МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

КОЗЛОВСЬКА В. П.

МЕТОДИ ТА ЗАСОБИ ІНТЕГРАЦІЇ ДАНИХ

Конспект лекцій

Одеса
Одеський державний екологічний університет
2016

УДК 681.3
К59

Рекомендовано методичною радою Одеського державного екологічного університету Міністерства освіти і науки України як конспект лекцій (протокол №10 від 04.07. 2016 р.)

Козловська В. П.
Методи та засоби інтеграції даних: конспект лекцій. Одеса, Одеський державний екологічний університет, 2016, 121с.

Метою дисципліни «Методи та засоби інтеграції даних» є формування у студентів теоретичних знань та практичних навичок, необхідних для роботи у будь-яких адміністративних, наукових та виробничних підрозділах, в яких використовують сховища даних, а також здійснюють обслуговування та розробку інформаційних систем та сховищ даних.

В курсі розглядаються призначення сховищ даних та принципи їх організації. Сховища даних є джерелом інформації для систем підтримки прийняття рішень (СППР), за допомогою яких отримуються рішення по тактичному та стратегічному плануванню розвитку підприємств та корпорацій.

Основною вимогою сучасного бізнесу є глобальний погляд на діяльність підприємства і вміння управляти наскрізними процесами, інтегруючи всі системи автоматизації, що їх підтримують. Однак серйозно перешкоджають єдиному поданню задач бізнесу інформаційні активи, які можуть бути «замкнені» в розрізнених корпоративних додатках. Вирішити проблему можна за допомогою систем інтеграції даних.

Курс «Методи та засоби інтеграції даних» призначені для магістрів І року, що навчаються за спеціальністю 122 Компютерні науки.

ISBN 978-966-186-091-8

ЗМІСТ

1 СХОВИЩЕ ДАНИХ 5
1.1 Сховище даних як об'єкт для інтеграції даних 5
1.2 Системи обробки та зберігання інформації 6
1.2.1 Причини виникнення СД 8
1.2.2 Основні особливості концепції СД 11
1.3 Основні концепції сховищ даних 12
1.3.1 Визначення та основні характеристики 13
1.3.2. Завдання, що вирішуються за допомогою сховищ даних 16
1.3.3. Характеристика даних у сховищі 19
1.4. Складові сховища даних 33
1.4.1. Учасники СД 33
1.4.2 Програмне забезпечення 36
1.4.3. Параметри 37
1.4.4. Запити 38
1.5 Огляд архітектури СД 39
1.6 Багатовимірні сховища даних 40
1.6.1 Основи багатовимірного представлення даних 40
1.6.2 Вимірювання і факти – базові поняття багатовимірної моделі даних 41
1.6.3 Структура багатовимірного куба 42
1.6.4 Робота з вимірами 44
1.7 Реляційні сховища даних 46
1.7.1 Схеми побудови РСД 47
1.7.2 Переваги та недоліки РСД 49
1.8 Гібридні сховища даних 50
1.9. Види ієрархії вимірів 52
1.10 Підвиди сховищ даних 55
1.10.1. Вітрини даних 55
1.10.2. Операційні сховища даних 58
Резюме 60
2 ІНТЕГРАЦІЯ ДАНИХ 62
2.1. Значення інтеграції для сховищ даних 62
2.2. Поняття інтеграції даних 66
2.2.1 Історія засобів інтеграції 67
2.2.2 Проблеми, що призводять до інтеграції даних 68
2.3. Характеристики інтеграції даних 70
2.4. Методи інтеграції даних 71
2.4.1. Консолідація даних 71
2.4.2. Федералізація даних 74
2.4.3. Розповсюдження даних 76
2.4.4. Гібридний підхід 77

2.5. Технології інтеграції 78
2.6 Технологія ETL 80
2.6.1. Структура процесу перевантаження даних 80
2.6.2. Підпроцес STER 86
2.6.3. Підпроцес STAC 88
2.6.4. Переваги та недоліки ETL 94
2.7. Технологія EII 94
2.8. Технологія EAI 95
2.9. Технологія ECM 97
2.9.1. Поняття ECM 97
2.9.2. Компоненти ECM-рішення 98
2.10 Засоби інтеграції даних 103
2.10.1 Основний інструментарій 103
2.10.2 Архітектура систем інтеграції 103
2.10.3 Методи реалізації інтеграції даних 105
2.10.4 Магічний квадрант Gartner 109
2.10.5 Засоби інтеграції даних компанії Informatica 112
2.11. Семантична інтеграція даних 115
2.11.1 Способи представлення даних для інтеграції 115
2.11.2 Семантична інтеграція 116
2.11.3 Приклад продуктів сучасного ринку ПЗ для реалізації семантичної інтеграції 117
2.11.4 Засоби семантичної інтеграції даних 118
Резюме 119
СПИСОК ЛІТЕРАТУРИ 121

1 СХОВИЩЕ ДАНИХ

1.1 Сховище даних як об’єкт для інтеграції даних

Основною вимогою сучасного бізнесу стає глобальний погляд на діяльність підприємства і вміння управляти наскрізними процесами, інтегруючи всі системи автоматизації, що їх підтримують. Однак серйозно перешкоджають єдиному поданню задач бізнесу інформаційні активи, які досі «замкнені» в розрізнених корпоративних додатках. Вирішити проблему можна за допомогою систем інтеграції даних

За визначенням аналітиків Gartner, інтеграція даних охоплює практики, архітектурні підходи і програмні інструменти для забезпечення узгодженого доступу і доставки даних для всього спектру додатків і бізнес-процесів компанії (Magic quadrant for data integration tools, Gartner, 2006). Як свідчать дослідження, витрати на програмні засоби інтеграції даних сьогодні неухильно ростуть в самих різних індустріях і географічних регіонах. Це відбувається через невідповідність існуючих підходів до управління даними і ситуації з автоматизованою підтримкою операцій бізнесу з боку прикладних систем. Управління наскрізними бізнес-процесами, які охоплюють різні підрозділи компанії і її зовнішніх партнерів і замовників, демонструє свою ефективність і підкріплено цілком зрілими методами і технологіями інтеграції, в тому числі на базі SOA (service-oriented architecture – сервісно-орієнтована архітектура). При цьому ключові дані на різних стадіях бізнес-процесів тісно прив’язані до додатків, які з ними працюють, але узгоджене для різних систем подання даних відсутнє.

Тим часом таке подання дійсно необхідно. Наприклад, основні дані по клієнтах і продуктах в різних підрозділах – маркетинговому, фінансовому, відділі продажів і т.д. – можуть визначатися і використовуватися по-різному; більш того, кожен підрозділ може мати своє власне джерело даних. Щоб включити операції цих підрозділів до загального бізнес-процесу, потрібно привести розрізнені дані до загального вигляду – тільки в цьому випадку буде забезпечена необхідна ефективність і простота виконання бізнес-процесу. Однак якщо відсутня загальна корпоративна стратегія і відповідний інструментарій інтеграції даних, їх зведення воєдино реалізується за допомогою спеціальних програмних налаштувань для кожного конкретного додатку, що тільки ускладнить задачу.

Неузгодженість даних і неможливість уніфікованого доступу до них стає особливо очевидною і болючою в ситуаціях злиття і поглинання компаній. Існування безлічі розрізнених джерел даних і відсутність механізмів їх об’єднання і узгодження, ускладнюють використання інформації, оскільки для прийняття обґрунтованих рішень потрібне єдине джерело достовірних даних в масштабах всієї компанії.

До програмних систем інтеграції даних аналітики відносять рішення, що забезпечують інфраструктуру доступу і доставки даних для наступних сценаріїв інтеграції:

1. Отримання даних для сховищ даних і систем бізнес-аналітики – вилучення даних з систем підтримки оперативної діяльності, трансформація і об'єднання цих даних, уявлення інтегрованих даних для вирішення аналітичних задач. Сховища даних зберігають свою домінуючу роль серед можливих застосувань засобів інтеграції даних.
2. Створення інтегрованих сховищ основних, або майстер-даних – забезпечення консолідації та раціоналізації даних, що мають важливе значення для бізнесу (наприклад, дані про клієнтів, продукти і співробітників). У рішеннях по управлінню основними даними (Master Data Management, MDM) інструментарій інтеграції забезпечує ключові процеси їх консолідації і синхронізації.

1.2 Системи обробки та зберігання інформації

До середини 80-х рр. XX ст. практично повністю завершився перший етап оснащення бізнесу і державних структур засобами обчислювальної техніки і почався період бурхливого розвитку інформаційних систем для організації збору і зберігання великих масивів різного роду ділової та службової інформації. В основному це були корпоративні системи, призначені для оперативної обробки інформації, які обслуговували бухгалтерію, інформаційні архіви, телефонні мережі, реєстрацію документів, банківські операції і т.д. З появою персональних комп'ютерів такі системи стали доступними для безлічі дрібних і середніх фірм, підприємств і організацій. Системи оперативної обробки інформації отримали назву OLTP (On-Line Transaction Processing – оперативна, тобто в режимі реального часу, обробка транзакцій).

Узагальнена структура системи OLTP представлена на рис.1.1.

Типовим прикладом застосування OLTP-систем є масове обслуговування клієнтів, наприклад, бронювання авіаквитків або оплата послуг телефонних

компаній. Обидві ці ситуації мають два загальних властивості: дуже велике число клієнтів і безперервне надходження інформації.

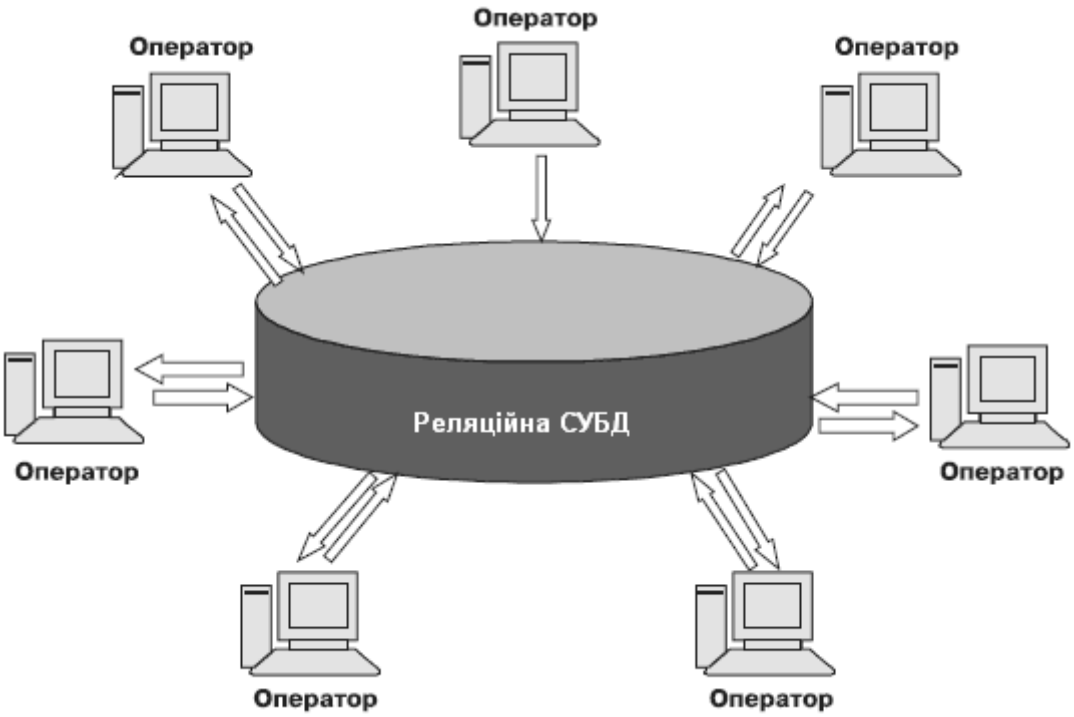


Рис. 1.1 – OLTP-система

Такі транзакції виконуються тисячі разів в день в сотнях пунктів продажу. Очевидно, що основним пріоритетом в даному випадку є забезпечення мінімального часу відгуку при максимальному завантаженні системи. Розглянемо характерні риси даного процесу, властиві в тій чи іншій мірі всім OLTP-систем.

- Запити і звіти повністю регламентовані. Оператор не може сформувати власний запит, щоб уточнити чи проаналізувати будь-яку інформацію.
- Як тільки переліт завершився , інформація про обслуговування даного клієнта втрачає сенс, стає неактуальною і підлягає видаленню після певного часу (тобто історичні дані не підтримуються).
- Операції виробляються над даними з максимальним рівнем деталізації, тобто по кожному клієнту окремо.

Ситуація докорінно змінюється, коли керівництво авіакомпанії приймає рішення про вивчення пасажиропотоків з метою, наприклад, їх оптимізації. Такі дослідження можуть бути реакцією на інформацію про те, що останнім часом у багатьох пунктах продажів почастішали випадки нестачі квитків на

певні маршрути, що дозволяє зробити припущення про доцільність організації додаткових рейсів.

Однак для проведення таких досліджень необхідні як мінімум три речі. По-перше, потрібні дані про продажі квитків за досить тривалий період (кілька місяців або років). По-друге, дані не повинні містити протиріч, пропусків, аномальних значень і інших чинників, які не дозволять виконати коректний аналіз. По-третє, необхідна додаткова інформація про бізнес-середовищі: про конкурентів, ринкові тенденції, ціни на паливо та ін. Очевидно, що типова OLTP-система не може забезпечити нічого з перерахованого. Саме з розумінням цих проблем приходить усвідомлення необхідності використання більш розвинених систем зберігання даних, орієнтованих на аналіз.

1.2.1 Причини виникнення СД

Головна вимога до OLTP-систем – швидке обслуговування відносно простих запитів великої кількості користувачів, при цьому час очікування виконання типового запиту не повинно перевищувати кілька секунд. Згодом в таких системах почали акумулюватися великі обсяги даних – документи, відомості про банківські операції, інформація про клієнтів, укладених угодах, надані послуги тощо

Поступово виникло розуміння того, що збір даних не самоціль. Зібрана інформація може виявитися досить корисною в процесі управління організацією, пошуку шляхів вдосконалення діяльності та отримання за допомогою цього конкурентних переваг. Але для цього потрібні системи, які дозволяли б виконувати не тільки найпростіші дії над даними: підраховувати суми, середні, максимальні і мінімальні значення.

З’явилася потреба в інформаційних системах, які дозволяли б проводити глибоку аналітичну обробку, для чого необхідно вирішувати такі завдання, як пошук прихованих структур і закономірностей в масивах даних, висновок з них правил, яким підпорядковується дана предметна область, стратегічне і оперативне планування, формування нерегламентованих запитів, прийняття рішень і прогнозування їх наслідків.

Розуміння переваг, які здатний дати інтелектуальний аналіз, призвело до появи нового класу систем – інформаційних систем підтримки прийняття рішень (інформаційних СППР), орієнтованих на аналітичну обробку даних з метою отримання знань, необхідних для розробки рішень в галузі управління. Додатковим стимулом вдосконалення цих систем стали такі фактори, як зниження вартості високопродуктивних комп'ютерів і витрат на зберігання

великих обсягів інформації, поява можливості обробки великих масивів даних і розвиток відповідних математичних методів. Узагальнена структурна схема інформаційної СППР представлена на рис. 1.2.

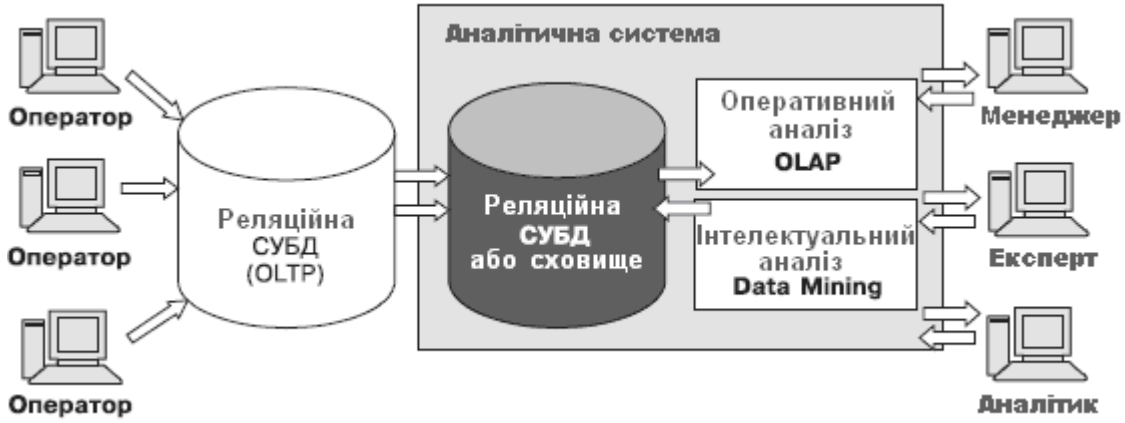


Рис. 1.2 – Структура інформаційної СППР

В основі роботи з такою СППР лежать запити, з якими до неї звертається користувач (особа, яка приймає рішення (ОПР) – менеджер, експерт чи аналітик). При цьому запити, допустимі в традиційних системах оперативної обробки даних, дуже примітивні. Наприклад, для банку це може бути запит типу «Скільки грошей на рахунку клієнта?». Або «Скільки грошей клієнт витратив за останній місяць?». Очевидно, що цінність інформації, отриманої за допомогою подібного запиту, невелика. У той же час аналітична система може відповісти на набагато більш складні запити, наприклад: «Визначити середній час між виставленням та сплатою рахунку для кожної категорії клієнтів».

В процесі розробки систем аналізу інформації і методології їх застосування виявилося, що для ефективного функціонування такі системи повинні бути організовані трохи іншим способом, ніж той, який застосовується в OLTP-системах. Це обумовлено наступними причинами.

- Для виконання складних аналітичних запитів необхідна обробка великих масивів даних з різноманітних джерел.
- Для виконання запитів, пов'язаних з аналізом тенденцій, прогнозуванням протяжних в часі процесів, необхідні історичні дані, накопичені за досить тривалий період, що не забезпечується звичайними OLTP-системами.
- Дані, що використовуються для цілей аналізу і обслуговування аналітичних запитів, відрізняються від використовуваних в звичайних OLTP-системах. При аналітичної обробки перевагу надають

детальним даними, а узагальненим (агрегованих). Очевидно, що для аналізу продажів великого супермаркету інтерес представляє не інформація про окремі покупках, а скоріше відомості про продажі протягом певних часових інтервалів (наприклад, тижня чи місяця).

У зв'язку з цим можна виділити ряд принципових відмінностей СППР і OLTP-систем. Ці відмінності представлені в табл. 1.1.

