) Міністерства фінансів України при розробленні та реалізації інноваційних механізмів управління знаннями на основі інтуїції та при навчанні фахівців фінансової сфери у Віртуальному університеті Міністерства фінансів України, при визначенні пріоритетних напрямів функціонування Віртуального університету та визначенні впливу реалізованої інформаційної технології системи «Віртуальний університет» на ефективність навчання.

Голова комісії Директор департаменту
забезпечення діяльності Міністра
(патронатна служба)
Міністерства фінансів України _____ **Яроцький О.С.**
(підпис)

Члени комісії

заступник директора _____ Романов І.В
(посада) (підпис) (прізвище та ініціали)

помічник Міністра _____ Онищенко М.
(посада) (підпис) (прізвище та ініціали)

"27" грудня 2011р.

Додаток А

" ЗАТВЕРДЖУЮ "

Проректор з наукової роботи та міжнародних зв`язків Київського національного університету будівництва і архітектури, д.т.н., професор

_____ Плоский В. О.

«30» 11 2015р.

АКТ

про впровадження результатів наукових досліджень дисертаційної роботи Яцишина Андрія Юрійовича на тему «Інформаційна технологія побудови розподілених сховищ даних гібридного типу» за спеціальністю 05.13.06 – «Інформаційні технології» при виконанні науково-дослідної роботи на тему: «Моделювання та прогнозування розвитку фінансової системи України» (згідно договору 15010-02/83 від 20 травня 2011 р, державний реєстраційний №0111U007014, науковий керівник к.т.н., доцент А.О. Білощицький)

Комісія у складі :

Голова комісії - *завідувач кафедри управління проектами, доктор технічних наук, професор Бушуєв С.Д.*

Члени комісії: - *завідувач кафедри інформаційних технологій, доктор технічних наук, професор Білощицький А.О.,*

- *завідувач кафедри інформаційних технологій проектування та прикладної математики, доктор технічних наук, професор Міхайленко В.М.*

встановила, що при виконанні науково-дослідної роботи на тему: «Моделювання та прогнозування розвитку фінансової системи України» (згідно договору 15010-02/83 від 20 травня 2011 р, державний реєстраційний №0111U007014, науковий керівник к.т.н., доцент А.О. Білощицький) при розробленні інформаційно-аналітичної системи «Прозорий бюджет» було застосовано інформаційну технологію побудови розподілених сховищ даних гібридного типу.

При виконанні цієї роботи були використані наступні наукові та практичні результати дисертаційної роботи А.Ю.Яцишина:

1. на основі проведеного аналізу існуючих підходів та методів до побудови моделей сховищ даних для розподілених сховищ даних гібридного типу було розроблено:

– метод побудови з мінімальною сукупною вартістю збереження та обробки даних;

– математичну модель опису різних рівнів;

– методику фізичного та логічного розподілу даних.

2. досліджено вплив параметрів модифікованого генетичного алгоритму на його ефективність і збіжність та визначено значення параметрів фаз формування початкової популяції та генетичного пошуку;

3. розроблено інформаційну технологію побудови розподілених сховищ даних гібридного типу на базі комплексу математичних моделей і методів аналізу та оцінювання процесів опрацювання даних у сховищах зазначеного типу і оптимізації їх структури;

що дозволило підвищити швидкодію виконання запитів та знизити сукупну вартість системи збереження даних.

Рішення комісії:

Результати дисертаційних досліджень А.Ю.Яцишина використані при розробці інформаційно-аналітичної системи «Прозорий бюджет» для Міністерства фінансів України, а також використовуються у навчальному процесі Київського національного університету будівництва і архітектури з 2012 року.

Голова комісії:

Зав. каф. управління проектами _____ *Бушуєв С.Д.*
(посада) (підпис) (прізвище, ініціали)

Члени комісії:

Зав. каф. інформаційних технологій _____ *Білощицький А.О.*
(посада) (підпис) (прізвище, ініціали)

Зав. каф. інформаційних технологій
проектування та прикладної математики _____ *Міхайленко В.М.*
(посада) (підпис) (прізвище, ініціали)

ЗАТВЕРДЖУЮ

Президент Державної навчально-наукової установи "Академія фінансового управління", чл.- кор. НАН України, д.е.н., професор

Єфименко Т.І.

