УДК 004.6

А.Ю. Яцишин
Національний технічний університет України "Київський політехнічний інститут"

ПІДХОДИ ТА АЛГОРИТМИ ПРОЕКТУВАННЯ ГІБРИДНИХ СХОВИЩ ДАНИХ

Проаналізовано відомі підходи та алгоритми до проектування гібридних сховищ даних. Запропоновано концепцію узагальненого гібридного сховища даних.

Ключові слова: гібридне сховище даних, проектування, підхід, алгоритм.

This paper is devoted to the analysis of approaches and algorithms of the hybrid warehouse design. Generalized hybrid data warehouse is introduced.

Keywords: hybrid data warehouse, design, approach, algorithm.

Постановка проблеми

Під час проектування сховищ даних постає питання коректної побудови структур даних для ефективного його функціонування. Здебільшого використовують бази даних (БД) двох класів – OLTP (On-Line Transaction Processing), які застосовуються для підтримки оперативних сховищ даних, та OLAP (On-Line Analytical Processing). Прикладами таких систем є відповідно банківські транзакційні системи (OLTP) й аналітичні системи бюджетування (OLAP). Однак у практичній діяльності для виконання можливих запитів недостатньо спроектувати сховище даних лише одного типу. Тому сховище повинно поєднувати бази даних обидвох класів у т.зв. гібридне сховище даних (ГСД). Актуальним та невирішеним сьогодні є питання автоматизованого чи напівавтоматизованого проектування гібридних сховищ даних. Ця проблема зв'язана з практичною необхідністю автоматизованої побудови сховищ даних з оптимальною швидкодією.

Аналіз останніх досліджень та публікацій

Питаннями проектування та оптимізації сховищ даних займалися Claudia Imhoff, Ryan Sousa [1], Douglas Hackney [2], Goran Velinov, Danilo Gligoroski, Margita Kon Popovska [3], Hugh J. Watson, Thilini Ariyachandra [4], Ralph Kimball, Margy Ross [5], W. H. Inmon [6] , Wayne Eckerson [7], Wen-Yang Lin,I-Chung Kuo [8], Tapio Niemi, Jyrki Nummenmaa, Peter Thanisch [9]. У праці [10] розглянуто основні принципи побудови та функціонування сховищ даних. У роботах [11] і [12] досліджуються питання проектування сховищ даних для бюджетування.

У книзі «The Data Warehouse Toolkit. Second Edition» R. Kimball запропонував підхід до побудови сховищ даних, який також відомий як підхід «знизу вгору», оскільки проектування сховища починається з нижнього рівня сховища (DAT) і закінчується його верхнім рівнем (OSS) [5].

Запропонована архітектура багатовимірного сховища даних дає змогу ефективно виконувати запити як до агрегованих даних, так і зберігати детальні дані. Використання архітектури шини та понять узгодженості дає змогу отримати незалежні між собою вітрини, які використовують один набір вимірів, що забезпечує можливість паралельного розроблення вітрин різними підрозділами, а також цілісність усієї системи загалом.

Сховище, побудоване за цим підходом, містить такі компоненти (рис. 1):

- операційні бази даних (OSS);
- перехідна область (DSS);
- вітрини даних (DPA);
- засоби доступу до даних (DAT).