Таблиця 1.1. Відмінності СППР і OLTP-систем

Властивість	**OLTP-система**	**СППР**
Цілі використання даних	Швидкий пошук, найпростіші алгоритми обробки	Аналітична обробка з метою пошуку прихованих закономірностей, побудови прогнозів і моделей і т.д.
Рівень узагальнення (деталізації) даних	Деталізовані	Як деталізовані, так і узагальнені (агреговані)
Вимоги до якості даних	Можливі некоректні дані (помилки реєстрації, введення і т.д.)	Помилки в даних не допускаються , оскільки можуть призвести до некоректної роботи аналітичних алгоритмів
Формат зберігання даних	Дані можуть зберігатися в різних форматах в залежності від програми , в якому вони були створені	Дані зберігаються і обробляються в єдиному форматі
Час зберігання даних	Як правило, не більше року (в межах звітного періоду)	Роки, десятиліття
Зміна даних	Дані можуть додаватися , змінюватися і віддалятися	Допускається тільки поповнення; раніше додані дані змінюватися не повинні, що дозволяє забезпечити їх хронологію
Періодичність оновлення	Часто, але в невеликих обсягах	Рідко, але в більших обсягах
Доступ до даних	Повинен бути забезпечений доступ до всіх поточних (оперативних) даними	Повинен бути забезпечений доступ до історичних (тобто накопиченим за досить тривалий період часу) даними з дотриманням їх хронології
Характер виконуваних запитів	Стандартні, налаштовані заздалегідь	Нерегламентовані , що формуються аналітиком «на льоту» в залежності від необхідного аналізу
Час виконання запиту	Кілька секунд	До декількох хвилин

Як видно з табл. 1.1, вимоги до СППР і OLTP-систем істотно відрізняються. Тому в СППР використовуються спеціалізовані бази даних, які називаються сховищами даних (СД). Сховища даних орієнтовані на аналітичну обробку та задовольняють вимогам, що пред'являються до систем підтримки прийняття рішень.

Системи управління базами даних (СУБД) (реляційні, постреляційні тощо) відносяться до OLTP-систем. СУБД забезпечує загальний репозиторій для зберігання і опрацювання структурованих даних. СУБД підтримує набір взаємозв'язаних послуг і дозволяє розробникам зосередитись на специфічних проблемах їх застосувань, а не на завданнях, які виникають при потребі в узгодженому й ефективному керуванні великими обсягами даних. Проте, СУБД вимагає, щоб всі дані знаходилися під єдиним адміністративним керуванням і відповідали єдиній схемі. У відповідь на задоволення цих обмежень СУБД може забезпечити розвинені засоби маніпулювання даними та опрацювання запитів зі зрозумілою і строгою семантикою, а також строгі транзакційні гарантії оновлень, паралельного доступу і довготривалого зберігання (так звані властивості ACID).

Сховища даних краще пристосовані до зберігання та аналітичного опрацювання великих обсягів даних і, в основному, є інтеграцією реляційної та багатовимірної моделей (корпоративна інформаційна фабрика Білла Інмона, шина Ральфа Кімбола, зведення даних корпорації TDAN) [6-8, 21, 27]. Вони мають розвинені засоби інтеграції даних з різних джерел та дозволяють працювати як з деталізованою, так і агрегованою інформацією

1.2.2 Основні особливості концепції СД

В даний час однозначного визначення СД не існує, через те що розроблено велику кількість різних архітектур і технологій СД, а самі сховища використовуються для вирішення найрізноманітніших завдань. Кожен автор вкладає в це поняття своє бачення питання. Узагальнюючи вимоги, що пред'являються до СППР, можна дати наступне визначення СД, яке не претендує на повноту і однозначність, але дозволяє зрозуміти основну ідею.

Сховище даних – різновид систем зберігання, орієнтована на підтримку процесу аналізу даних, що забезпечує цілісність, несуперечність і хронологію даних, а також високу швидкість виконання аналітичних запитів.

Найважливішим елементом СД є семантичний шар – механізм, що дозволяє аналітику оперувати даними за допомогою бізнес-термінів предметної

області. Семантичний шар дає користувачеві можливість зосередитися на аналізі і не замислюватися про механізми отримання даних.

Типове СД істотно відрізняється від звичайних систем зберігання даних. Головною відмінністю є цілі використання. Наприклад, реєстрація продажів і виписка відповідних документів – завдання рівня OLTP-систем, що використовують звичайні реляційні СУБД. Аналіз динаміки продажів і попиту за кілька років, що дозволяє виробити стратегію розвитку фірми і спланувати роботу з постачальниками і клієнтами, найзручніше виконувати за підтримки СД.

Інша важлива відмінність полягає в динаміці зміни даних. Бази даних в OLTP-системах характеризуються дуже високою динамікою зміни записів через повсякденної роботи великої кількості користувачів (звідки, до речі, велика ймовірність появи протиріч, помилок, порушення цілісності даних і т.д.). Що стосується СД, то дані з нього не видаляються, а поповнення відбувається відповідно до визначеного регламенту (раз на годину, день, тиждень, в певний час).

В останні десятиліття технологія СД стрімко розвивається. Десятки компаній пропонують на ринку свої рішення в області СД, і тисячі організацій вже використовують цей потужний засіб підтримки аналітичних проектів.

1.3 Основні концепції сховищ даних

Прийнято вважати, що біля витоків концепції СД стояв технічний директор компанії Prism Solutions Білл Інмон, який на початку 1990-х рр. опублікував ряд робіт, які стали основоположними для подальших досліджень в області аналітичних систем.

В основі концепції СД лежать наступні положення:

- **інтеграція та узгодження даних з різних джерел**, таких як звичайні системи оперативної обробки, бази даних, облікові системи, офісні документи, електронні архіви, розташовані як усередині підприємства, так і в зовнішньому оточенні;
- **розділення наборів даних і застосувань,** що використовуються для оперативного опрацювання і для розв'язування задач аналізу.

На рис. 1.3 представлена концептуальна модель сховища даних.

Рис. 1.3 – Концептуальна модель сховища даних

Основні вимоги до сховищ даних сформовані у книзі Т.А. Гаврилової, В.Ф. Хорошевського «Базы знаний интеллектуальных систем» [13].

Ідея сховищ, як відомо, не нова: вони з'явилися у середині 1980-х років, коли компанія IBM висунула термін «інформаційне сховище». Білла Інмона (Bill Inmon) часто іменують «батьком сховищ даних» за ту роботу, яку він здійснював у ті часи в канадському філіалі компанії Shell [12]. У 1994 році термін був широко визнаний і набув поширення в галузі, хоча на ринку ця новинка стала з'являтися лише в кінці 90-х років минулого століття. Якщо не рахувати дебатів між Біллом Інмоном і Ральфом Кімболом (Ralph Kimball), які пропонували дещо різні підходи, фактично, сам собою підхід до розроблення СД з тих часів не дуже змінився. Річард Хакаторн, ще один основоположник цієї концепції, писав, що мета сховищ даних – забезпечити для організації «єдиний образ реальності» [5].

1.3.1 Визначення та основні характеристики

Що ж таке сховище даних? Деякі автори пропонують наступне визначення:

Сховище даних – це агрегований інформаційний ресурс, що містить консолідовану інформацію з усієї проблемної області та використовується для підтримки прийняття рішень.

Консолідована інформація – це одержані з декількох джерел та системно інтегровані різнотипні інформаційні ресурси, які в сукупності наділені ознаками повноти, цілісності, несуперечності та складають адекватну інформаційну модель проблемної області, з метою її аналізу, опрацювання та ефективного використання в процесах підтримки прийняття рішень.

Інмон дав таке визначення СД: предметно-орієнтований, інтегрований, незмінний набір даних, що підтримує хронологію, призначений для забезпечення прийняття управлінських рішень .

Під предметною орієнтованістю в даному випадку мається на увазі, що СД має розроблятися з урахуванням специфіки конкретної предметної області, а не аналітичних додатків, з якими його передбачається використовувати. Структура СД повинна відображати уявлення аналітика про інформацію, з якої йому доводиться працювати.

Також прийнятим є наступне визначення сховища даних.

Сховище даних (англ. Data Warehouse) – предметно-орієнтована інформаційна база даних, спеціально розроблена і призначена для підготовки звітів і бізнес-аналізу з метою підтримки прийняття рішень в організації. Будується на базі систем управління базами даних і систем підтримки прийняття рішень.

Основні вимоги до СД

Ральф Кімболл (Ralph Kimball) означує сховище даних як «місце, де люди можуть дістати доступ до своїх даних». Він же сформулював і основні вимоги до сховищ даних:

- ♦ 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 у сховищі;
- ♦ повнота і достовірність даних, що зберігаються;
- ♦ підтримка якісного процесу поповнення даних.

Задовольняти всі перераховані вимоги у межах одного і того ж продукту часто не вдається. Тому для реалізації сховищ даних зазвичай використовується декілька продуктів, одні з яких є власне засоби зберігання даних, інші – засоби їх витягання і перегляду, треті, – засоби їх поповнення та ін.

Типове сховище даних зазвичай відрізняється від традиційної реляційної бази даних. По-перше, традиційні бази даних призначені для того, щоб допомогти користувачам виконувати повсякденну роботу, тоді як сховища даних призначені для прийняття рішень. Наприклад, продаж товару і виписування рахунку здійснюється з використанням бази даних, призначеної для опрацювання транзакцій, а аналіз динаміки продажів за декілька років, що дозволяє спланувати роботу з постачальниками, – за допомогою сховища даних.

По-друге, традиційні бази даних характеризуються постійними змінами у процесі роботи користувачів, а сховище даних відносно стабільне: дані у ньому зазвичай оновлюються за розкладом (наприклад, щотижня, щодня або

щогодини – залежно від потреб). В ідеалі процес поповнення (або як далі ми будемо називати *завантаження)* є просто додаванням нових даних за певний період часу без зміни попередньої інформації, що вже міститься в сховищі.

І, по-третє, традиційні бази даних найчастіше є джерелом даних, що потрапляють у сховище. Крім того, сховище може поповнюватися за рахунок зовнішніх джерел, наприклад статистичних звітів.

Розглянемо більш докладно властивості сховища даних, які вказує Інмон у своєму визначенні СД.

Предметна орієнтація. Сховище даних призначене для аналізу даних. Воно концентрується на предметі, а не на процесі. Структура СД повинна відображати уявлення аналітика про інформацію, з якої йому доводиться працювати. Так, можна побудувати сховище, орієнтоване на продажі.

Інтегрованість. Сховище даних повинне наводити дані, отримані з різних джерел або в різний час, до єдиного формату, щоб було легко здійснювати аналіз. Крім того, дуже важливо перевірити завантажуються дані на цілісність і несуперечливість, забезпечити необхідний рівень їх узагальнення (агрегування). Обсяг даних в сховищі повинен бути достатнім для ефективного вирішення аналітичних задач, тому в СД може накопичуватися інформація за кілька років і навіть десятиліть.

Незмінюваність. Дані, що один раз потрапили у сховище даних, не змінюються, за винятком додавання нових даних.

Варіантність у часі. Для дослідження бізнесу в часі сховище даних повинне підтримувати всю історію змін, щоб можна було подивитися на бізнес із історичної перспективи. Підтримка хронології означає дотримання порядку проходження записів, для чого в структуру СД вводяться ключові атрибути Дата і Час. Крім того, якщо фізично впорядкувати записи в хронологічному порядку, наприклад, в порядку зростання атрибута Дата, можна зменшити час виконання аналітичних запитів.

Ще однією властивістю сховища даних є його кросплатформність, оскільки інформація про джерела даних зберігається у метаданих. Наприклад, сховище даних складатиметься з трьох частин, реалізованих в СКБД Oracle, MS SQL та Firebird. Зв'язок цих частин з користувачем показано на рис. 1.4.

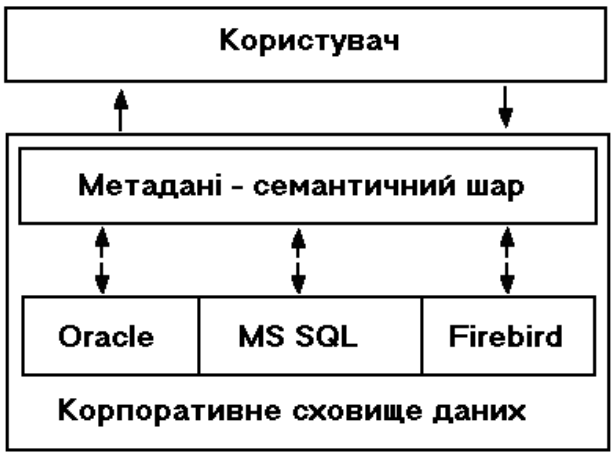


Рис. 1.4 – Кросплатформність як одна з ознак (необов’язкових) сховища даних.

Враховуючи специфіку, сховище даних має такі особливості проектування та побудови:

- отримання інформації з різних джерел даних (у тому числі і з реляційних баз даних) у деталізованому та агрегованому вигляді;
- багатовимірне подання інформації;
- наявність метаданих для опису джерел метаданих та структури самого сховища даних;
- наявність пакетного завантаження даних в сховище даних та вивантаження даних;
- наявність процедур аналізу даних та отримання нових даних;
- орієнтованість даних на аналітичне, а не на статичне опрацювання.

Використання концепції СД в СППР і аналізі даних сприяє досягненню таких цілей, як:

- своєчасне забезпечення аналітиків і керівників всією інформацією, необхідної для вироблення обґрунтованих і якісних управлінських рішень;
- створення єдиної моделі представлення даних в організації;
- створення інтегрованого джерела даних, що надає зручний доступ до різних видів інформації та гарантує отримання однакових відповідей на однакові запити з різних аналітичних програм.

1.3.2. Завдання, що вирішуються за допомогою сховищ даних

Процес розробки СД досить трудомісткий, деякі організації витрачають на нього кілька місяців і навіть років, а також вкладають значні фінансові кошти. Основними задачами, які потрібно вирішити в процесі розробки СД, є:

- вибір структури зберігання даних, що забезпечує високу швидкість виконання запитів і мінімізацію обсягу оперативної пам'яті;
- початкове заповнення і подальше поповнення сховища;
- забезпечення єдиної методики роботи з різнорідними даними і створення зручного інтерфейсу користувача.

Коло завдань інтелектуального аналізу даних досить широкий, а самі завдання істотно різняться за рівнем складності. Тому в залежності від специфіки вирішуваних завдань і рівня їх складності архітектура СД і моделі даних, що використовуються для їх побудови, можуть відрізнятися. Узагальнена концептуальна схема СД представлена на рис. 1.5.

Згідно зі схемою дані витягуються з різних джерел і завантажуються в СД, яке містить як власне дані, подані відповідно до деякої моделлю, так і метадані.

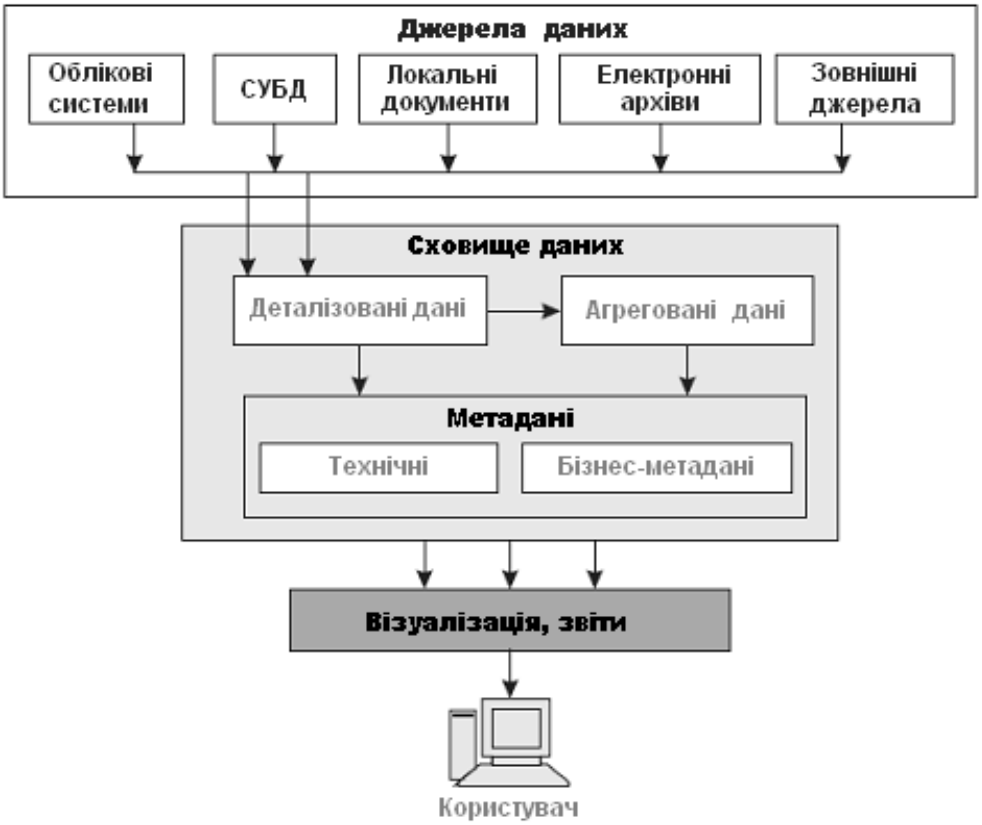


Рис. 1.5 – Концептуальна схема СД

Спектр застосування технології сховищ даних достатньо широкий. Вирішувані за допомогою сховищ даних завдання відносяться, як правило, до завдань керівного аналізу і стратегічного планування. Нижче наведені

приклади типових питань, на які можна відповідати за допомогою сховищ даних.

Фінансовий аналіз:

- Наскільки точно різні підрозділи компанії виконують встановлений бюджет?
- Які тенденції витрат за різними підрозділами, статтями бюджету?
- Наскільки вчасно надходять платежі?

Аналіз продажу:

- Наскільки виконаний план з продажу?
- Які ключові показники продуктивності компанії в поточному періоді?
- Які тенденції зміни ключових показників продуктивності компанії з часом?

Аналіз прибутковості:

- Які продукти (послуги) дають найбільший прибуток?
- Яка комбінація підрозділів і товарів (послуг) просуває бізнес?
- Які сегменти ринку дають найбільший прибуток?
- Які клієнти забезпечують найбільший прибуток?
- Який набір продуктів купують найприбутковіші клієнти?

Аналіз каналів продажу:

- Хто з торгових представників краще за всіх продає товари (послуги)?
- Як змінюється ціна на товар (послугу) в різних філіалах компанії?
- Які з партнерів забезпечують найбільший прибуток?
- Які продукти, групи продуктів найкраще продає певний партнер?
- Які тенденції зміни обсягів продажу через партнерів?