«23» 12 2014 р.

АКТ

про впровадження результатів наукових досліджень дисертаційної роботи Яцишина Андрія Юрійовича на тему «Інформаційна технологія побудови розподілених сховищ даних гібридного типу» за спеціальністю 05.13.06 «Інформаційні технології»

Комісія у складі:

Голова – директор Науково-дослідного фінансового інституту ДННУ «Академії фінансового управління», к.е.н., доцент Гасанов С.С.

Члени комісії:

директор Інституту післядипломної освіти ДННУ «АФУ», к.е.н., доцент Котляревський Я.В.;

директор інформаційно-аналітичного департаменту Міністерства фінансів України Дубовенко В.В.

цим Актом засвідчує, що результати наукових досліджень дисертаційної роботи Яцишина Андрія Юрійовича, а саме розроблена ним інформаційна технологія побудови розподілених сховищ даних гібридного типу на базі комплексу математичних моделей і методів аналізу та оцінювання процесів опрацювання даних у сховищах із гетерогенними даними, які зберігаються розподілено, використана при розробці програмних продуктів ДННУ «Академії фінансового управління» Міністерства фінансів України - Порталу Інституту післядипломної освіти та Порталу журналу «Фінанси України», що дозволило підвищити швидкодію виконання запитів та знизити сукупну вартість збереження та обробки даних у сховищі.

Голова комісії –директор Науково-дослідного фінансового інституту ДННУ «Академії фінансового управління» Гасанов С.С.

Члени комісії:

Директор Інституту післядипломної освіти ДННУ «Академії фінансового управління» Котляревський Я.В.

Директор інформаційно-аналітичного департаменту Міністерства фінансів України Дубовенко В.В.

Додаток М. Свідоцтва про реєстрацію авторського права на твір

УКРАЇНА

ДЕРЖАВНА СЛУЖБА ІНТЕЛЕКТУАЛЬНОЇ ВЛАСНОСТІ УКРАЇНИ

СВІДОЦТВО

про реєстрацію авторського права на твір

№ 43464

Комп'ютерна програма "Віртуальний університет Міністерства фінансів України"

(вид, назва твору)

Автор(и) Яцишин Юрій Володимирович, Яцишин Андрій Юрійович

(повне ім'я, псевдонім (за наявності))

Дата реєстрації 24.04.2012

Державна служба інтелектуальної власності України
Ідентифікаційний код 37552556
М.П.

Голова Державної служби
інтелектуальної
власності України
М.В. Паладій

УКРАЇНА · UKRAINE · УКРАЇНА · UKRAINE · УКРАЇНА · UKRAINE · УКРАЇНА · UKRAINE · УКРАЇНА

УКРАЇНА

ДЕРЖАВНА СЛУЖБА ІНТЕЛЕКТУАЛЬНОЇ ВЛАСНОСТІ УКРАЇНИ

СВІДОЦТВО

про реєстрацію авторського права на твір

№ 43465

Комп'ютерна програма "Інформаційно-аналітична система "Прозорий бюджет" ("ІАС "Прозорий бюджет")

(вид, назва твору)

Автор(и) Яцишин Юрій Володимирович, Яцишин Андрій Юрійович

(повне ім'я, псевдонім (за наявності))

Дата реєстрації 24.04.2012

Державна служба інтелектуальної власності України
Ідентифікаційний код 37552556
М.П.

Голова Державної служби
інтелектуальної
власності України
М.В. Паладій

УКРАЇНА

ДЕРЖАВНА СЛУЖБА ІНТЕЛЕКТУАЛЬНОЇ ВЛАСНОСТІ УКРАЇНИ

СВІДОЦТВО

про реєстрацію авторського права на твір

№ 55790

Комп'ютерна програма "Портал періодичного наукового видання" ("Портал ПНВ")

(вид, назва твору)

Автор(и) Яцишин Юрій Володимирович, Яцишин Андрій Юрійович

(повне ім'я, псевдонім (за наявності))

Дата реєстрації 28.07.2014

Державна служба інтелектуальної власності України Ідентифікаційний код 37552556 М.П.

Голова Державної служби інтелектуальної власності України
М.В. Ковіня