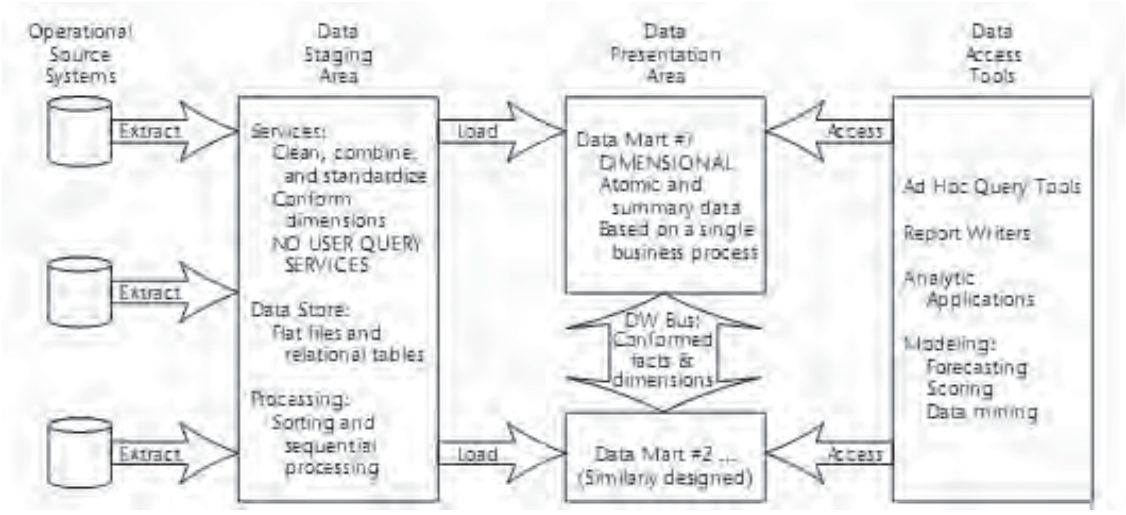

Рис. 1. Архітектура багатовимірного сховища даних

У цьому сховищі операційні бази даних (OSS) зберігають первинну інформацію. Перехідна область (DSA) слугує для перенесення даних з OSS до вітрин даних (DPA). Вітрини даних (DPA) є компонентом, у якому містяться як детальні, так і агреговані дані. Способом організації області представлення даних слугує шина, для якої вводиться матриця шини вимірів. Ця матриця описує наявність вимірів для кожної вітрини, застосовуючи поняття узгодженості вимірів. Поняття узгодженості вимірів вводять в означенні 1, використовуючи при цьому поняття повної узгодженості двох вимірів вітрин, узгодженості вітрин за атрибутами й узгодженості вітрин за вимірами.

- Два виміри у двох вітринах називаються повністю узгодженими, якщо таблиці цих вимірів повністю збігаються за атрибутами та елементами.
- Два виміри у двох вітринах називаються узгодженими за атрибутами, якщо множина атрибутів виміру однієї вітрини є підмножиною атрибутів виміру іншої вітрини.
- Два виміри у двох вітринах називаються узгодженими за елементами вимірів, якщо множина елементів виміру однієї вітрини є підмножиною елементів виміру іншої вітрини.

Виміри двох вітрин називатимемо узгодженими, якщо вони узгоджені повністю, узгоджені за атрибутами або за вимірами.

Використання узгоджених вимірів дає змогу отримати множину вітрин, які будуть синхронізовані між собою, а значить, однаково відображатимуть дійсність, з точністю до даних, які містяться в вітрині. Для забезпечення цілісності даних у межах всієї системи, недопущення різного трактування фактів, які фізично розміщені в різних вітринах даних, використовується поняття узгодженості фактів. Факт називається узгодженим, якщо він має в різних вітринах даних одне і те саме значення. До узгоджених фактів належать: ціна продуктів, загальні витрати та доходи, міри якості, задоволеності клієнтів та інші ключові показники продуктивності (KPI). Також використовуються однакові одиниці вимірювання в різних вітринах для запобігання існування різних значень одного і того самого факту.

Засоби доступу до даних (DAT) використовують для надання бізнес-користувачам інформації, яка міститься у вітринах даних, а також завантаження отриманих результатів роботи користувачів в OSS або DSA/DPA.

Сховище, запропоноване R. Kimball, має такі переваги:

- містить дружні для користувачів та гнучкі структури даних;
- мінімізує операції супроводу, а також надмірні структури даних для прискорення впровадження і зменшення вартості системи;
- знімає проблему деталізації даних, оскільки атомарні дані завжди зберігаються у вітринах даних;
- масштабується за допомогою розширення наявних зірок (вітрин) або побудови нових у межах тієї самої логічної моделі.