Аналіз клієнтської бази:

- Які сегменти ринку забезпечують найбільший прибуток?
- Які клієнти дають найбільший прибуток?
- Які властивості характерні клієнтам, що забезпечують найбільший прибуток?

Маркетинг:

- Яка вірогідність відгуку певного сегменту ринку на нову пропозицію?
- Яка вартість проведення маркетингової кампанії через заданий канал?
- Який канал проведення маркетингових кампаній є найефективнішим?

Аналіз якості обслуговування клієнтів:

- Яка вартість одного контакту з клієнтом?
- Наскільки клієнти задоволені якістю товарів (послуг)?
- Як задоволеність клієнтів змінюється з часом?

- Які продукти постачають вчасно, які - із запізненням?
- Чи мають певні клієнти або продукти неприпустимо довгий термін постачання?
- Наскільки швидшим або повільнішим стало постачання продуктів (послуг) в певний сегмент ринку?
- Які основні причини відмови від продукту (послуги)?
- Які клієнти найближчим часом, можливо, перестануть користуватися послугами компанії?

Аналіз складських запасів:

- Яка динаміка зміни запасу певного продукту?
- Яка оборотність складу?
- Скільки коштує зберігання певного продукту на складі?

Аналіз постачальників:

- Які з постачальників пропонують якнайкраще співвідношення ціна/ якість?
- Які з постачальників постачають товари швидше за інших?

Аналіз персоналу:

- Яка продуктивність персоналу, який навчався, в порівнянні з тими, хто не навчався?
- Які тенденції щорічного зростання персоналу компанії в різних регіонах, підрозділах?
- Яким набором навиків повинен володіти співробітник, щоб добре виконувати свої обов'язки?

1.3.3. Характеристика даних у сховищі

Всі дані у сховищі даних діляться на три основні категорії:

- метадані;
- детальні дані;
- агреговані дані.

Деталізовані і агреговані дані

Дані в ХД зберігаються як в деталізованому, так і в агрегованому вигляд . Дані в деталізованому вигляді надходять безпосередньо з джерел даних і відповідають елементарним подіям, що реєструється OLTP-системами. Такими даними можуть бути щоденні продажі, кількість вироблених виробів і т.д. Це неподільні значення, спроба додатково деталізувати які позбавляє їх логічного сенсу.

Багато задач аналізу (наприклад, прогнозування) вимагають використання даних певною мірою узагальнення. Наприклад, суми продажів, взяті по днях, можуть дати дуже нерівномірний ряд даних, що ускладнить виявлення характерних періодів, закономірностей або тенденцій. Однак, якщо узагальнити ці дані в межах тижня або місяця і взяти суму, середнє, максимальне і мінімальне значення за відповідний період, то отриманий ряд може виявитися більш інформативним. Процес узагальнення деталізованих даних називається агрегуванням, а самі великі цифри – агрегованими (іноді – агрегатами). Зазвичай агрегування піддаються числові дані (факти), вони обчислюються і містяться в СД разом з деталізованими даними.

Оскільки один і той же набір деталізованих даних може породити декілька наборів агрегованих даних з різним ступенем узагальнення, обсяг СД зростає, іноді істотно. Наприклад, набір, що містить дані про продажі по днях протягом року, крім своїх 360 значень, породжує 52 значення з узагальненням по тижнях і 12 – по місяцях. Якщо при цьому обчислюються всі види агрегації – сума, середнє, максимальне і мінімальне значення за відповідний період, – то кількість збережених агрегованих значень складе вже (52 + 12) • 4 = 256. Іноді це призводить до «вибухового», неконтрольованого зростання СД і викликає серйозні технічні проблеми: сховище «розпухає», через те що безперервний потік вхідних даних автоматично агрегується відповідно до налаштувань СД. Однак з цим доводиться миритися: якби агреговані дані не містилися в СД, а обчислювалися в процесі виконання запитів, час виконання запиту збільшився б в кілька разів.

Метадані та їх роль у **сховищі даних**

Метадані в широкому сенсі необхідні для опису значення і властивостей інформації з метою кращого її розуміння, використання і управління нею. ***Метадані*** (від грецьк. Meta і лат. Data), буквально перекладається як «дані про дані», – інформація про інший набір даних.

Одне з корисних визначень наступне: «Метадані – це структуровані, кодовані дані, які описують характеристики об'єктів-носіїв інформації, що сприяють ідентифікації, виявленню, оцінюванню, керуванню цими об'єктами» [25]. Майкл Брекет (Michael Brackett) визначає метадані (які він називає «даними про ресурси даних») як «будь-які дані про інформаційні ресурси організації». Адрієн Танненбаум (Adrienne Tannenbaum) називає метадані «детальним описом сутності даних». Ці визначення розкривають формулювання «дані про дані».

Тема ця підіймається відтоді, як існують дані: метадані були необхідні для опису значення і властивостей інформації з метою кращого її розуміння, керування і використання. Будь-яка людина, яка читала книги або користувалась бібліотекою, в тій чи іншій мірі мала справу з метаданими

Приклад 1.1.

Всім добре відомо, що в будь-якій книзі, крім власне тексту, міститься значна кількість додаткової інформації. Мета її полягає в тому, щоб, по-перше, допомогти читачеві швидше ознайомитися з вмістом книги і осмислити його, по-друге, описати структуру книги для більш ефективного пошуку потрібної інформації. Для вирішення першого завдання служать такі елементи, як анотація, коментарі, глосарій, примітки і т.д. Для пошуку потрібної інформації використовуються зміст, назви розділів і параграфів, номери сторінок, колонтитули, предметний покажчик і т.д. Крім цього, читачеві можуть знадобитися відомості про авторів або про видавництво. Вся ця інформація, яка не є частиною книги, а служить для підвищення ефективності роботи з нею, і являє собою метадані. У бібліотеці метадані застосовуються для пошуку потрібних видань і відстеження їх переміщень, наприклад, систематичний або алфавітний каталоги, в яких використовуються назви книг, прізвища авторів, рік видання і т.д. Таким чином, метадані мають дуже велике значення при роботі з різного роду інформацією.

Зазвичай під метаданими розуміється будь-яка інформація, необхідна для аналізу, проектування, побудови, впровадження і застосування в комп'ютерній системі. У випадку інформаційних систем метадані особливо спрощують керування, створення запитів, повноцінне використання і розуміння даних.

Одне з основних призначень метаданих – підвищення ефективності пошуку. Пошукові запити, які використовують метадані, роблять можливим виконання складних операцій по фільтрації та відбору даних.

Якщо розглядати поняття «метадані» в контексті технології СД, то його можна визначити наступним чином.

Метадані – високорівневі засоби відображення інформаційної моделі та опису структури даних, використовуваної в СД. Метадані повинні містити опис структури даних сховища і структури даних імпортованих джерел. Метадані зберігаються окремо від даних в так званому репозиторії метаданих

Багато недавніх проектів, як наукові, так і практичні, напрямлені на вивчення метаданих. Генерування, зберігання і керування метаданими допомагають в підтримці використання величезних обсягів інформації, доступних в наші дні в будь-якій електронній формі. Оскільки все, з чим

працює комп’ютер, за суттю є даними, і свого роду метадані супроводжують будь-які дані, то це поняття дотичне до будь-якої сфери застосувань і набуває різних форм залежно від застосування.

Метадані є ключовим фактором успіху при розробці та впровадженні СД. Вони містять всю інформацію, необхідну для отримання, перетворення і завантаження даних з різних джерел, а також для подальшого використання та інтерпретації даних, що містяться в СД.

Можна виділити два рівня метаданих – технічний (адміністративний) і бізнес-рівень. Технічний рівень містить метадані, необхідні для забезпечення функціонування сховища (статистика завантаження даних і їх використання, опис моделі даних і т.д.). Бізнес-метадані забезпечують користувачеві можливість концентруватися на процесі аналізу, а не на технічних аспектах роботи з сховищем; вони включають бізнес-терміни і визначення, якими звик оперувати користувач.

Фактично бізнес-метадані являють собою опис предметної області, для роботи в якій створюється аналітична система або СД. До формування бізнес-метаданих повинні активно залучатися експерти і аналітики, які згодом і будуть використовувати систему для отримання аналітичних звітів.

Бізнес-метадані описують об'єкти предметної області, інформація про яких міститься в СД, – атрибути об’єктів і їх можливі значення, відповідні поля в таблицях і т.д. Бізнес-метадані утворюють так званий семантичний шар. Користувач оперує близькими йому термінами предметної області: товар, клієнт, продаж, покупки і т.д., а семантичний шар транслює бізнес-терміни в низькорівневі запити до даних в сховищі.

Метадані систем сховищ даних іноді розділяють на два типи:

1) службові метадані, що використовуються для функцій витягання, перетворення і завантаження, для перенесення інформації з транзакційних систем у сховищі;
2) інтерфейсні метадані, що використовуються для опису екранів і створення звітів.

Ральф Кімболл (Ralph Kimball) перераховує наступні типи метаданих у сховищі:

метадані початкової системи:

- специфікації джерел даних, таких як репозиторії;
- описова інформація (наприклад, частота оновлення, юридичні обмеження і методи доступу);

- інформація про процеси, такі як графік завдань і коди витягання;

метадані перетворення даних:

- інформація про отримання даних (наприклад, планування передавання даних і результатів, а також відомості про використання файлів);
- керування таблицями вимірів, наприклад, визначення вимірів і присвоєння сурогатних ключів;
- перетворення і агрегація, наприклад, розширення і відображення даних, програми (скрипти) завантаження СКБД, визначення агрегатів даних;
- документування перевірок, робіт і журналів, наприклад, журналів перетворення даних і записів стеження за походженням даних;

метадані СУБД, такі як:

- зміст системних таблиць СКБД;
- рекомендації з опрацювання.

У загальному випадку, для користувача сховища даних потрібні метадані, принаймні, наступних типів.

1. Описи структур даних, їх взаємозв'язків.
2. Інформація про дані, що зберігаються у сховищі, і підтримувані ним агрегати даних.
3. Інформація про джерела даних і про міру їх достовірності. Одна і та ж інформація могла потрапити у сховище даних з різних джерел. Користувач повинен мати можливість взнати, яке джерело було вибране основним, і яким чином здійснюється узгодження і очищення даних.
4. Інформація про періодичність оновлень даних. Бажано знати не тільки те, якому моменту часу відповідають необхідні для користувача дані, але і коли вони наступного разу будуть оновлені.
5. Інформація про власників даних. Користувачу системи підтримки прийняття рішень може виявитися корисною інформація про наявність в системі даних, до яких він не мас доступу, про власників цих даних і про дії, які він повинен зробити, щоб дістати доступ до даних.
6. Статистичні оцінки часу виконання запитів. До виконання запиту корисно мати хоч би приблизну оцінку часу, який буде потрібний для отримання відповіді, і обсягу цієї відповіді.

Найкраще можна пояснити сутність метаданих, описуючи їх роль і призначення в реалізації процесів сховища даних. Метадані можна використовувати трьома способами:

а) пасивно, забезпечуючи чітку документацію про структуру, процес розроблення і використання системи СД; доступна документація необхідна всім учасникам (тобто кінцевим користувачам, системним адміністраторам, а також розробникам застосувань);

б) активно, шляхом зберігання конкретних семантичних аспектів (наприклад, правил перетворення) у вигляді метаданих, які можна інтерпретувати і використовувати під час виконання; у цьому випадку процеси сховища даних керуються метаданими. А отже, код (тобто активні метадані) і додаткова документація погоджено і уніфіковано керуються в одному репозиторії, при цьому актуальність документації зростає;

в) напівактивно, за рахунок зберігання статичної інформації (наприклад, визначень структур, специфікацій конфігурацій), яку прочитуватиме інший програмний компонент під час виконання; наприклад, обробникам запитів необхідні метадані для перевірки існування атрибутів; на відміну від активного використання, тут метадані тільки читаються, але не виконуються.

Створення і керування метаданими служить двом цілям:

1) мінімізації робіт з розроблення і адміністрування СД;
2) ефективнішому витяганню інформації з СД.

Перша мета, в основному, стосується:

підтримки інтеграції систем:

- схеми й інтеграція даних залежать від метаданих, що описують структуру і сенс окремих джерел даних і цільових систем;
- правила перетворення можна застосувати до початкових даних і зберігати як метадані;
- інтеграція різних інструментів можлива тільки тоді, коли вони розділяють «дані», які в такому випадку є метаданими системи сховища даних;

підтримки аналізу і проектування нових застосувань:

- метадані підвищують контрольованість і надійність процесу розроблення застосувань, забезпечуючи інформацію про сенс даних, їх .структуру і джерела;
- метадані, що стосуються рішень з проектування застосувань, можна використовувати повторно;

***підвищення гнучкості системи і можливості повторного використання наявних програмних модулів*:**

- це можливо тільки для активного і напівактивного використання метаданих; семантичні аспекти, що швидко змінюються, явним чином зберігаються у вигляді метаданих поза прикладними програмами;
- підтримка істотно простіша;
- систему можна розширити і адаптувати без жодних труднощів;
- цей підхід також дає можливість повторного використання «фрагментів коду»;

***автоматизації адміністративних процесів*:**

- метадані керуються запуском різних процесів СД (наприклад, завантаження і оновлення);
- інформація про їх виконання (журнали доступу, кількість доданих у сховищі записів і т.ін.) також міститься в репозиторії, легко доступному адміністратору;

посилення механізмів безпеки:

- метадані повинні забезпечити правила доступу і призначені для користувача права для всієї системи СД;
- керування доступом в сховищі даних іноді вимагає застосування складних методів; наприклад, оперативне джерело може містити нешкідливу інформацію про окремі показники роботи компанії, проте, сумарні значення у сховищі іноді виявляються найважливішим секретом; з другого боку, персональні доходи кожного співробітника є таємницею, але при цьому підсумкова сума зарплат в СД може зовсім не бути критичною інформацією.

Друга мета відноситься до ефективного витягання інформації, а точніше до:

1) підвищення якості даних (якість даних визначається наступними характеристиками):
   а) узгодженістю (чи є подання даних однорідним, чи немає дублікатів, даних з пересічними або конфліктними визначеннями);
   б) повнотою (чи всі дані присутні);
   в) точністю (збігом значень, що зберігаються, і фактичних);
   г) своєчасністю (чи актуальне значення, що зберігається у сховищі);
2) правила перевірки якості даних;
3) поліпшення взаємодії усередині системи сховища даних;
4) поліпшення аналізу даних;
5) застосування загальної термінології і мови взаємодії усередині корпорації.

Правила перевірки якості даних необхідно задати, зберегти у вигляді метаданих і перевіряти при кожному оновленні сховища даних. Крім того, висока якість вимагає підтримки контролю даних. Метадані забезпечують інформацію про час створення і про автора даних, про джерело, значення даних у момент отримання (про спадковість даних), і про подальший шлях від джерела до поточного місцеперебування (data lineage – про походження даних). Отже, користувачі можуть відновити ланцюжок, яким рухаються дані за час перетворення, і перевірити точність поверненої інформації;

Поліпшення взаємодії усередині системи сховища даних. Взаємодія відбувається як за допомогою виконання простих запитів і звітних застосувань, так і з використанням складних аналітичних інструментів. Метадані забезпечують відомості про значення даних, термінологію і бізнес-концепції підприємства, а також їх зв'язок з даними. Тому метадані підвищують якість запитів за рахунок точнішого і строгого формулювання, а також скорочують витрати на користувачів, яким необхідний доступ, оцінка і застосування відповідної інформації;

Поліпшення аналізу даних. Методи аналізу даних подані широко – починаючи від простих застосувань звітності та систем підтримки прийняття рішень і закінчуючи складними застосуваннями видобування даних. У цьому напрямі метадані необхідні для розуміння предметної області і її подання у сховищі з тим, щоб адекватно застосувати та інтерпретувати результати;

Застосування загальної термінології і мови взаємодії усередині корпорації. Доступність метаданих як унікального джерела документації для користувачів має і інші переваги. Вона гарантує узгоджені засоби взаємодії й інтерпретації інформації зі сховища, а також усуває двозначність і забезпечує узгодженість відомостей усередині компанії, дозволяє розділяти знання і досвід.

Метадані системи сховища даних містяться в репозиторії – структурованій системі зберігання і витягання, реалізованій на основі СУБД. Для інтерпретації метаданих необхідно зберігати структуру репозиторію (тобто схему метаданих) і їх семантику.

Формалізація метаданих.

Для інформації про дані сховища доцільно застосовувати шестивимірну класифікаційну схему Захмана (відповідно до відповідей на запитання що? хто? де? коли? чому? як?):

- об'єкти (що?);
- суб'єкти (хто?);
- місцезнаходження (де?);

- час (коли?);
- фактори впливу, чинники (чому?);
- способи (як?).

Таблиця. 1.2 – Модель Захмана в контексті побудови метаданих сховища даних

	Мотивація	Люди	Дані	Функції	Місце	Час
Контекст	Мета і стратегія побудови СД	Люди, що використовують СД	Об'єкти, які моделює СД	Основні бізнес-процеси	Географія СД	Періоди завантаження даних
Модель СД	Дії, які виконуються над даними	Моделі потоків робіт	Семантичні моделі	Моделі бізнес-процесів	Система логістики	Базовий графік робіт
Системна модель	Моделі бізнес-правил	Архітектура інтерфейсу користувача	Концептуальна модель даних	Архітектура застосувань	Архітектура розподіленої системи	Структура опрацювання подій
Технологічна модель	Моделі правил опрацювання подій	Архітектура подання	Фізична модель даних	Архітектура програмно-апаратної системи	Технологічна архітектура	Структура циклів керування
Детальне подання	Специфікації правил роботи системи	Специфікації ролей та прав доступу	Специфікації форматів даних	Код програмних компонент	Специфікації архітектури мережі	Специфікації опрацювання подій і переривань
Робоче СД	Стратегія і практика	Структура СД	Дані	Функції, що виконуються	Географічне положення і мережі	Плани

При цьому використовується наступна формалізація:

- сутність або вміст сховища даних;
- люди, які використовують сховище даних;
- місцерозташування даних, важливе з погляду керування сховищем даних;
- моменти завантаження даних і обчислення підсумкових таблиць;
- рушійні сили створення і розвитку сховища даних;
- дії, які виконуються з даними;
- повчальні метадані (як новий чинник, що використовуватиметься для підтримки розвитку моделі сховища даних).

Метадані зберігаються в окремій базі даних метаданих або репозиторії. Системне програмне забезпечення для створення репозиторію пропонується рядом компаній, в першу чергу, розробниками СУБД.

Важливість цього ключового аспекту сховища даних можна продемонструвати у різних аспектах:

- єдині правила найменування об'єктів;
- єдині одиниці вимірювання для однотипних об'єктів;
- єдине фізичне подання однотипних об'єктів;
- єдині атрибути подання однотипних об'єктів, тощо.