Основні недоліки цього сховища такі:

- ускладнюється з'єднання даних між декількома фізично відокремленими вітринами даних;
- вимагає узгодженого коректного використання вимірів від різних робочих груп організації під час його проектування;
- не призначене для підтримки операційних сховищ даних або операційних структур даних.

Другим основним підходом є підхід B.Inmon. У своїх роботах [1, 6] B.Inmon та ін. виклали підхід до побудови сховищ даних, який дає можливість зберігати дані централізовано в одному сховищі даних, забезпечуючи одну версію правди та запобігаючи розбіжностям. Сховище, побудоване за таким підходом, було названо «корпроративною інформаційною фабрикою». Крім того, наявність вітрин даних дає змогу отримувати агреговані дані та використовувати переваги систем класу on-line analytic processing (OLAP). Цей підхід також відомий як підхід «зверху вниз», оскільки проектування системи починається з верхнього рівня (EDW, описана нижче) і закінчується нижнім рівнем (вітрини даних).

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/видобування даних, альтернативні системи зберігання даних.

Структура такого сховища показана на рис. 2.

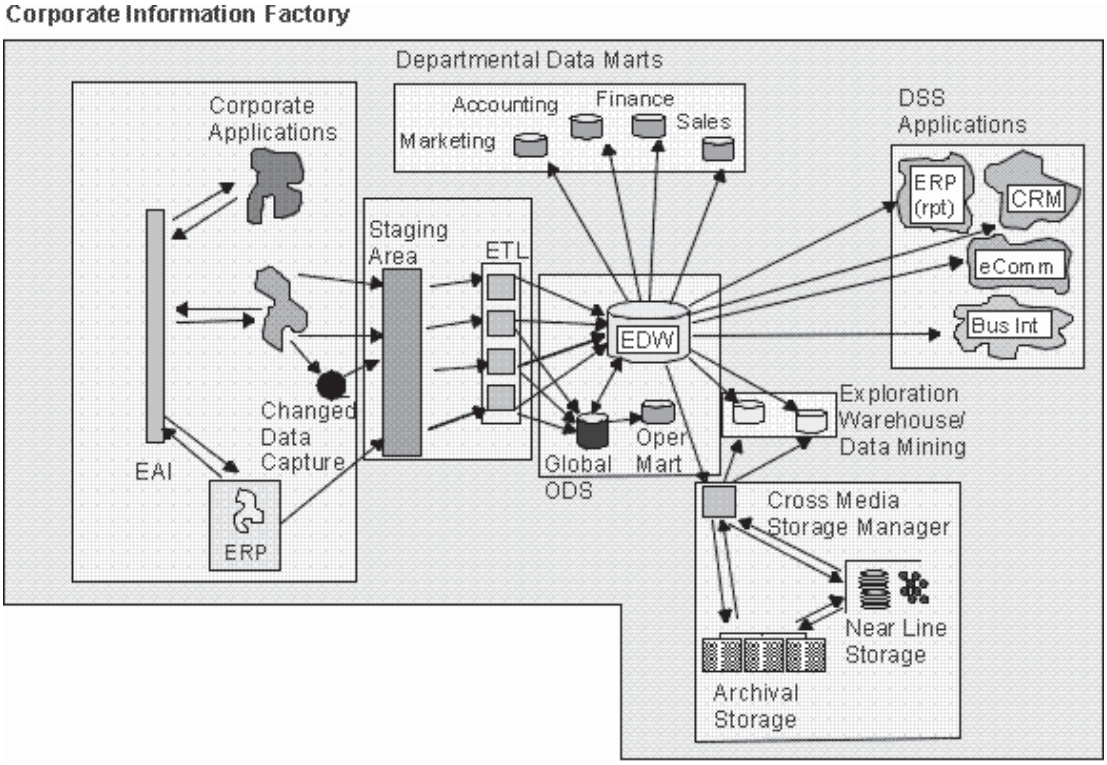

Рис. 2. Архітектура корпоративної інформаційної фабрики [4]