Приклад 1.2. Наведемо приклад заповнення метаданих за класифікацією Захмана.

Метадані про об'єкти

Код	Назва сутності	Таблиці	Атрибути
1	Дата продажу	Вимір_дата	ID, Дата. Квартал
2	Товар	Вимір_товар	ID, Назва товару, Категорія товару, Одиниці виміру
3	Регіон продажу	Вимір_регіон	ID, Назва регіону, Категорія регіону
4	Покупець	Вимір_ покупець	ID, Назва покупця, Категорія, Адреса, Телефон, E-mail
5	Продаж	Продаж_факти	ID, ДатаID, РегіонID, ТоварID, ПокупецьID, Кількість товару, Ціна

Метадані про особи

№	Особа	Прізвище, ініціали	Функції	Контакт
1	Системний адміністратор	Різник В.В.	Обслуговування технічних та програмних засобів	кімн. 1, тел. 1112211
2	Програміст	Петрушенко Л.Г.	Розроблення та супровід програмного забезпечення	кімн. 2, тел. 1123212
3	Адміністратор баз даних	Василенко К.Д.	Підтримка та супровід баз даних сховища	кімн. 3, тел. 11122213

Приклад метаданих про місцезнаходження елементів сховища даних

№	Елемент	Шлях	ПЗ
1	База даних сховища	D:\DataBase\DWH.dbo	SQL Server 2005
2	Система аналізу даних відділу збуту	H:\DataMarts\Market.mdb	MS Office 2005

Метадані про час

№	Подія	Час виконання	Тривалість	Періодичність
1	Завантаження нових даних до СД	1 число місяця	1 день	щомісяця
2	Оновлення систем аналізу даних	3 число місяця	2 год	щоквартально
3	Архівування застарілих даних	10 січня	1 год	щорічно

Метадані про чинники сховища даних

№	Назва фактору	Відповідальний	На що впливає
1	Угода на виконання досліджень для створення СД	фірма *Моя фірма1*	проект сховища даних
2	Технічний проект сховища даних	фірма *Моя фірма2*	реалізація сховища даних
3	Регламент формування аналітичних звітів	керівник аналітичного департаменту	БД відділів

Метадані про способи виконання операцій

№	Операція	Способи виконання	Засоби виконання	Примітки
1	Завантаження оперативних даних	ETL	Oracle ETL Manager	D: \DataWarehouse\ Progs\ Oracle\EtlManager.exe
2	Формування вітрин даних	Агрегування, аналіз, збереження	Aggregation Express, Oracle Express Server	
3	Формування регламентованих звітів	Виклик прикладної програми	Prepared Report Manager	D:\Data\Varehouse\Progs\ Aplication\ReptManager.exe

XML і метадані

XML у наш час охоплює практично всі аспекти інформаційних технологій. Що стосується метаданих, то переоцінити використання мови XML тут складно, вона розповсюджується на безліч застосувань, у тому числі і на сховища даних.

Основна функція XML – визначати інші мови розмітки. XML - це метамова, а тому вона виявляється дуже ефективним форматом подання і обміну метаданими.

XML має безліч переваг, які роблять її ідеальним засобом опису.

1. Вона зрозуміла людям в читанні і описі. А отже, доступна новачкам і не викликає страху.
2. Це відкрита технологія. Стандарт XML запропонований W3C. Ніхто не має прав власності на цю мову. Вона незалежна.
3. XML може застосовуватися повсюдно. Аналізатор XML можна знайти скрізь, і, використовуючи відповідні інструменти, нескладно відразу ж впровадити цю технологію.
4. Мова гнучка. Мабуть, одна з головних причин використання XML у тому, що немає чітких рамок застосування. Кожен самостійно вирішує, як використовувати її в своєму застосуванні. XML недорога для впровадження як у великій, так і в малій організації.

Можна навести інші причини використання XML, а не інших засобів. У першу чергу, структура метаданих часто буває складною, в ній є множина вкладених відношень, а деякі елементи метаданих можуть повторюватися. По-друге, якщо для зберігання метаданих використовується, наприклад, РСКБД

(реляційна система керування базою даних), то таблиці в базі не відображають складних зв'язків між елементами метаданих (важко згенерувати визначення таблиць для опису відношень). І навпаки, XML задає структуру документа «самоописовим» чином. її можна використовувати для завдання не тільки змісту, але і схеми. А отже, не складно знайти взаємозв'язок між різними ділянками XML-документа.

XML дозволяє публікувати метадані, що використовуються будь-якою програмою або базою даних, у вигляді мови спілкування. XML забезпечує зв'язок між структурованою базою і неструктурованим текстом, що передається в форматі XML. Оскільки XML дозволяє задавати свою власну мову розмітки, то можна використовувати всі розширені гіпертекстові можливості для зберігання самих метаданих або посилань в будь-якому форматі.

Якщо є програмне забезпечення, яке може прочитати і розшифрувати XML-файли, то метадані в будь-якому сховищі можна подати у вигляді звичного XML-файла, створеного на основі загального DTD (*document type definition* – опис типу документа).

Очевидно, що XML стає все популярнішою, оскільки вирішує задачі зберігання і доступу до метаданих. Багато хто прагне до створення застосувань керування метаданими за принципом повторного використання і забезпечення активного застосування схем і DTD. Всім відомо, що необхідно створювати стандарти і визначення даних, що класифікуються за функціональністю об'єктів проблемної області.

Проте, хто ж вирішуватиме ці задачі? У більшості організацій програмісти, дизайнери, інтегратори і менеджери проектів «переступають» через XML-технологію і навіть не згадують про те, що її можна використовувати для керування ресурсами даних. Не варто дивуватися, якщо раптом в одному з XML-файлів, що описують метадані, виявляться проблеми: один і той самий атрибут описується в різних місцях по-різному, використовуються різні стандарти іменування полів, не узгоджено формати даних.

А що буде, якщо таких XML файлів виявиться 1000, причому всі вони будуть написані відповідно до різних стандартів?

Для досягнення максимальної віддачі, використання метаданих в рамках проекту повинне плануватися на етапі створення моделі майбутньої системи. При цьому бажано розробити єдину концепцію використання метаданих для всієї системи, а не для окремих її фрагментів. Це дозволить забезпечити інтероперабельність метаданих і уникнути надмірності подання даних.

Детальні дані.

Оскільки сховище даних отримує інформацію із зовнішніх джерел, то у ньому містяться детальні дані, що описують різні об'єкти проблемної області. Ці дані використовуються в подальшому для підтримки прийняття рішень та для агрегування (скорочення кількості вимірів та використання функцій групування).

У процесі експлуатації необхідність у ряді детальних даних може значно зменшитись, що є причиною поділу детальних даних на поточні і застарілі. Тоді як поточні дані регулярно використовуються і тому зберігаються на накопичувачах зі швидким доступом (в основному на жорстких дисках), застарілі детальні дані можуть зберігається на більш ємних накопичувачах з повільні¬шим доступом (наприклад, на магнітних стрічках).

Проведені дослідження показали, що більшість кінцевих користувачів не працюють з детальними даними, а, в основному, з агрегованими показниками. Структура СД відображає цю ситуацію і дозволяє кінцевому користувачу швидко і зручно одержувати необхідну йому агреговану інформацію з подальшою навігацією за всіма рівнями агрегації.

Ще однією особливістю даних в СД є те, що вони є денормалізованими, в порівнянні з нормалізованими даними в більшості БД, які зазвичай реалі¬зовані в реляційній моделі. Ця властивість СД дозволяє істотно підвищити швидкість доступу до необхідних даних, хоч і вимагає більшої місткості носіїв інформації.

Наявність добре розвиненої ієрархії агрегованих даних за рівнями агрегації є відмінною рисою сховища даних.

Агреговані дані.

Агрегація даних – це обчислення узагальнених показників для підтримки стратегічного або тактичного керування з детальних даних. Наприклад, всі записи про продаж двохсот тисяч найменувань товарів тисячі оптовим покупцям за кожен день року перетворяться в дані про продаж десяти категорій товарів п'яти категоріям покупців в розрізі місяців і кварталів року і регіонів. Ці дані використовуються згодом менеджерами для прийняття рішень про зміни напрямів бізнесу, розширення ринку, аналізу сезонних коливань попиту на товари різних категорій.

Попередній розрахунок агрегованих показників застосовується для того, щоб керівник одержував відповіді на подібні запити гранично швидко. У той же час у сховищі збираються максимально детальні дані, що дозволяє будувати

звіти в довільних аналітичних розрізах, обчислюючи агрегати у міру виникнення в них потреби (рис. 1.6).

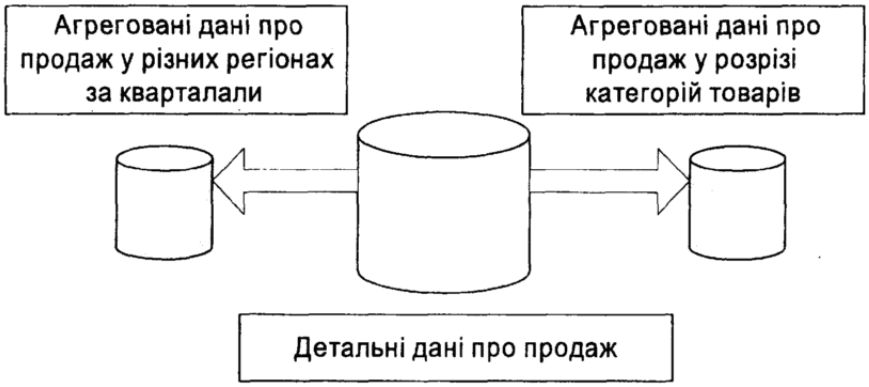


Рис. 1.6 – Попередня агрегація за різними розрізами

Агрегатами називають агреговані за певними умовами початкові значення показників. Зазвичай під агрегацією розуміється будь-яка процедура формування меншої кількості значень (агрегатів) на підставі більшої кількості початкових значень. Надалі під термінами агрегація і агрегування розумітимемо виключно процес підсумовування даних. Завчасне формування і збереження агрегатів з метою зменшення часу відгуку на призначений для користувача запит є основною властивістю систем підтримки оперативного аналізу.

Зв'язок між агрегованими та деталізованими даними подано на рис. 1.7.

На рисунку використано наступні позначення.

- Дуже рідке використання – кілька разів на рік (наприклад, для складання щоквартальних та річних звітів).
- Порівняно рідке використання – кілька разів у квартал (наприклад, для складання місячних звітів).
- Часте використання – кілька разів на місяць (наприклад, для складання так званих «п’ятиденок», аналізу даних за тиждень тощо).
- Дуже часте використання – декілька разів на день (наприклад, для визначення сум продажів за день, кількості клієнтів тощо).
- Сильно індексовані дані – в одному вимірі є декілька атрибутів-індексів, за такими даними часто виконуються операції пошуку (це дані з традиційних баз даних).
- Слабо індексовані дані – в одному вимірі є нуль або більше атрибутів-індексів, які найчастіше використовуються для агрегування у запитах.

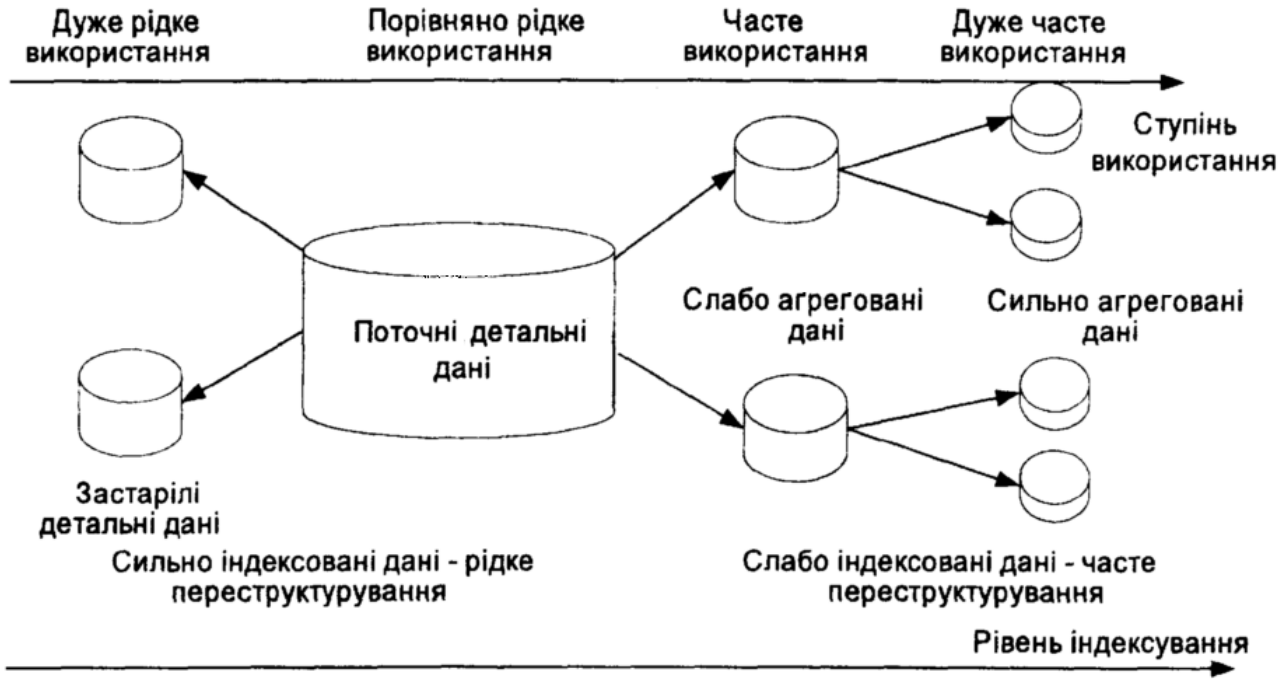


Рис. 1.7 – Зв’язок між детальними та агрегованими даними в СД

Окрім самих даних, сховище даних містить інші складові, які забезпечують його комплексну роботу. Розглянемо ці складові частини.

1.4. Складові сховища даних

1.4.1. Учасники СД

Всі учасники діляться на три основні групи:

а) кінцеві користувачі;

б) група розвитку;

в) група підтримки .

Оскільки технологія сховища даних реалізується шляхом поетапного впровадження проектів використання інформації в окремих проблемних областях, то всі групи учасників можуть працювати паралельно, взаємодіючи один з одним.

Кінцеві Користувачі

Виділяють три типи кінцевих користувачів:

- аналітики;
- середня ланка керівних працівників;
- вищий ешелон керівництва (найвища ланка).

Найінтенсивніше використовують дані за всіма рівнями агрегації аналітики. В їх завдання входить глибоке і ретельне дослідження даних із

застосуванням всіх доступних засобів аналізу. Рішення таких завдань супроводжується серйозним вивченням змістовної наповненості СД (тобто як детальних, так і агрегованих даних), а також побудовою додаткових структур даних. Використовуючи засоби типу систем підтримки прийняття рішень аналітики в процесі досліджень шукають закономірності між даними і подають результати у найзручнішому для використання вигляді. Отже, аналітики повинні володіти не тільки методами дослідження проблемної області, але і мати уявлення про СД, а також володіти інструментами видобування даних. У взаємодії з групою розвитку аналітик може надати неоціненну допомогу в розробленні додаткових структур даних в СД.

Середня ланка керівних працівників є відповідальною за підготовку рішень на рівні свого підрозділу і тому використовує дані СД для інформаційної підтримки формування рішень. Цей тип кінцевих користувачів рідко використовує деталізовані дані, зосереджуючи увагу на слабо і сильно агрегованих даних. У завдання керівників входить також формулювання напрямів дослідження аналітиків. Інструментами роботи з даними зазвичай є стандартні системи підтримки прийняття рішень (СППР), налаштовані на використання в підрозділі, і спеціалізовані застосування. При взаємодії з групою розвитку допомагають формулювати напрями подальшого розвитку СД.

Вищий ешелон керівництва, в основному, використовує сильно агреговані дані за основними показниками, що відображають діяльність організації в цілому для прийняття стратегічних рішень, застосовуючи, в основному, спеціалізовані застосування у вигляді інтерактивних звітів.

Група Підтримки.

У групі підтримки СД виділяються наступні функціональні ролі:

- адміністратор СД;
- підтримка користувачів.

Функції адміністратора СД значно відрізняються від функцій адміністратора БД. Функції адміністратора СД орієнтовані на підтримку якості даних для аналізу і підтримки прийняття рішень, тоді як функції адміністратора БД орієнтовані на технічну підтримку БД.

Основні інструменти адміністратора СД – монітор завантаження і монітор використання даних.

Контролюючи інформацію про завантаження даних в СД, адміністратор СД стежить за виконанням регламенту завантаження даних, а також за інформацією автоматичного контролю якості завантажуваних даних, і, у разі

виникнення порушень, контактує з відповідальними за джерело даних особами. Відповідальними за джерело даних зазвичай призначаються керівники підрозділів, що збирають первинні дані.

Інформація про використання даних в СД використовується адміністратором при взаємодії з групою розвитку для підвищення ефективності і надійності роботи.

Підтримка кінцевих користувачів передбачає розроблення методик і рекомендацій щодо використання засобів аналізу, навчання фахівців, а також допомогу в рішенні проблем, що виникають у користувачів.

Група Розвитку

У групі розвитку виділяються наступні функціональні ролі:

- постановник завдань;
- проектувальник даних;
- системна підтримка;
- розробник застосувань.

Постановник завдань досліджує інформаційні потреби організації, доступні джерела інформації, виділяючи проблемні області діяльності організації і формулюючи напрями і завдання розвитку СД. У його обов'язки входить також дослідження потреб в засобах аналізу і постановка завдань на розроблення спеціалізованих застосувань.

Тісна взаємодія з кінцевими користувачами і адміністратором СД є необхідною вимогою до постановника завдань. Тільки у цьому випадку кінцеві користувачі дістануть доступ до необхідної їм інформації за допомогою найефективніших засобів.

Основним завданням проектувальника даних є створення логічної структури СД, що забезпечує ефективний доступ до всіх необхідних даних. Вирішуючи поставлене постановником завдання, проектувальник даних вирішує наступні проблеми:

- опис структури детальних даних;
- опис структури агрегованих даних;
- опис підтипів сховищ даних (залежних сховищ даних);
- опис регламенту і процедур завантаження, трансформації, контролю і очищення даних.

Вирішення цих проблем неможливе без тісної співпраці з групою підтримки або розроблення БД, джерелами даних, що є. При розробленні логічної структури СД велику цінність може мати інформація про використання даних наявного сховища, одержана у адміністратора БД.

Системна підтримка СД передбачає визначення і рішення технічних проблем створення і розвитку СД. Сюди входить вибір платформи, ОС, необхідних вимог щодо пам'яті і дискового простору, забезпечення безпеки даних та ін.

При дослідженні предметної області може з'ясуватися, що для використання інформації, що зберігається, не досить набору стандартних засобів аналізу. У цьому випадку розробник застосувань займається реалізацією спеціалізованих застосувань аналізу, прогнозу і підтримки прийняття рішень. Зазвичай такі застосування будуються на основі простіших стандартних застосувань, але у ряді випадків вимагають окремого серйозного розроблення.

1.4.2 Програмне забезпечення

Програмне забезпечення СД ділиться на три основні категорії:

- засоби завантаження;
- засоби моніторингу;
- засоби створення і розвитку.

Засоби завантаження для виконання завантаження даних в СД використовують наступне ПЗ:

- диспетчер процесів;
- завантажувач даних;
- аналізатор даних.

1. *Диспетчер процесів*. Для нормального функціонування СД необхідне регулярне завантаження нових даних. Розроблення регламенту процесів завантаження даних з джерел є необхідною частиною побудови логічної структури СД. Диспетчер здійснює виконання процесів завантаження згідно з регламентом.

2. *Завантажувач даних*. Джерелами даних для СД можуть служити найрізноманітніші БД, а також зовнішні дані в інших форматах. Важливо, щоб завантажувач даних мав можливість доступу до максимальної кількості СКБД і інших форматів даних. У функції завантажувача входить також трансформація даних в заданий формат.

3. *Аналізатор даних*. Якість даних, отриманих з різних джерел, часто залишає бажати кращого. Автоматичний аналіз даних на коректність і несуперечність є важливою частиною технології СД. Дані, що не проконтрольовані, можуть привести до вибору неправильного рішення, і тому не повинні бути доступні кінцевим користувачам.

Деякі типи даних можуть опрацьовуватися для виявлення наперед невідомих залежностей і, у разі виявлення таких, створювати інформативні структури даних.

Засоби моніторингу

1. *Монітор завантаження*. Мета використання технології СД - надання достовірних даних для засобів аналізу, тому необхідним засобом є монітор завантаження даних, що збирає інформацію про виконання процесів завантаження даних і інформує адміністратора про хід цих процесів.

2. *Монітор використання даних*. Монітор використання даних є вельми корисним компонентом ПЗ для підвищення ефективності доступу до даних, а також може надавати інформацію про можливість перекладу поточних детальних даних в статус застарілих детальних даних.

Засоби створення і розвитку. Створення і розвиток сховища даних вимагає наступних компонентів ПЗ.

СКБД сховища даних. СКБД сховища даних повинна бути орієнтована на особливості технології СД – працювати з великими обсягами даних, забезпечувати необхідну безпеку даних, дозволяти створювати дуже складні структури даних (такі як багатовимірні бази даних), здійснювати швидкий, розрахований на багато користувачів, доступ до даних.

Засоби керування структурою даних СД. Для швидкої реалізації логічної структури даних необхідно мати зручний інтерактивний засіб керування структурою СД. Якість цього засобу визначає швидкість розроблення і розвитку СД, тому є дуже важливим чинником. Засіб використовується не тільки розробниками, але і кінцевими користувачами (аналітиками) для побудови своїх структур даних у спеціальних сховищах даних (розглянуто далі) і мусить мати зручний і зрозумілий інтерфейс.

Засоби завдання джерел даних. Служать для завдання джерел даних, що завантажуються у сховищі, визначення зв’язку між структурами СД і джерелами, створення процедур трансформації, очищення, автоматичного аналізу, завдання регламенту завантаження.

Засоби побудови спеціалізованих сховищ даних. Спеціалізовані сховища даних виконують функції певних підрозділів і є важливою частиною технології СД, і з розвитком СД часто необхідно переносити спеціалізовані СД на інше технічне устаткування, тому засіб повинен мати гнучкий інтерфейс роботи з спеціалізованими СД.

1.4.3. Параметри

Параметри складаються з двох компонентів:

1) чисельна характеристика факту, наприклад, ціна або дохід від продажу;
2) формула, зазвичай проста агрегатна функція, скажімо, сума, яка може об'єднувати декілька значень параметрів в одне.

У багатовимірній базі даних параметри, як правило, відображають властивості факту, який користувач хоче вивчити. Параметри набувають різних значень для різних комбінацій вимірювань. Властивість і формула вибираються так, щоб подавати осмислену величину для всіх комбінацій рівнів агрегації. Оскільки метадані визначають формулу, дані, на відміну від випадку електронних таблиць, не тиражуються.

При обчисленнях три різні класи параметрів поводяться абсолютно по-різному.

Адитивні параметри можуть змістовним чином комбінуватися в будь-якому вимірі. Наприклад, має сенс підсумовувати загальний обсяг продажу для продукту, місцезнаходження і часу, оскільки це не викликає накладання серед явищ реального світу, які генерують кожне з цих значень.

Напівадитивні параметри які не можуть комбінуватися в один або декілька вимірів. Наприклад, підсумовування запасів за різними товарами і складами має сенс, але підсумовування запасів товарів у різний час недоцільне, оскільки одне і те ж фізичне явище може враховуватися кілька разів.

Неадитивні параметри не комбінуються в будь-якому вимірі зазвичай тому, що вибрана формула не дозволяє об'єднати середні значення низького рівня в середньому значенні вищого рівня.

Адитивні і неадитивні параметри можуть описувати факти будь-якого роду, тоді як напівадитивні параметри, як правило, використовуються з миттєвими знімками або сукупними миттєвими знімками.

1.4.4. Запити

СД природним чином призначене для певних типів запитів. Оскільки одна з основних функцій сховища даних – аналіз даних, а не їх оперативне опрацювання, то запити до СД мають інший характер, ніж запити до БД.

Запити вигляду slice-and-dice. В основі концепції сховища даних лежать дві основні ідеї: здійснення вибору, зменшення куба. Наприклад, можна розглянути перетин куба, взявши до уваги тільки ті комірки, які стосуються хліба, а потім ще більше зменшити його, залишивши комірки, що відносяться тільки до 2006 року. Фіксація значення вимірів зменшує розмірність куба, але

при цьому можливі і загальніші операції.

Запити вигляду drill-down і roll-up – взаємообернені операції, які використовують ієрархію вимірів і параметри для агрегації. Узагальнення до вищих значень відповідає виключенню розмірності. Наприклад, згортка від рівня «Місто» до рівня «Країна» агрегує значення для Львова і Києва в одне значення – Україна.

Запити вигляду drill-across комбінують куби, які мають одне або декілька загальних вимірів. З погляду реляційної алгебри така операція виконує злиття (join).

Запити вигляду ranking повертає тільки ті результати, які з'являються у верхній або нижній частині впорядкованого певним чином списку, наприклад, 10 продуктів, що найкраще продаються у Львові в 2006 році.

Поворот (rotating) куба дає користувачам можливість побачити дані, згруповані за іншим виміром.

1.5 Огляд архітектури СД

Розробка і побудова корпоративного СД – це дорога і трудомістка задача. Успішність впровадження СД багато в чому залежить від рівня інформатизації бізнес-процесів в компанії, інформаційних потоків, що встановилися, обсягу і структури використовуваних даних, вимог до швидкості виконання запитів і частоті поновлення сховища, характеру розв’язуваних аналітичних завдань і т.д. Щоб наблизити СД до умов і специфіки конкретної організації, в даний час розроблено декілька архітектур сховищ – реляційні, багатовимірні, гібридні і віртуальні.

Реляційні СД використовують класичну реляційну модель, характерну для оперативних OLTP-систем. Дані зберігаються в реляційних таблицях, але утворюють спеціальні структури, адаптовані багатовимірне представлення даних. Така технологія позначається абревіатурою ROLAP – Relational OLAP (OnLine Analytical Processing – аналітична обробка в реальному часі).

Багатовимірні СД реалізують багатовимірне представлення даних на фізичному рівні у вигляді багатовимірних кубів. Дана технологія отримала назву MOLAP – Multidimensional OLAP.

Гібридні СД поєднують в собі властивості як реляційної, так і багатовимірної моделі даних. У гібридних СД деталізовані дані зберігаються в реляційних таблицях, а агрегати – в багатовимірних кубах. Така технологія побудови СД називається HOLAP – Hybrid OLAP.

Віртуальні СД не є сховищами даних в звичному розумінні. У таких системах робота ведеться з окремими джерелами даних, але при цьому емулюється робота звичайного СД. Інакше кажучи, дані не консолідуються фізично, а збираються безпосередньо в процесі виконання запиту. Крім того, всі СД можна розділити на одноплатформні і крос-платформні. Одноплатформні СД будуються на базі тільки одної СУБД, а крос-платформні можуть будуватися на базі кількох СУБД.

1.6 Багатовимірні сховища даних

Основне призначення багатовимірних сховищ даних (БСД) – підтримка систем, орієнтованих на аналітичну обробку даних, оскільки такі сховища краще справляються з виконанням складних нерегламентованих запитів.

Багатовимірна модель даних, що лежить в основі побудови багатовимірних сховищ даних, спирається на концепцію багатовимірних кубів, або гіперкубів. Вони являють собою впорядковані багатовимірні масиви, які також часто називають OLAP-кубами (абревіатура OLAP розшифровується як On-Line Analytical Processing – оперативна аналітична обробка). Технологія OLAP є методику оперативного вилучення потрібної інформації з великих масивів даних і формування відповідних звітів.

1.6.1 Основи багатовимірного представлення даних

Сутність багатовимірного представлення даних полягає в наступному. Більшість реальних бізнес-процесів описується безліччю показників, властивостей, атрибутів і т.д. Наприклад, для опису процесу продажів можуть знадобитися відомості про найменування товарів або їх груп, про постачальника і покупця, про місто, де проводилися продажу, а також про ціни, кількості проданих товарів і загальні суми. Крім того, для відстеження процесу в часі повинен бути введений в розгляд такий атрибут, як дата. Якщо зібрати всю цю інформацію в таблицю, то вона виявиться складною для візуального аналізу і осмислення. Більш того, вона може виявитися надмірною: якщо, наприклад, один і той же товар продавався в один і той же день в різних містах, то доведеться кілька разів повторити одну й ту саму відповідність «місто – товар» із зазначенням різних суми і кількості. Все це здатне остаточно заплутати і збити з пантелику будь-якого, хто спробує витягти з такої таблиці корисну інформацію з метою аналізу поточного стану продажів і пошуку

шляхів оптимізації процесу торгівлі. Зазначені проблеми виникають з однієї простої причини: в плоскій таблиці зберігаються багатовимірні дані.

Прояснимо суть питання за допомогою геометричної аналогії. Уявіть собі тривимірну фігуру (наприклад, тетраедр або паралелепіпед) і спроектуйте його на площину, а потім за отриманою плоскою проекції спробуйте оцінити форму і розміри вихідної об'ємної фігури. Зробити це буде важко: по-перше, втрачена інформація про одному вимірі, а по-друге, фігура тепер представлена в абсолютно невластивому їй плоскому вигляді.

Приблизно те ж саме можна сказати про інформацію, представлену декількома рядами даних. Кожен такий ряд (поле таблиці) можна розглядати як свого роду інформаційний вимір, і тоді «плоска» таблиця може бути інтерпретована як результат перетворення багатовимірної інформаційної структури в абсолютно невластиву їй плоску форму. Щоб компенсувати втрату інформації від виключення одного або декількох вимірювань, доводиться ускладнювати структуру таблиці, а це в більшості випадків призводить до того, що розібратися в ній стає дуже складно.

Можна піти іншим шляхом – виконати декомпозицію інформації в кілька простіших таблиць, зв’язати їх деяким набором відносин і перейти до реляційної моделі, яку використовують класичні бази даних. Однак доведено, що реляційна модель не є оптимальною з точки зору завдань аналізу, оскільки свідчить про високий ступінь нормалізації, в результаті чого знижується швидкість виконання запитів. Тому розробка багатовимірної моделі представлення даних, яка реалізується за допомогою багатовимірних кубів, стала природним кроком.

***Зауваження*.** Не слід намагатися провести геометричну інтерпретацію поняття «багатовимірний куб», оскільки це просто службовий термін, що описує метод представлення даних .

1.6.2 Вимірювання і факти – базові поняття багатовимірної моделі даних

В основі багатовимірного представлення даних лежить їх поділ на дві групи – вимірювання і факти.

Вимірювання – це категоріальні атрибути, найменування і властивості об'єктів, що беруть участь в деякому бізнес-процесі. Значеннями вимірювань є найменування товарів, назви фірм-постачальників і покупців, ПІБ людей, назви міст і т.д. Вимірювання можуть бути і числовими, якщо який-небудь категорії (наприклад, найменуванню товару) відповідає числовий код, але в будь-якому випадку це дані дискретні, тобто приймають значення з обмеженого набору.

Вимірювання якісно описують досліджуваний бізнес-процес.

Факти – це дані, кількісно описують бізнес-процес, безперервні за своїм характером, тобто вони можуть приймати безліч значень. Приклади фактів – ціна товару або виробу, їх кількість, сума продажу або закупівель, зарплата співробітників, сума кредиту, страхова винагорода і т.д.

1.6.3 Структура багатовимірного куба

Багатовимірний куб можна розглядати як систему координат, осями якої є вимірювання, наприклад Дата, Товар, Покупець. По осях будуть відкладатися значення вимірювань – дати, найменування товарів, назви фірм-покупців, ПІБ фізичних осіб і т.д.

У такій системі кожному набору значень вимірів (наприклад, «дата – товар – покупець») буде відповідати осередок, в якій можна розмістити числові показники (тобто факти), пов’язані з даними набором. Таким чином, між об'єктами бізнес-процесу і їх числовими характеристиками буде встановлений однозначний зв’язок.

Принцип організації багатовимірного куба пояснюється на рис. 1.8. В осередку 1 будуть розташовуватися факти, які стосуються продажу цементу ТОВ «Спецбуд» 3 листопада, в осередку 2 – продажу плит ЗАТ «Піраміда» 6 листопада, а в осередку 3 – продажу плит ТОВ «Спецбуд» 4 листопада.

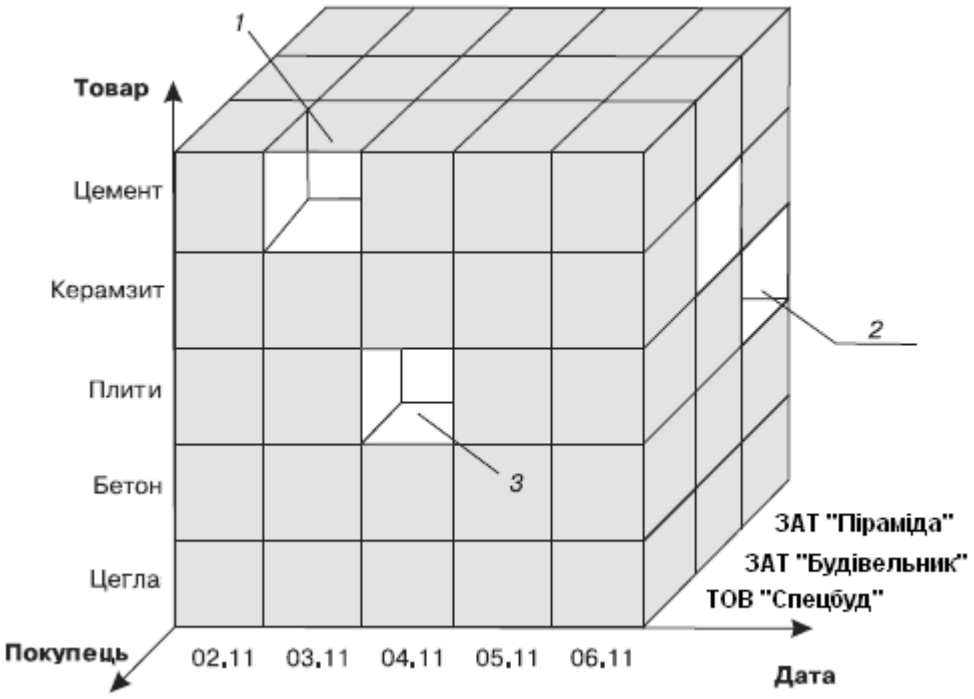


Рис.1.8 – Принцип організації багатовимірного куба

Багатовимірний погляд на вимірювання Дата, Товар і Покупець представлений на рис. 1.9. Фактами в даному випадку можуть бути Ціна, Кількість, Сума. Тоді виділений сегмент буде містити інформацію про те, скільки плит, на яку суму і за якою ціною придбала фірма ЗАТ «Будівельник» 3 листопада.

Рис.1.9 – Вимірювання і факти в багатовимірному кубі

Таким чином , інформація в багатовимірному сховищі даних є логічно цілісною. Це вже не просто набори строкових і числових значень, які в разі реляційної моделі потрібно отримувати з різних таблиць, а цілісні структури типу «кому, що і в якій кількості було продано на даний момент часу». Переваги багатовимірного підходу очевидні.

- Представлення даних у вигляді багатовимірних кубів наочніше, ніж сукупність нормалізованих таблиць реляційної моделі, структуру якої представляє тільки адміністратор БД.
- Можливості побудови аналітичних запитів до системи, що використовує БСД, більш широкі.
- У деяких випадках використання багатовимірної моделі дозволяє значно зменшити тривалість пошуку в БСД, забезпечуючи виконання аналітичних запитів практично в режимі реального часу. Це пов'язано з

тим, що агреговані дані обчислюються попередньо і зберігаються в багатовимірних кубах разом з деталізованими, тому витрачати час на обчислення агрегатів при виконанні запиту вже не потрібно.

В принципі, OLAP-куб може бути реалізований і за допомогою звичайної реляційної моделі. У цьому випадку має місце емуляція багатовимірного представлення сукупністю плоских таблиць. Такі системи отримали назву ROLAP – Relational OLAP.

Використання багатовимірної моделі даних пов’язане з певними труднощами. Так, для її реалізації потрібно більший обсяг пам'яті. Це пов’язано з тим, що при реалізації фізичної багатовимірності використовується велика кількість технічної інформації, тому обсяг даних, який може підтримуватися БСД, зазвичай не перевищує кількох десятків гігабайт. Крім того, багатовимірна структура важче піддається модифікації; при необхідності вбудувати ще один вимір потрібно виконати фізичну перебудову всього багатовимірного куба. На підставі цього можна зробити висновок, що застосування систем зберігання, в основі яких лежить багатовимірне представлення даних, доцільно тільки в тих випадках, коли обсяг використовуваних даних порівняно невеликий, а сама багатовимірна модель має стабільний набір вимірювань.

1.6.4 Робота з вимірами

У процесі пошуку і вилучення з гіперкуба потрібної інформації над його вимірами проводиться ряд дій , найбільш типовими з яких є :

- переріз (зріз);
- транспонування;
- згортка;
- деталізація.