Корпоративні застосування є джерелом інформації у системі. Перехідна область переносить дані до основного сховища даних. Операційне сховище даних та операційні вітрини даних являють собою реляційні та багатовимірні бази даних, які зберігають всі або деяку частину первинних даних (як правило, поточних). Запити користувачів до цих сховищ не виконуються. Enterprise Data Warehouse зберігає детальні дані в системі. Вітрини цих відділів є багатовимірними базами даних або реляційними на основі схеми типу зірка, що належать до класу OLAP. Застосування систем прийняття рішень (ERP,CRM, електронної комерції) використовуються для отримання інформації з сховища. Сховища для дослідження/видобування даних являють собою окреме підсховище даних для виконання специфічних завдань роботи з даними (дослідження/видобування даних). Альтернативні системи зберігання даних зберігають архівні дані (історичну інформацію). Сюди входять систем архівування та близького зберігання даних, а також системи керування ними.

Сховище, запропоноване B. Imnon, має такі переваги:

- забезпечує наявність єдиної версії правди;
- забезпечує наявність єдиної множини процесів та бізнес-правил;
- забезпечує наявність загальної семантики;
- є централізованим, контрольованим середовищем;
- містить вітрини даних, які легко створювати та наповнювати;
- потребує лише єдиного репозиторію метаданих.

До основних недоліків цього сховища можна зарахувати:

- високу вартість реалізації;
- інтенсивне використання наданих йому ресурсів (обладнання);
- існує ризик краху всієї системи під час виходу з ладу центрального сховища.

Існують і інші, менш поширені підходи до побудови сховищ даних, які основані на двох вищенаведених.

Певною модифікацією підходу R. Kimball є т.зв. підхід незалежних вітрин даних [3]. Основною відмінністю цього підходу є незалежність і неузгодженість між собою вітрин даних. Також цей підхід не деталізує первинні бази даних, перехідну область і застосування користувачів.

Узагальненим варіантом підходу Інмона є підхід до побудови централізованого сховища даних [4], у якому багатовимірні представлення забезпечуються реляційною базою даних. 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 [4] передбачає наявність уже розгорнутої системи (бази даних, сховища даних, застосування тощо) і призначена для отримання доступу до даних з наявних джерел. Дані інтегруються (логічним чи фізичним способом), використовуючи спільні ключі, глобальні метадані, запити та інші методи. Застосування цієї архітектури рекомендується в існуючій складній інфраструктурі підтримки прийняття рішень, коли фірма не хоче перебудовувати цю інфраструктуру. Основною перевагою цієї архітектури є можливість використання наявних систем для побудови сховища. Основним недоліком цієї архітектури є складність та гетерогенність системи, що побудована за цією архітектурою.

Цей підхід найчастіше бажано використовувати у випадку, коли ресурси обмежені, набуті навики IT-персоналу є низькими, і сховище не є стратегічним рішенням (наприклад, частиною домену).

Перевагою підходу незалежних вітрин даних є простота проектування, а основним недоліком є складність аналізу даних у вітринах різної структури.

Своєрідним поєднанням підходів Інмона та Кімболла є так зване об'єднане (federated) сховище даних [6]. Таке сховище складається з первинних баз даних, загальної перехідної області, власне об'єднаного сховища даних та вітрин даних. Структура сховища показана на рис. 3.

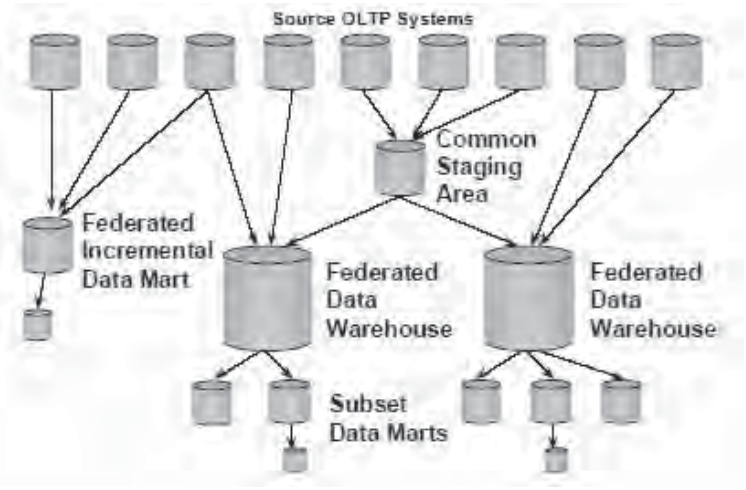

Рис. 3. Архітектура об'єднаного сховища даних

Автор пропонує в архітектурі сховища даних використовувати дві стандартні моделі сховищ даних – реляційну базу даних, в якій зберігаються первинні дані, і багатовимірні сховища, які виконують роль вітрин даних. Крім того, автор запропонував об'єднане сховище даних, яке отримує дані з реляційних баз даних і подає їх в багатовимірні вітрини даних.

Особливою рисою цієї архітектури є те, що в ній загальні дані розподіляються між сховищами даних і вітринами даних, забезпечуючи цим одну версію правди в сховищі.

За твердженням автора, цей підхід має такі переваги:

- загальна семантика і правила функціонування предметної області;
- один набір процесів витягання і функціонування даних;
- децентралізовані ресурси і керування;
- можливість паралельного розроблення.

Також автор вказує на такі недоліки підходу:

- необхідність в узгодженні робіт різних підрозділів у проектуванні сховища;
- складнощі в подоланні «політичних» моментів і вирішенні питань авторських прав;
- складне технічне середовище;
- наявність численних репозиторіїв метаданих.

Основною працею з проектування сховищ даних, зокрема багатовимірних баз даних класу OLAP, є [9]. У цій статті описується побудова кубів OLAP на основі запитів користувачів. Основна ідея статті в тому, що куб OLAP проектується не під предметну область, а під запити користувачів, що робить їх виконання ефективнішим.

Метод побудови сховищ складається з таких кроків:

1. Сховище даних представляється користувачу як загальний базовий куб.
2. Користувач будує запити за допомогою загального куба даних.
3. Система будує OLAP куб, що відповідає запитам користувача. Метод побудови куба оснований на використанні функціональних залежностей.
4. Куб OLAP відображається користувачу, він може бути прийнятий, змінений та відхилений. Зміни можуть бути як в запитах, так і прямо в структурі куба, тобто деякі атрибути можуть бути додані або вилучені.
5. Система аналізує куб і повідомляє користувача про якість куба даних.

Також у цій статті було використане поняття нормалізації кубів і схожості запитів для визначення кількості та структури кубів.

Основною роботою з оптимізації сховищ даних є [3]. У ній запропонований гібридний генетично-жадібний алгоритм до побудови сховищ даних, зокрема реляційних баз даних. Цей алгоритм полягає в поєднанні генетичного і жадібного алгоритмів для оптимізації реляційних баз даних за цільовою функцією, введеною в роботі.

Більшість роботи в галузі OLAP були зосереджені на питанні ефективної оптимізації. В деяких роботах також вивчалося питання проектування OLAP кубів, але запропоновані методи можуть бути занадто складними для початківців.

Об'єкт (просторові відношення, агреговані подання та індекси) представляється масивом бітів, тобто за частинами, які будуть об'єднані в хромосому. Просторові подання представляються (як особливий тип представлень для матеріалізації) з їх різними варіантами. Кількість бітів, необхідних для подання кожного просторового відношення з всіма його варіантами, дорівнює $k+1$, де k – кількість елементів додаткового набору атрибутів. Тому перший біт використовується для подання просторового відношення, а інші n бітів – для подання додаткових атрибутів. Всі просторові відношення мають бути матеріалізовані, тому кожне з них представляється 1. Атрибут в додатковому наборі, при додаванні до його просторового відношення, представляється 1, інакше 0. Агреговані подання представляються схожим способом, тобто для кожного подання один біт використовується для його подання (1 – якщо вибране, 0 – не вибране для матеріалізації) і k бітів використовуються для подання його додаткових атрибутів , тобто варіантів представлень. Атрибути додаткового набору агрегованих подаються у схожий спосіб з атрибутами додаткових наборів просторових відношень. Для кожного агрегованого