Переріз полягає у виділенні підмножини комірок гіперкуба при фіксуванні значення одного або декількох вимірювань. В результаті перерізу виходить зріз або кілька зрізів, кожен з яких містить інформацію, пов’язану зі значенням виміру, за яким він був побудований. Наприклад, якщо виконати переріз за значенням ЗАТ «Будівельник» вимірювання Покупець, то отриманий в результаті зріз буде містити інформацію про історію продажів всіх товарів даного підприємства, яку можна буде звести в плоску таблицю. При необхідності обмежити інформацію тільки одним товаром (наприклад, керамзитом) буде потрібно виконати ще одне переріз, але тепер уже за

значенням Керамзит вимірювання Товар. Результатом побудови двох зрізів буде інформація про продажі одній фірмі по одному товару. Маніпулюючи таким чином перерізами гіперкуба, користувач завжди може отримати інформацію в потрібному розрізі. Потім на основі побудованих зрізів може бути сформована крос-таблиця і з її допомогою дуже швидко отриманий необхідний звіт. Дана методика лежить в основі технології OLAP-аналізу.

На рис.1.10 схематично представлені перерізи гіперкуба. Зліва переріз виконано при деякому фіксованому значенні виміру Дата. Отриманий зріз (світло-сіра область) містить інформацію про всі товари і всіх покупців на певну дату. На правому фрагменті малюнка отримано два зрізи, переріз яких буде містити інформацію про всі покупців, але на певний товар і на певну дату.

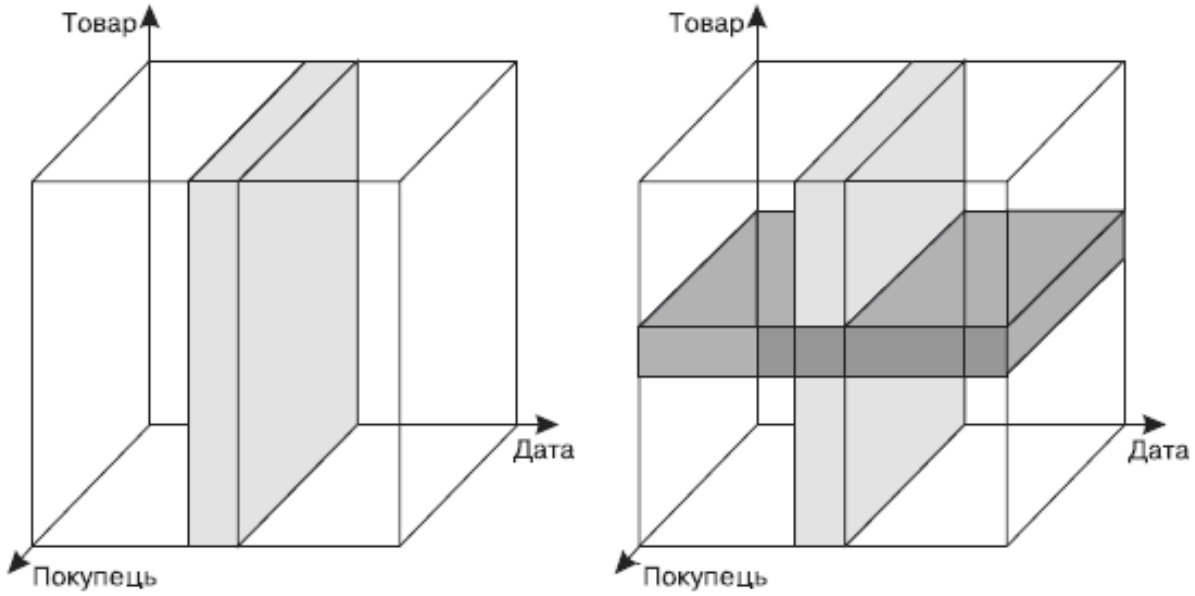


Рис.1.10 – Перерізи гіперкуба

Транспонування (обертання) зазвичай застосовується до плоских таблиць, отриманим, наприклад, в результаті зрізу, і дозволяє змінити порядок подання вимірювань таким чином, що вимірювання, які відображалися в стовпцях, будуть відображатися в рядках, і навпаки. У ряді випадків транспонування дозволяє зробити таблицю більш наочною.

Операції згортки (угруповання) і деталізації (декомпозиції) можливі тільки тоді, коли має місце ієрархічна підпорядкованість значень вимірів. При згортку одне або кілька підлеглих значень вимірів замінюються тими значеннями, яким вони підпорядковані. При цьому рівень узагальнення даних зменшується. Так, якщо окремі товари утворюють групи, наприклад

Будматеріали, то в результаті згортки замість окремих найменувань товарів буде вказано найменування групи, а відповідні їм факти будуть агреговані. Проілюструємо результати згортки: в табл. 2 представлена вихідна таблиця, а в табл. 3 – результат її згортки з вимірювання Товар.

Таблиця 1.3. Вихідна таблиця

Група	Товар	Сума
Будматеріали	Цегла	22 000
	Цемент	12 000
	Керамзит	4500
	Дошка	7400
Інструмент	викрутка	1200
	Електропила	7600
	Дриль	2450
	Шпатель	780

Таблиця 1.4. Результат згортки вихідної таблиці по вимірюванню «Товар»

Група	Сума
Будматеріали	45 900
Інструмент	12 030

Деталізація – це процедура, зворотна згортку; рівень узагальнення даних зменшується. При цьому значення вимірювань вищого ієрархічного рівня замінюються одним або декількома значеннями нижчого рівня, тобто замість найменувань груп товарів відображаються найменування окремих товарів.

1.7 Реляційні сховища даних

На початку 1970-х рр. англо-американський вчений Е. Кодд розробив реляційну модель організації даних, що зберігаються, яка поклала початок новому етапу еволюції СУБД. Завдяки простоті і гнучкості реляційна модель стала домінуючою, а реляційні СУБД стали промисловим стандартом де-факто.

Застосування реляційної моделі при створенні СД в ряді випадків дозволяє отримати переваги над багатовимірної технологією, особливо в частині ефективності роботи з великими масивами даних і використання пам'яті комп'ютера. На основі реляційних сховищ даних (РСД) будуються ROLAP-системи, і ця ідея теж належить Кодду.

В основі технології РСД лежить принцип, відповідно до якого вимірювання зберігаються в плоских таблицях так само, як і в звичайних реляційних СУБД, а факти (дані, що агрегуються) – в окремих спеціальних таблицях цієї ж бази даних. При цьому таблиця фактів є основою для пов'язаних з нею таблиць вимірів. Вона містить кількісні характеристики об'єктів і подій, сукупність яких передбачається в подальшому аналізувати.

1.7.1 Схеми побудови РСД

На логічному рівні розрізняють дві схеми побудови РСД – «зірка» і «сніжинка». При використанні схеми «зірка» центральною є таблиця фактів, з якою пов'язані всі таблиці вимірювань. Таким чином, інформація про кожен вимір розташовується в окремій таблиці, що спрощує їх перегляд, а саму схему робить логічно прозорою і зрозумілою користувачеві (рис.1.11).

Однак розміщення всієї інформації про вимірювання в одній таблиці виявляється не завжди виправданим. Наприклад, якщо товари, що продаються, об'єднані в групи (має місце ієрархія), то доведеться в той чи інший спосіб показати, до якої групи належить кожен товар, що призведе до багаторазового повторення назв груп. Це не тільки викличе зростання надмірності, але і підвищить ймовірність виникнення протиріч (якщо, наприклад, один і той же товар помилково віднесуть до різних груп).

Для більш ефективної роботи з ієрархічними вимірами була розроблена модифікація схеми «зірка», яка отримала назву «сніжинка». Головною особливістю схеми «сніжинка» є те, що інформація про один вимір може зберігатися в декількох зв'язаних таблицях. Тобто якщо хоча б одна з таблиць вимірів має одну або кілька пов'язаних з нею інших таблиць вимірів, в цьому випадку буде застосовуватися схема «сніжинка» (рис. 1.12).

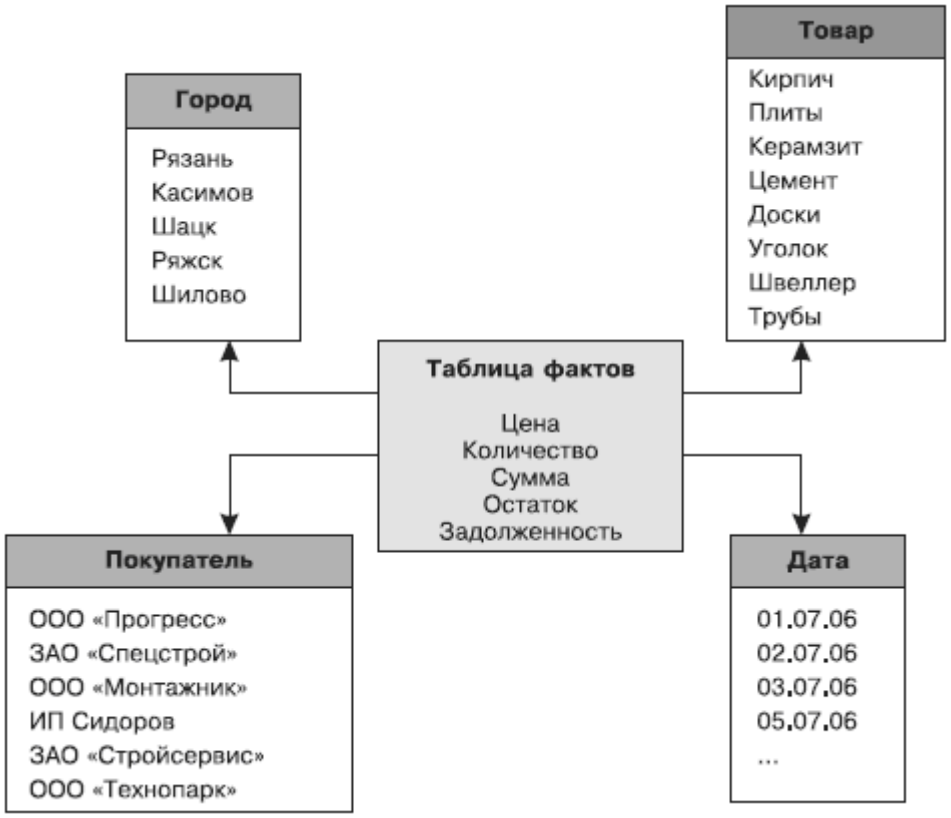


Рис. 1.11 – Схема побудови РСД «зірка»

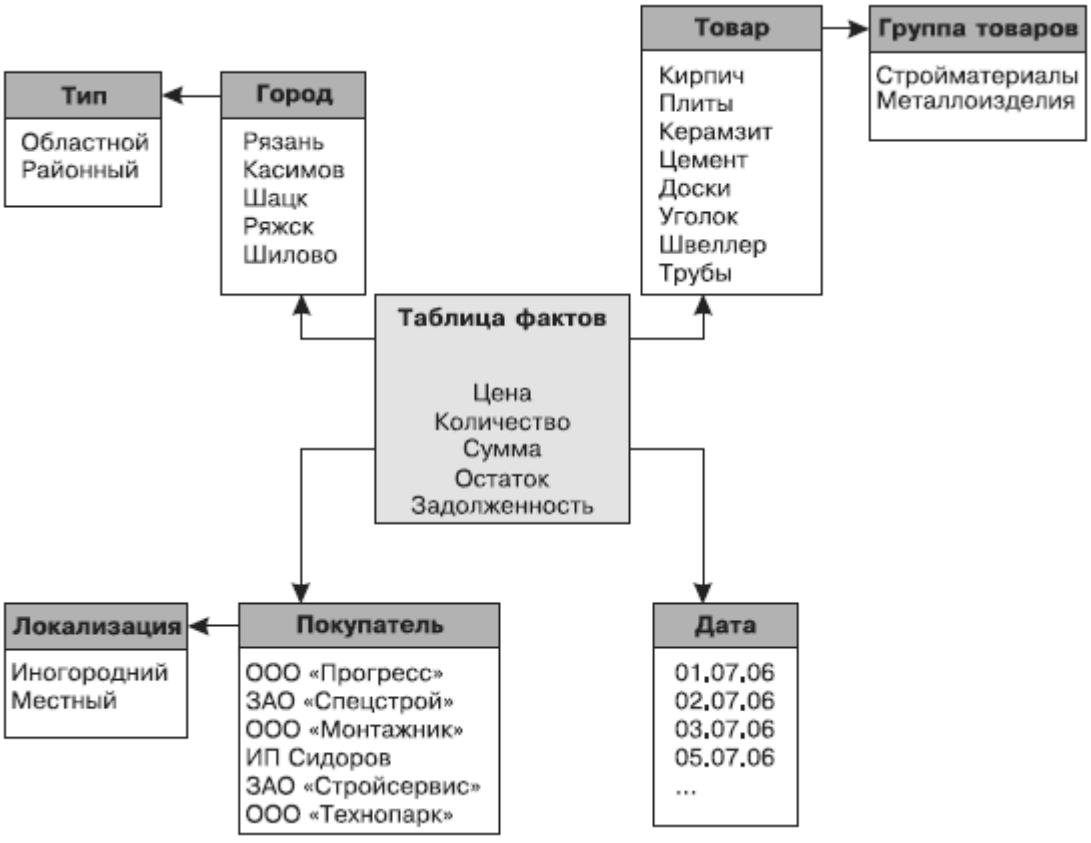


Рис. 1.12 – Схема побудови РСД «сніжинка»

Основна функціональна відмінність схеми «сніжинка» від схеми «зірка» – це можливість роботи з ієрархічними рівнями, що визначають ступінь деталізації даних. У наведеному прикладі схема «сніжинка» дозволяє працювати з даними на рівні максимальної деталізації, наприклад з кожним товаром окремо, або використовувати узагальнене уявлення по групах товарів з відповідною агрегацією фактів.

Вибір схеми для побудови РСД залежить від використовуваних механізмів збору та обробки даних. Кожна зі схем має свої переваги і недоліки, які, однак, можуть проявлятися в більшій чи меншій мірі в залежності від особливостей функціонування СД в цілому. До переваг схеми «зірка» можна віднести:

- простоту і логічну прозорість моделі;
- більш просту процедуру поповнення вимірювань, оскільки доводиться працювати тільки з однією таблицею.

Недоліками схеми «зірка» є:

- повільна обробка вимірювань, оскільки одні й ті ж значення вимірювань можуть зустрічатися декілька разів в одній і тій же таблиці;
- висока ймовірність виникнення невідповідностей в даних (зокрема, суперечностей), наприклад, через помилки введення.

Перевагами схеми «сніжинка» є наступні:

- вона ближче до представлення даних в багатовимірної моделі;
- процедура завантаження з РСД в багатовимірні структури більш ефективна і проста, оскільки завантаження проводиться з окремих таблиць;
- набагато нижча ймовірність появи помилок, невідповідності даних;
- велика, в порівнянні зі схемою «зірка», компактність представлення даних, оскільки всі значення вимірювань згадуються тільки один раз.

Недоліки схеми «сніжинка»:

- досить складна для реалізації і розуміння структура даних;
- ускладнена процедура додавання значень вимірів.

Крім того, існує ряд технічних особливостей, які можуть визначити переваги розробників РСД при виборі схеми його побудови.

1.7.2 Переваги та недоліки РСД

Основні переваги РСД наступні:

- практично необмежений обсяг збережених даних;

- оскільки реляційні СУБД лежать в основі побудови багатьох систем оперативної обробки (OLTP), які зазвичай є головними джерелами даних для СД, використання реляційної моделі дозволяє спростити процедуру завантаження і інтеграції даних в сховищі;
- при додаванні нових вимірів даних немає необхідності виконувати складну фізичну реорганізацію сховища, на відміну, наприклад, від багатовимірних СД;
- забезпечуються високий рівень захисту даних і широкі можливості розмежування прав доступу.

Головний недолік реляційних сховищ даних полягає в тому, що при використанні високого рівня узагальнення даних та ієрархічності вимірювань в таких сховищах починають «розмножуватися» таблиці агрегатів. В результаті швидкість виконання запитів реляційних сховищем сповільнюється.

У той же час в багатовимірних сховищах, де дані зберігаються у вигляді багатовимірних кубів, ця проблема практично не виникає і в більшості випадків вдається досягти більш високої швидкості виконання запитів. Таким чином, вибір реляційної моделі при побудові СД доцільний в наступних випадках.

- Значний обсяг збережених даних (багатовимірні СД стають неефективними).
- Ієрархія вимірювань нескладна (іншими словами, небагато агрегованих даних).
- Потрібно часта зміна розмірності даних. При використанні реляційної моделі можна обмежитися додаванням нових таблиць, а для багатовимірної моделі доведеться виконувати складну перебудову фізичної структури сховища.

1.8 Гібридні сховища даних

Багатовимірна і реляційна моделі сховищ даних мають свої переваги і недоліки. Наприклад, багатовимірна модель дозволяє швидше отримати відповідь на запит, але не дає можливості ефективно управляти такими ж великими обсягами даних, як реляційна модель.

Логічно було б використовувати таку модель СД, яка представляла б собою комбінацію реляційної і багатовимірної моделей і дозволяла б поєднувати високу продуктивність, характерну для багатовимірної моделі, і можливість зберігати скільки завгодно великі масиви даних, властиву реляційної моделі. Така модель, що поєднує в собі принципи реляційної і багатовимірної моделей, отримала назву гібридної, або HOLAP (Hybrid OLAP).

Сховища даних, побудовані на основі HOLAP, називаються гібридними сховищами даних(ГСД) (рис . 1.13). Головним принципом побудови ГСД є те, що деталізовані дані зберігаються в реляційній структурі (ROLAP), яка дозволяє зберігати великі обсяги даних, а агреговані – в багатовимірної (MOLAP), яка дозволяє збільшити швидкість виконання запитів (оскільки при виконанні аналітичних запитів вже не потрібно обчислювати агрегати).