подання атрибут міри подається n бітами. Атрибут міри, якщо він додається до відповідного подання, представляється 1, інакше 0. Для кожного подання має бути доданий якнайменше один атрибут міри. Після подання кожного подання відбувається подання їх можливих індексів. Кожний індекс представляється одними бітом, тобто 1 – якщо індекс вибраний, 0 – не вибраний. Кількість і порядок індексів визначають попередньо.

Розроблені авторами алгоритми є гібридними, тому що вони є поєднанням жадібного і генетичного алгоритмів. Алгоритм 1 називається GGLA – Greedy-Genetic Linear Algorithm.

Вхідними параметрами алгоритмів є: NC – кількість хромосом (розмір популяції), NG – кількість поколінь, LC – довжина хромосоми (кількість бітів, необхідних для подання всього простору рішень, NF – кількість фрагментів простору рішень, яка дорівнює кількості кроків жадібної процедури, SQ – набір запитів, AV – набір агрегованих представлень. POP(i) – і-те покоління популяції.

Нижче подана схема алгоритму 1.

```
Algorithm 1: GGLA(NC, NG, LC, NF, AV, SQ)
Begin
AVB:=Ø;
Choose_views(Round(LC/NF), CL, AVB,AVC, AV);
Extend_chromosome(POP(1), AVC, NC);
Evaluate_population(POP(1), SQ);
Evaluate_population_materialization(POP(1));
j:=2;
For i=2 to NG Do
If Mod(i,Round(NG/NF))=1 Then
Choose_views(Round(LC*j/NF),CL,AVB,
AVC,AV);
Extend_chromosome(POP(i), AVB, NC);
j:=j+1;
End If;
Perform_crossover(POP(i-1), NC);
Perform_mutation(POP(i), NC);
Evaluate_population(POP(i), SQ);
Evaluate_population_materialization(POP(i));
Perform_selection(POP(i), NC);
End For;
End GGLA;
```

У статті [8] 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 і того самого набору вимірів за допомогою групування, як вказано на рис. 4.

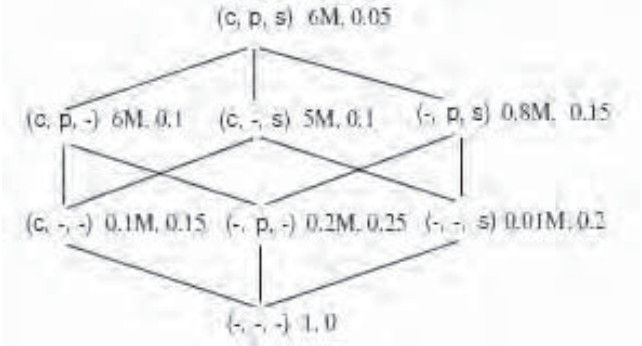

Рис. 4. Решітка кубів

На цьому рисунку позначені куби з вказанням вимірів, за якими відбувається групування, розмір кожного куба. Границя між двома кубами c_i та c_j представляє відношення залежності між цими двома кубами. c_i є залежним від c_j (позначається $c_i \prec c_j$), якщо будь-який запит, відповідь на який отримується з сі, може одержати відповідь з c_j, але не навпаки.