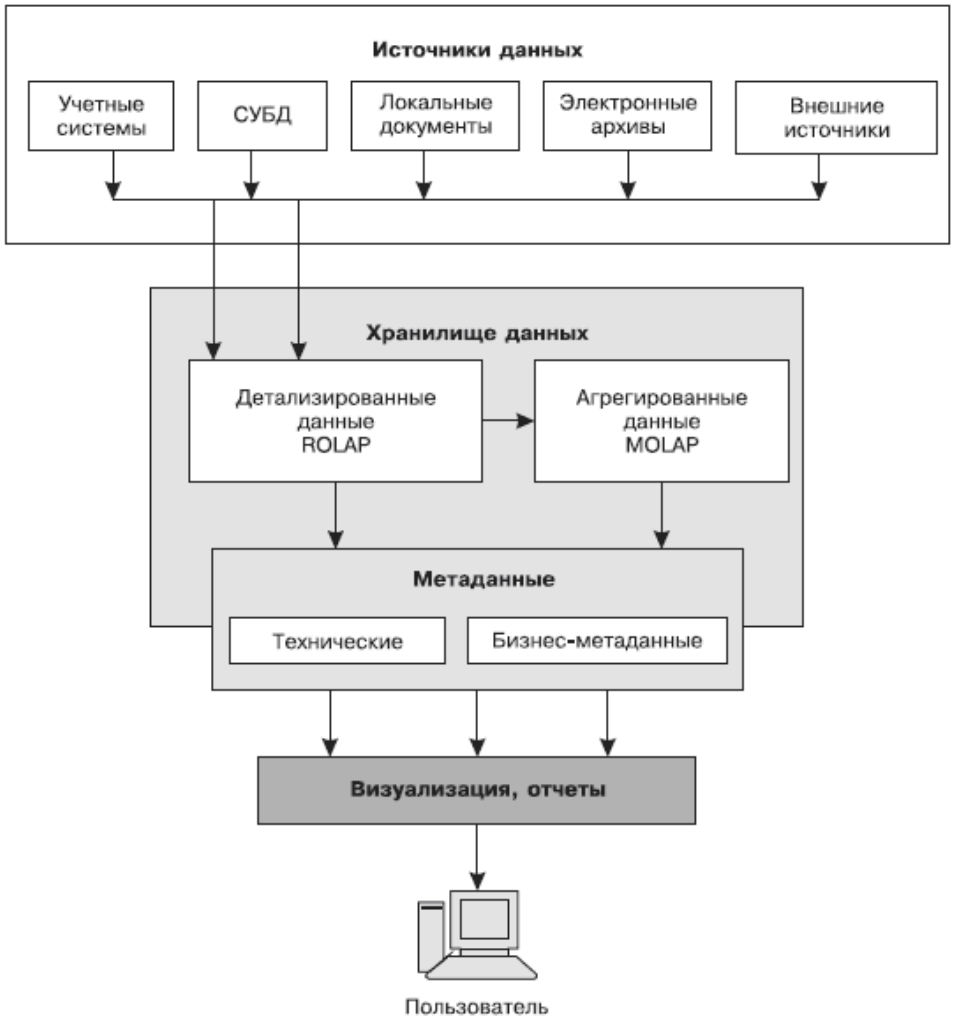


Рис.1.14 – Гібридне СД

Приклад. У супермаркеті, де щодня обслуговуються десятки тисяч покупців, встановлена OLTP-система, що реєструє. При цьому максимального рівня деталізації реєстрованих даних відповідає покупка по одному чеку, в якому зазначаються загальна сума покупки, найменування або коди придбаних товарів і вартість кожного товару. Оперативна інформація, яка складається з деталізованих даних, консолідується в реляційної структурі СД. З точки зору аналізу представляють інтерес узагальнені дані, наприклад, за групами товарів, відділам або деяким інтервалах дат. Тому вихідні деталізовані дані

агрегуються, і обчислені агрегати зберігаються в багатовимірної структурі гібридного СД (рис.1.14).

Рис.1.14 – Реалізація ГСД для супермаркету

Якщо дані, що надходять з OLTP-системи, мають великий обсяг (кілька десятків тисяч записів в день і більше) і високий ступінь деталізації, а для аналізу використовуються в основному великі цифри, гібридна архітектура сховища виявляється найбільш підходящою.

Недоліком гібридної моделі є ускладнення адміністрування СД через більш складний регламент його поповнення, оскільки при цьому необхідно узгоджувати зміни в реляційної і багатовимірної структурах. Однак існує і ряд переваг, які роблять гібридну модель СД привабливою:

- Зберігання даних в реляційної структурі робить їх більшою мірою системно незалежними, що особливо важливо при використанні в управлінні підприємством економічної інформації (показників) .
- Реляційна структура формує стійкі і несуперечливі опорні точки для багатовимірного сховища.
- Оскільки реляційне сховище підтримує актуальність і коректність даних, воно забезпечує дуже надійний транспортний рівень для доставки інформації в багатовимірне сховище.

1.9. Види ієрархії вимірів

Ієрархії у вимірах необхідні для можливості агрегації і деталізації значень показників та подаються у ієрархічній структурі. Існують наступні типи ієрархій [24].

1. Збалансовані (balanced) – ієрархії, в яких кількість рівнів визначена її структурою і незмінна, і кожна гілка ієрархічного дерева містить об'єкти кожного з рівнів. Кожному виробнику автомобілів може відповідати декілька

марок автомобілів, а кожній марці – декілька моделей автомобілів, тому можна говорити про триївневу ієрархію цих об'єктів,. В цьому випадку на першому рівні ієрархії розташовуються виробники, на другому – марки, а на третьому – моделі. Як неважко зрозуміти, для формування збалансованої ієрархії необхідна наявність зв’язку «один-до-багатьох» між об’єктами менш детального рівня по відношенню до об’єктів детальнішого рівня. У принципі, кожен рівень збалансованої ієрархії можна подати як окремі прості виміри, але тоді ці виміри виявляться залежними, а значить, неминуче підвищення розрідженості куба.

2. Незбалансовані (unbalanced) – ієрархії, в яких кількість рівнів може бути змінена, і кожна гілка ієрархічного дерева може містити об’єкти, що належать не всім рівням, тільки декільком першим. Необхідно зазначити, що всі об’єкти незбалансованої ієрархії належать до одного типу. Типовий приклад незбалансованої ієрархії – ієрархія типу «керівник – підлеглий», де всі об’єкти мають один і той самий тип – «Співробітник».

3. Нерівні (ragged) – ієрархії, в яких кількість рівнів визначена її структурою і незмінна, проте, на відміну від збалансованої ієрархії, деякі гілки ієрархічного дерева можуть не містити об’єкти якогось рівня. Ієрархії такого вигляду містять такі члени, логічні «батьки» яких не перебувають на безпосередньо вищому рівні. Типовим прикладом є географічна ієрархія, в якій є рівні «Країни», «Штати» і «Міста», але при цьому в наборі даних є країни, що не мають штатів або регіонів між рівнями «Країни» і «Міста».

Також існують ієрархії вимірів з декількома різними ієрархіями для одного і того ж виміру. Типові приклади таких ієрархій – ієрархії для календарного і фінансового року (за умови, що фінансовий рік починається не з 1 січня), або з різними способами групування членів виміру (наприклад, групувати товари можна за категоріями, а можна і за компаніями-постачальниками). У цьому випадку таблиця вимірів містить поля для всіх можливих ієрархій з одними і тими ж членами нижнього рівня, але з різними членами верхніх рівнів (приклад такої таблиці наведений на рис. 1.15).

Як вже зазначалося вище, таблиця вимірів може містити поля, що не мають відношення до ієрархій і є додатковими атрибутами членів вимірів. Іноді такі атрибути використовуються при аналізі даних.

Типовий приклад незбалансованої ієрархії – ієрархія типу «керівник – підлеглий» (її можна побудувати, зокрема, використовуючи значення поля ***Прізвище_підлеглого*** початкового набору даних з розглянутого вище прикладу), поданий на рис. 1.16.

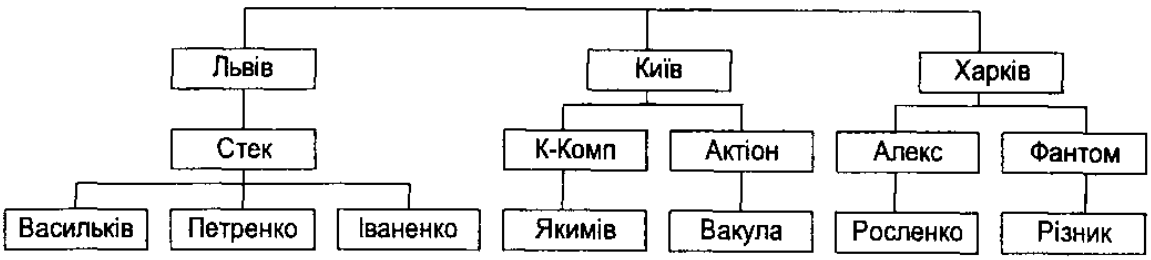


Рис. 1.15 – Ієрархія у вимірі, пов'язаному з географічним положенням клієнтів

Рис. 1.16 – Незбалансована ієрархія «керівник – підлеглий»

Іноді для таких ієрархій використовується термін Parent-child hierarchy.

Існують також ієрархії, що займають проміжне положення між збалансованими і незбалансованими (вони позначаються терміном ragged – «нерівний»). Зазвичай вони містять такі члени, логічні «батьки» яких перебувають не на безпосередньо вищому рівні (наприклад, у географічному розміщенні підрозділів підприємства, але є випадки, що в одному корпусі розміщено лише один підрозділ; (рис. 1.17)).

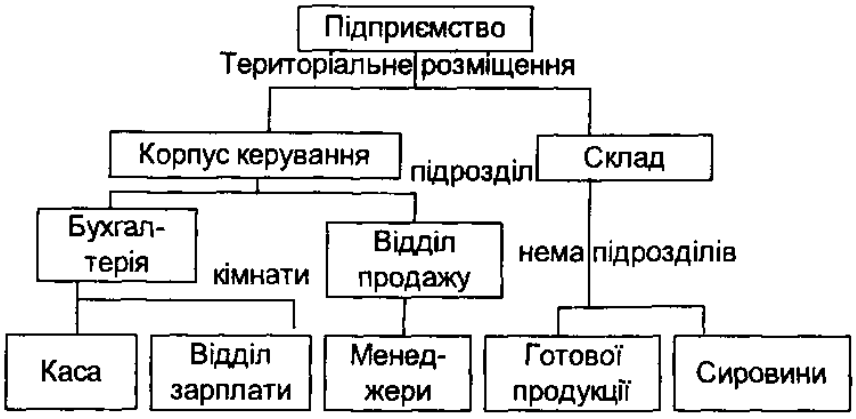


Рис. 1.17 – Нерівна ієрархія

Нерегулярні ієрархії виникають в різних застосуваннях, зокрема в ієрархії адміністративних структур [4], ієрархії медичних діагнозів [5] і ієрархії концепцій для Web-порталів, подібних до Yahoo.

1.10 Підвиди сховищ даних

1.10.1. Вітрини даних

Вітрина даних (ВД) – зріз сховища даних, масив тематичної, вузько напрямленої інформації, що орієнтований, наприклад, на користувачів однієї робочої групи або департаменту.

Концепція вітрин даних була запропонована Форестером Рісечером (ForesterResearch) ще в 1991 році. За визначенням авторів ***вітрини даних*** – множина тематичних БД тих, що містять інформацію, яка відноситься до окремих аспектів діяльності організації.

Концепція має ряд безперечних переваг.

1. Аналітики бачать і працюють тільки з тими даними, які їм реально потрібні
2. Цільова БД максимально наближена до кінцевого користувача.
3. ВД зазвичай містять тематичні підмножини наперед агрегованих даних, їх простіше проектувати і налаштовувати.
4. Для реалізації ВД не потрібна потужна обчислювальна техніка.

Але концепція ВД має і дуже серйозні недоліки. За суттю, передбачається реалізація територіально розподіленої інформаційної системи з низько контрольованою надмірністю, але, не пропонується способів, як забезпечити цілісність і несуперечність даних, що зберігаються в ній.

Ідея з'єднати дві концепції – сховищ даних і вітрин даних належить М. Демаресту (M. Demarest), який в 1994 році запропонував використати СД як єдине інтегроване джерело даних для ВД.

Наведемо структуру даних в СД з врахування підвидів СД (рис.1.18).

На сьогодні існує таке багаторівневе рішення:

- перший рівень – загальнокорпоративна БД на основі реляційної СУБД з нормалізованою або слабо денормалізованою схемою (деталізовані дані);
- другий рівень – БД рівня підрозділу (або кінцевого користувача), реалізовані на основі багатовимірної СУБД (агреговані дані);
- третій рівень – робочі місця кінцевих користувачів, на яких безпосередньо встановлений аналітичний інструментарій.

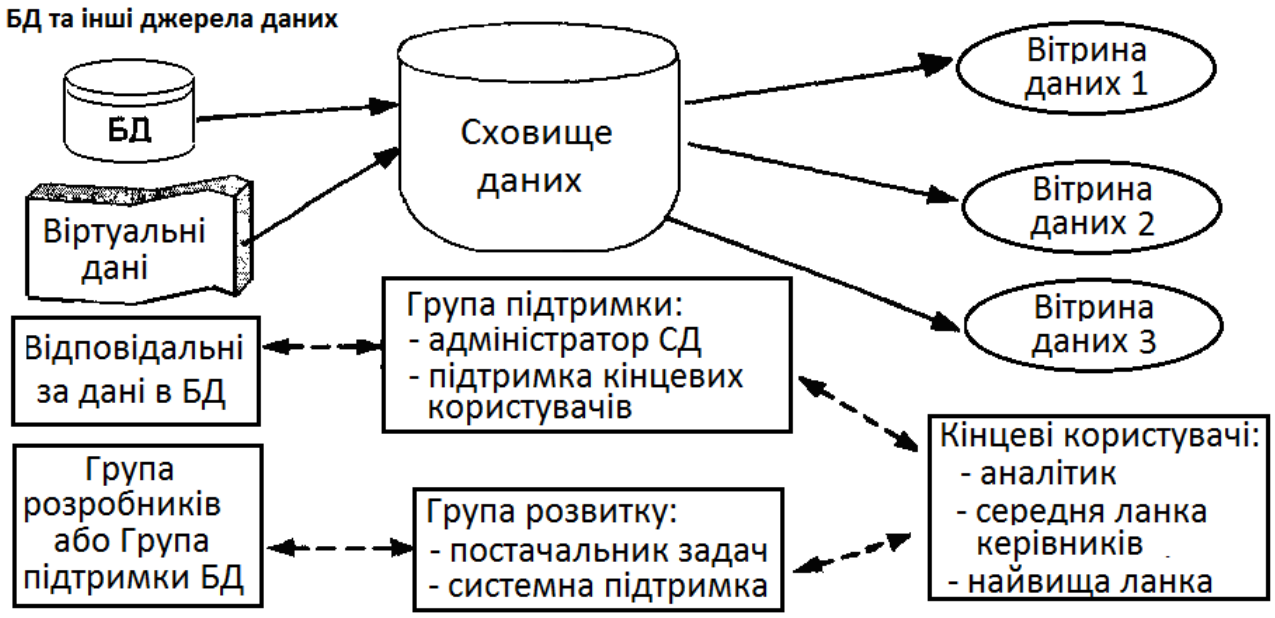


Рис. 1.18 – Структура даних сховища даних з використанням ВД

Це рішення поступово стає стандартом де-факто, дозволяючи якнайповніше реалізувати і використати переваги кожного з підходів:

- компактне зберігання деталізованих даних і підтримка дуже великих БД що забезпечуються реляційними СУБД;
- простота налаштування і хороші часи відгуку при роботі з агрегованими даними, що забезпечуються багатовимірними СУБД.

Реляційна форма подання даних, яка використовується в центральній загальнокорпоративній БД, забезпечує найкомпактніший спосіб зберігання даних. А сучасні реляційні СУБД вже уміють працювати навіть з терабайтними базами. І хоча така центральна система зазвичай не зможе забезпечити оперативний режим опрацювання аналітичних запитів, при використанні нових способів індексації і зберігання даних, а також часткової денормалізації таблиць, час опрацювання наперед регламентованих запитів (а як такі, можна розглядати і регламентовані процедури вивантаження даних в багатовимірні БД) виявляється цілком прийнятним.

У свою чергу, використання багатовимірних СУБД у вузлах нижнього рівня забезпечує мінімальні часи опрацювання і відповіді на нерегламентовані запити користувача. Крім того, в деяких багатовимірних СУБД є можливість зберігати дані як на постійній основі (безпосередньо у багатовимірній БД), так і динамічно (на час сеансу) завантажити дані з реляційних БД (на основі регламентованих запитів).

Отже, є можливість зберігати на постійній основі тільки ті дані, які

найчастіше запитуються в цьому вузлі. Для всіх інших зберігаються тільки описи їх структури і програми їх вивантаження з центральної БД. І хоча при первинному зверненні до таких віртуальних даних час відгуку може виявитися достатньо тривалим, таке рішення забезпечує значну гнучкість і вимагає дешевших апаратних засобів.

Дворівнева архітектура сховища даних передбачає побудову вітрин даних без створення центрального сховища, при цьому інформація надходить із реєстраційних систем і обмежена конкретною предметною областю. При побудові вітрин використовуються основні принципи побудови сховищ даних, тому їх можна вважати сховищами даних у мініатюрі (рис. 1.19).

Рис. 1.19 – Дворівнева структура сховища даних

Перевагами такої структури є:

- простота й мала вартість реалізації;
- висока продуктивність за рахунок фізичного поділу реєстраційних і аналітичних систем;
- виділення завантаження і трансформації даних в окремий процес, оптимізованої під аналіз структурою зберігання даних;
- підтримка історії;
- можливість додавання метаданих.

Побудова повноцінного корпоративного сховища даних зазвичай виконується в ***трирівневій архітектурі*** (рис. 1.20). На першому рівні розташовані різноманітні джерела даних – внутрішні реєстраційні системи, довідкові системи, зовнішні джерела (дані інформаційних агентств, макроекономічні показники). Другий рівень містить центральне сховище, куди стікається інформація від всіх джерел з першого рівня, і, можливо, оперативне сховище даних (сховище поточної інформації, розглянуто далі), що не містить історичних даних і виконує дві основні функції:

1) по-перше, воно є джерелом аналітичної інформації для оперативного керування,
2) по-друге, тут підготовляються дані для наступного завантаження в

центральне сховище.