Загальний алгоритм побудови кубів виглядає так :

Ініціалізувати параметри;
Генерувати популяцію P випадковим способом;
  generation←1;
  поки generation≤max_gen виконувати
        Очистити нову популяцію P';
        для кожної особини $x \in P$ виконувати
              якщо x недопустиме тоді
Виконати алгоритм відновлення для x;
              Оцінити придатність x;
        кінець для
        поки $|P'|$ ≤population_size виконувати
              Вибрати двох батьків з P;
              Виконати схрещування;
              Виконати мутацію;
              Помістити нащадків в P';
        кінець поки;
        P←P';
        generation← generation+1;
  кінець поки;

Але для проектування узагальненого гібридного сховища даних, яке визначене вище, цих рішень недостатньо. Для проектування потрібна передусім наявність алгоритму (процедура) розподілу даних у сховищі. Тому в подальших дослідженнях розглядатимуться питання алгоритму автоматизованого проектування такого сховища даних.

Формулювання цілей статті

Метою статті є запропонувати нову структуру сховища даних, яка б дала змогу вирішити проблеми ефективності використання реляційної та багатовимірної баз даних.

Виклад основного матеріалу

Підходи централізованого сховища даних і незалежних вітрин даних є неефективними в загальному випадку, оскільки являють собою реляційну або багатовимірну БД, кожна з яких має певні недоліки, і неефективно виконуватиме запити (які не можуть бути ефективно виконані на відповідно багатовимірній та реляційній БД).

Підхід об'єднаного сховища даних не є прийнятним, оскільки він призначений для інтеграції наявних баз даних. Сховище даних не має певної визначеної структури, тому не може проектуватися.

Основоположними найпоширенішими є підходи Інмона та Кімболла. Порівняємо їх.

Спільні риси. Найочевиднішою подібністю між підходами є те, що вони основані на використанні ETL і використовують дані з відміткою часу, який є найважливішою визначальною характеристикою даних у сховищі. Час є важливим виміром в сховищі даних, оскільки він дає змогу 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

за інтервалами дат, окремими днями і періодами звітності. Завдяки шару ETL, підходи Інмона та Кімболла залежні у подібний спосіб від функції ETL – вивантаження даних з джерела, застосування правил для завантаження даних у сховище або у 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 використовувані вітрини 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 основним питанням є проектування сховища, тобто визначення структур даних, які необхідні для збереження цих даних, і, за змогою, оптимального функціонування (швидкодії) сховища.

Через недоліки моделей Інмона та Кімболла є очевидним поєднання цих архітектур і створення гібридного сховища даних. Це було зроблено в моделі Хекні, однак його підхід не є достатньо деталізованим, тому пропонуємо використання так званого узагальненого гібридного сховища даних. Основними рисами такого сховища є наявність своєї окремої системи керування та автоматизоване проектування сховища під популяції запитів.

Узагальнене гібридне сховище даних передбачає, що:

- Гібридне сховище даних має свою систему керування гібридним сховищем даних (СКГСД), за допомогою якої здійснюється робота з сховищем (виконання запитів до сховища).
- У гібридному сховищі наявна реляційна БД.
- У гібридному сховищі наявна багатовимірна БД, яка може містити як атомарні, так і узагальнені дані
- Поєднуються підходи проектування «зверху вниз» і «знизу вгору». Реляційна і багатовимірна бази даних проектуються в комплексі, для забезпечення оптимального функціонування сховища даних загалом.

Структура сховища показана на рис. 5.

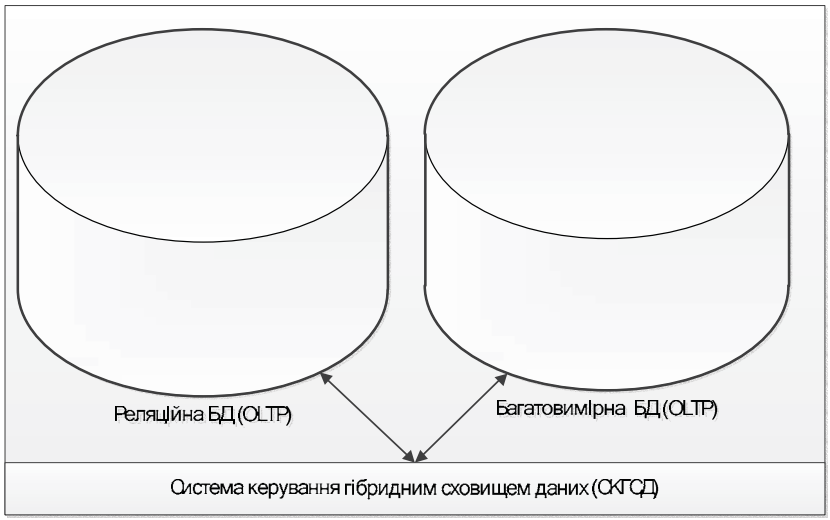

Рис. 5. Структура узагальненого гібридного сховища даних

Висновки

У статті проаналізовано підходи та алгоритми проектування сховищ даних. Запропонована концепція узагальненого гібридного сховища даних, яка дасть змогу проектувати сховище даних з урахуванням запитів, що надходять до сховища даних, тим самим забезпечуючи загальний інтерфейс доступу до інформації, незалежно від фізичної бази даних. Також дані можуть мігрувати з однієї бази даних в іншу, що допоможе підвищити швидкодію роботи сховища на певній популяції запитів.

У подальших дослідженнях необхідно розробити алгоритм проектування ГСД із визначенням оптимального розташування даних відповідно до запитів, які надходять до сховища.

1. Claudia Imhoff.Corporate information factory / Claudia Imhoff, Ryan Sousa – John Wiley & Sons, 2001 – 382 p. 2. Douglas Hackney. Architectures and Approaches for Successful Data Warehouses. / Douglas Hackney – www.egltd.com/presents/ArchitecturesApproaches.pdf. 3. Goran Velinov. Framework for Generalization and Improvement of Relational Data, IJCSNS International Journal of Computer Science and Network Security / Goran Velinov , Danilo Gligoroski, Margita Kon Popovska, VOL.8 No.3, March 2008. 4. Hugh J. Watson. Data Warehouse Architectures: Factors in the Selection Decision and the Success of the Architectures [Електронний ресурс] / Hugh J. Watson, Thilini Ariyachandra 2003 – Режим доступу: http://www.terry.uga.edu/~hwatson/DW_Architecture_Report.pdf. 5. Ralph Kimball. The data warehouse toolkit: the complete guide to dimensional modeling / Ralph Kimball – Wiley, 2002 – 436 p. 6. W. H. Inmon. Corporate Information Factory Components [Електронний ресурс] / W. H. Inmon – Inmon Data Systems – Режим доступу : http://www.inmoncif.com/view/26. 7. Wayne Eckerson. Four Ways to Build a Data Warehouse [Режим доступу] // Wayne Eckerson – TDWI -2003 – Режим доступу http://www.tdwi.org/research/display.aspx?ID=6699 8. Wen-Yang Lin. A Genetic Selection Algorithm for OLAP Data Cubes// Wen-Yang Lin,I-Chung Kuo – Knowledge and information systems, 2004, vol. 6. 9. Tapio Niemi. Constructing OLAP Cubes Based on Queries. Proceedings of the 4th ACM international workshop on Data warehousing and OLAP / Tapio Niemi, Jyrki Nummenmaa, Peter Thanisch ,2001. 10. Шаховська Н.Б. Сховища та простори даних: Монографія / Н.Б. Шаховська, В.В. Пасічник. – Львів: Вид-во Нац. ун-ту «Львівська політехніка» , 2009. – 244 с. 11. Яцишин Ю.В. База знань для формування переліку бюджетних програм інформаційно-аналітичної системи управління державними фінансами, III Міжнародна науково-технічна конференція "Комп'ютерні науки та інформаційні технології" (CSIT-2008) / Ю.В. Яцишин, А.Ю. Яцишин. – Львів, 2008. 12. Яцишин А.Ю. Проектування гібридного сховища даних для роботи з державним бюджетом України. II Всеукраїнська науково-практична конференція «Інформаційні технології та автоматизація – 2009» / А.Ю. Яцишин. – Одеса, 2009.