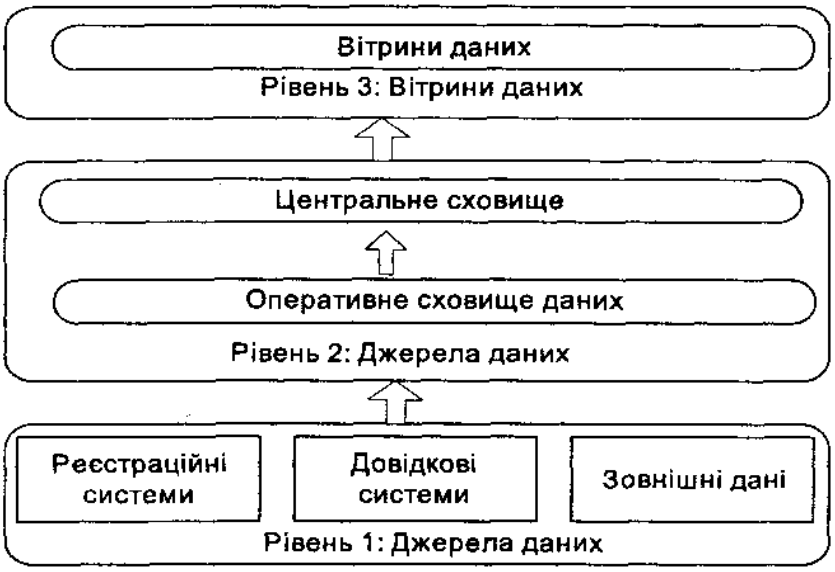


Рис. 1.20 – Корпоративне сховище даних

Під *підготовкою даних* розуміють їхнє перетворення і проведення певних перевірок. Наявність оперативного сховища даних просто необхідно при різному регламенті надходження інформації із джерел. Третій рівень являє собою набір предметно-орієнтованих вітрин даних, джерелом інформації для яких є центральне сховище даних. Саме з вітринами даних і працює більшість кінцевих користувачів.

Існує такий тип вітрин даних, як *Conformed data marts* – зв'язаний набір схем типу зірка, в основі якої лежить множина таблиць фактів і таблиць вимірів, зв'язаних відношенням первинних/зовнішніх ключів. Недоліки цього типу вітрин даних – обмежена предметно-орієнтована інформація, розрідженість даних, несумісність гранул даних.

1.10.2. Операційні сховища даних

Рішення завдань, пов'язаних з оперативним аналізом, забезпечують операційні сховища даних (Operational Data Store – ODS).

Операційне сховище даних (ОСД) – це предметно-орієнтований, інтегрований, змінюваний набір консолідованих даних, який містить поточну (не історичну) деталізовану інформацію.

На перший погляд, операційне сховище даних дуже схоже на сховище за структурою і змістом. Зазвичай за деякими характеристиками ОСД і сховище

даних дуже схожі, але ОСД має ряд властивостей, які істотно відрізняють його від сховища. Як ОСД, так і сховище даних є предметно-орієнтованим інтегрованим набором консолідованих даних. З цієї точки зору вони схожі, оскільки як в одному, так і в іншому випадку дані повинні бути завантажені з транзакційних систем. Але на цьому їх схожість закінчується. ОСД містить дані, що змінюються, тоді як в сховищі дані після завантаження не змінюються. Інша відмінність полягає у тому, що операційне сховище містить тільки дані, актуальні на певний момент часу, тоді як в сховищі містяться як поточні, так і історичні дані. При цьому актуальність даних в сховищі значно нижча, ніж в операційному сховищі. Як правило, в сховищі містяться дані, завантажені протягом останніх 24 годин, тоді як актуальність даних в ОСД може вимірюватися секундами. Ще однією відмінністю ОСД від сховища є те, що в ньому містяться тільки детальні дані, тоді як сховище містить як детальні, так і агреговані дані.

Окрім завдань оперативного аналізу даних, ОСД можуть використовуватися при інтеграції транзакційних систем. У разі використання ОСД інформаційні потоки між системами будуються таким чином, що обмін даними відбувається між транзакційною системою і ОСД, а не безпосередньо між транзакційними системами. Загальна кількість інтерфейсів між системами скорочується (рис.1.21).

Рис. 1.21 – Схема сховища даних з використанням ОСД

Як приклад подібного ОСД можна навести довідник контрагентів. Як правило, інформація про контрагентів міститься в більшості транзакційних систем. При цьому, в кожній інформаційній системі враховується якийсь свій блок інформації і виділити одну систему як головну (у якій додаватимуться контрагенти, а потім розповсюджуватися в інші системи) не завжди можливо.

Для розв’язування такого завдання створюється ОСД, в яку з транзакційних систем потрапляє інформація про контрагентів (з кожної системи свій блок інформації). Отже, одержуємо єдиний довідник контрагентів, в якому міститься максимально повна інформація. При цьому в ОСД повинен бути реалізований механізм пошуку «двійників», за яким система визначатиме, що такий контрагент вже існує. Найпростішим прикладом такого пошуку може бути порівняння якогось унікального коду, наприклад ідентифікаційного номера.

Отже, ОСД використовується для прямого копіювання даних з початкових систем в те ж середовище, де розташовується справжнє сховище. При цьому підході в ОСД містяться точні копії даних початкових систем для полегшення подальшого завантаження в сховище даних з використанням однотипних засобів вивантаження/завантаження. Зазвичай ОСД використовується при ускладненому доступі до початкових даних за часом або розташуванням.

Резюме

1. Сховище даних – це агрегований інформаційний ресурс, що містить консолідовану інформацію з усієї проблемної області та використовується для підтримки прийняття рішень.
2. Консолідована інформація – це одержані з декількох джерел та системно інтегровані різнотипні інформаційні ресурси, які в сукупності наділені ознаками повноти, цілісності, несуперечності та складають адекватну інформаційну модель проблемної області, з метою її аналізу, опрацювання та ефективного використання в процесах підтримки прийняття рішень.
3. Дві основні ідеї сховища даних: інтеграція роз’єднаних детальних даних; розділення наборів даних і застосувань, що використовуються для оперативного опрацювання і для розв'язування задач аналізу.
4. Особливості проектування та побудови сховищ даних: отримання даних з різних джерел, багатовимірне подання, наявність метаданих, пакетне завантаження даних, наявність процедур аналізу даних та отримання нових даних, орієнтованість даних на аналітичне опрацювання.
5. Є такі типи даних у сховищі даних: метадані, детальні дані, агреговані дані.
6. Метадані – це структуровані, кодовані дані, які описують

характеристики об'єктів-носіїв інформації, що сприяють ідентифікації, виявленню, оцінюванню, керуванню цими об'єктами. Для інформації про дані сховища доцільно застосовувати шестивимірну класифікаційну схему Захмана (відповідно до відповідей на запитання що? хто? де? коли? чому? як?).

7. Агрегація даних -- це обчислення узагальнених показників для підтримки стратегічного або тактичного керування з детальних даних.
8. Учасники сховища даних: кінцеві користувачі, група розвитку, група підтримки.
9. Типи запитів до сховища даних: зріз, згортка, деталізація, комбінація кубів за одним або декількома значеннями вимірів, впорядкування вимірів, поворот куба.
10. Типи ієрархії вимірів: збалансовані, незбалансовані, нерівні.
11. Вітрина даних (ВД) – зріз сховища даних, масив тематичної, вузьконапрямленої інформації, що орієнтований, наприклад, на користувачів однієї робочої групи або департаменту. Це множина тематичних БД тих, що містять інформацію, яка відноситься до окремих аспектів діяльності організації.
12. Дворівнева архітектура сховища даних передбачає побудову вітрин даних без створення центрального сховища, при цьому інформація надходить із реєстраційних систем і обмежена конкретною предметною областю. При побудові вітрин використовуються основні принципи побудови сховищ даних, тому їх можна вважати сховищами даних у мініатюрі.
13. У трирівневій архітектурі сховища даних на першому рівні розташовані різноманітні джерела даних – внутрішні системи, що реєструють, довідкові системи, зовнішні джерела (дані інформаційних агентств, макроекономічні показники). Другий рівень містить центральне сховище, куди стікається інформація від всіх джерел з першого рівня, і, можливо, третій – оперативне сховище даних.
14. Операційне сховище даних (ОСД) – це предметно-орієнтований, інтегрований, змінюваний набір консолідованих даних, який містить поточну (не історичну) деталізовану інформацію. ОСД містить дані, що змінюються, тоді як в сховищі дані після завантаження не змінюються. Інша відмінність полягає у тому, що операційне сховище містить тільки дані, актуальні на певний момент часу, тоді як в сховищі містяться як поточні, так і історичні дані.

2 ІНТЕГРАЦІЯ ДАНИХ

2.1. Значення інтеграції для сховищ даних

Сховище даних має значення для розв'язування багатьох аналітичних проблем. Хоча форми існування сховищ бувають різноманітними (зокрема сюди відносяться вітрини даних і оперативні сховища даних, що містять поточну, а не історичну інформацію), кожна з них здатна створити платформу даних, яка може використовуватися в аналітичних цілях. Консолідуючи, стандартизуючи і, у багатьох випадках, об'єднуючи дані, що містяться в декількох операційних системах, організація може аналізувати ці сумарні дані для отримання найбільш об'єктивної картини.

Сховище даних може створюватися з наступною метою:

- інтеграція поточних і історичних значень даних;
- об'єднання даних з розрізнених джерел;
- створення надійної платформи даних для аналітичних цілей;
- забезпечення однорідності даних в організації;
- полегшення впровадження корпоративних стандартів даних без зміни наявних операційних систем;
- забезпечення широкої історичної картини і можливостей для аналізу тенденцій.

Оперативне сховище даних створюється в наступних цілях:

- отримання повної інформації про об'єкти предметної області;
- інтеграція поточних фінансових даних для обов'язкової звітності і виконання вимог законодавства;
- консолідація поточної інформації з декількох джерел; аналіз поточних значень і тенденцій.

Навіть якщо операційні системи зберігають історичні дані, їх кількість і рівень детальності зазвичай достатньо обмежені. Наприклад, продаж минулого року по кожному клієнту може бути згрупований в єдине значення, а детальніша інформація, необхідна для оцінки щомісячного продажу по клієнтах, після цього є недоступною (або відсилається в архів в автономне резервне сховище даних). Однією з переваг сховища даних є те, що навіть при зберіганні сумарних даних і відсутності детальніших записів сховище дозволяє мати декілька сумарних показників (наприклад, щоденний продаж по клієнтах, продуктах, магазинах).

Через дію процесів предметної області величини показників постійно змінюються, оскільки більшість транзакцій зазвичай приводять до трансформації одного або декількох значень даних. Це створює труднощі при аналізі даних, оскільки зміна навіть однієї величини веде до трансформації всього результату. Такої проблеми можна уникнути, якщо вміщати «миттєві знімки» даних в сховищі. Ці значення можуть відрізнятися від поточних величин кожного подальшого моменту, але вони зазвичай збираються в добре визначених, закінчених циклах (наприклад, щомісяця, щотижня, щодня або щогодини), що дозволяє здійснювати надійні порівняння різних періодів.

Дані як активи корпорації.

Хоча нерідко організації розглядають свої дані як корпоративні активи, їх кількість, як і у разі багатьох інших активів, не завжди обмежена. Дані подають такий актив, який може розростатися і тиражуватися практично безмежно, при цьому часто змінюючись у ході процесу. Необхідно мати на увазі, що дані в реальності не переміщаються з однієї системи в іншу, а мають тенденцію спонтанно тиражуватися з кожним новим запитом на витягання. І якщо тут не встановлена відповідна дисципліна, подібна тій, яка існує при інтеграції в сховищах даних, то в кожному новому поколінні даних з’являтимуться мутації.

Інтеграція в сховищі даних з різних операційних систем сприяє отриманню найбільш об’єктивної картини. Це дозволяє працювати з даними як з дуже важливим корпоративним активом, яким вони, за суттю, і є. Щоб така робота була ефективною, відомості про дані, зокрема про їх походження і/або трансформацію, повинні бути легко доступні і у жодному випадку не загублені.

Відмінності у визначеннях даних і правилах бізнесу.

Після того, як організація виявляє відмінності у визначеннях даних і здійснює стандартизацію єдиних корпоративних визначень, сховище даних може сприяти їх впровадженню в практику. Перебудовувати кожну операційну систему під корпоративний стандарт непрактично. Але в процесі завантаження в СД даних, витягнутих з різних операційних систем, можливо здійснити їх трансформацію для того, щоб вони відповідали визначенням корпоративних стандартів і спискам значень.

Підтримка продуктивності і часу реагування операційних систем.

Опрацювання запитів або створення звітів за допомогою бази даних, яка використовується оперативною прикладною програмою, негативно позначається на продуктивності і часі виконання запитів користувачів цією прикладною програмою. Якщо аналітичний запит негативно впливає на час

реагування операційної системи, то його виконання буде відкладене, можливо, назавжди. При використанні сховища даних ця проблема розв'язується, оскільки запит вивантажується в середовище, де база даних може бути оптимізована для його виконання.

Інтегровані дані забезпечують структуру, яка допомагає організації:

- мати повну інформацію про клієнтів;
- знизити навантаження операційних систем;
- стандартизувати процеси бізнесу і визначення даних;
- об'єднувати поточні і минулі значення з розрізнених джерел для отримання повної картини бізнесу.

Надійні дані – це основа прийняття зважених рішень. А інтеграція даних – це ключ до контролю інформації, оскільки користувачі інструментів Business Intelligence (BI) повинні бути впевнені, що їх рішення ґрунтуються на надійних даних. Найкращі інструменти BI виявляються малоефективними, якщо вони використовуються для аналізу неповних і неточних даних.

Операційні й аналітичні системи доповнюють одні одних. Для досягнення успіху організаціям необхідно ефективно використовувати і ті, й інші. Аналітичні цілі, такі як аналіз тенденцій зміни і прогнозування, вимагають збирання даних з відмітками часу з численних джерел в єдине сховище або вітрину даних. Для операційних цілей часто необхідно створювати звіти на основі даних, що містяться в операційних системах. Консолідація поточних даних з численних операційних систем може здійснюватися за допомогою операційних сховищ даних.

При збиранні всіх цих даних використовуються різні підходи і методи інтеграції даних. Рішення для інтеграції даних і підвищення їх якості – це ключ до контролю інформації. Засоби інтеграції даних сприяють створенню сховищ, вітрин і операційних сховищ даних. Все ці структури забезпечують організації можливостями для прийняття надійних рішень. Успішна інтеграція даних – це ключовий чинник для успіху організації в Business Intelligence.

Багатоаспектність проблеми

Проблема інтеграції даних надзвичайно багатоаспектна і різноманітна. Складність і характер використовуваних методів її вирішення істотним чином залежать від рівня інтеграції, який необхідно забезпечити, властивостей окремих джерел даних і усієї множини джерел в цілому, необхідних способів інтеграції.

Системи інтеграції даних можуть забезпечувати інтеграцію даних на фізичному, логічному і семантичному рівні. Інтеграція даних на фізичному

рівні з теоретичної точки зору є найбільш простим завданням і зводиться до конверсії даних з різних джерел в необхідний єдиний формат їх фізичного представлення. У доповіді обговорюються, головним чином, два інші випадки. Інтеграція даних на логічному рівні передбачає можливість доступу до даних, що містяться в різних джерелах, в термінах єдиної глобальної схеми, яка описує їх спільне представлення з урахуванням структурних і, можливо, поведінкових (при використанні об'єктних моделей) властивостей даних. Семантичні властивості даних при цьому не враховуються. Підтримку єдиного представлення даних з урахуванням їх семантичних властивостей в контексті єдиної онтології предметної області забезпечує інтеграція даних на семантичному рівні.

Джерела даних можуть мати різні властивості, істотні для вибору методів інтеграції даних, — вони можуть підтримувати представлення даних в термінах тієї або іншої моделі даних, можуть бути статичними або динамічними і тому подібне. Множина джерел інтегрованих даних може бути однорідною або неоднорідною відносно характеристик, що відповідають використовуваному рівню інтеграції.

Що стосується способів інтеграції даних, то можливі два підходи — віртуальне або актуальне (матеріалізоване) представлення інтегрованих даних. При першому підході створюється механізм доступу, який при обробці запиту користувача породжує дані в необхідному відображенні безпосередньо з джерел даних. Повне матеріалізоване відображення інтегрованих даних в термінах єдиного інтерфейсу користувача при цьому не підтримується. Віртуальний підхід найчастіше застосовується при використанні часто оновлюваних джерел даних. Навпаки, при другому підході на стадії інтеграції формується повне матеріалізоване представлення інтегрованих даних, що відчужене від початкових джерел і співіснує з ними. Саме це представлення даних використовується для обробки запитів користувача. Такий підхід використовується, зокрема, в сховищах даних.

Неоднорідність джерел даних

Неоднорідність джерел даних проявляється в системах інтеграції даних в різних аспектах. При цьому, природно, йде мова про неоднорідність характеристик джерел, що відповідають використовуваному рівню інтеграції даних.

Так, при інтеграції на фізичному рівні в джерелах даних можуть використовуватися різні формати файлів. На логічному рівні інтеграції може мати місце неоднорідність використовуваних моделей даних для різних джерел

або розрізняються схеми даних, хоча використовується одна і та ж модель даних. Одні джерела можуть бути веб-сайтами, а інші — об'єктними базами даних і так далі

При інтеграції на семантичному рівні різним джерелам даних можуть відповідати різні онтології. Наприклад, можливий випадок, коли кожне з джерел представляє інформаційні ресурси, що моделюють деякий фрагмент предметної області, якому відповідає своя понятійна система, і ці фрагменти перетинаються.

Завдання, що виникають

При створенні системи інтеграції виникає ряд завдань, склад яких залежить від вимог до неї і використовуваного підходу. До них, зокрема, відносяться:

- Розробка архітектури системи інтеграції даних.
- Створення інтегруючої моделі даних, що є основою єдиного призначеного для користувача інтерфейсу в системі інтеграції.
- Розробка методів відображення моделей даних і побудова відображень в інтегруючу модель для конкретних моделей, що підтримуються окремими джерелами даних.
- Інтеграція метаданих, які використовуються в системі джерел даних.
- Подолання неоднорідності джерел даних.
- Розробка механізмів семантичної інтеграції джерел даних.

2.2. Поняття інтеграції даних

Інтеграція даних в інформаційних системах розуміється як забезпечення єдиного уніфікованого інтерфейсу для доступу до деякої сукупності, взагалі кажучи, неоднорідних незалежних джерел даних [16, 17]. Таким чином, для користувача інформаційні ресурси усієї сукупності джерел даних, що інтегруються, видаються як нове єдине джерело. Система, що забезпечує користувачу такі можливості, називається ***системою інтеграції даних.***

Система інтеграції даних звільняє користувачів від необхідності пізнання, дані з яких джерел, окрім інтегрованого, вони використовують, які властивості цих джерел і як здійснити доступ до них. Інтегрованими джерелами даних можуть бути традиційні системи баз даних, що підтримують різні моделі даних (реляційні, об'єктні, об'єктно-реляційні, графові і тому подібне), різноманітні успадковані системи, репозиторії, веб-сайти, файли структурованих даних. Забезпечення доступу до даних багатьох джерел через

єдиний інтерфейс означає фактично, що йдеться про підтримку представлення сукупності даних з множини незалежних джерел в термінах єдиної моделі даних. Важливо, нарешті, відмітити, що склад множини джерел може бути наперед заданим або динамічно поповнюваним, джерела даних можуть мати незмінний або оновлюваний зміст.

2.2.1 Історія засобів інтеграції

Перші засоби інтеграції з'явились на початку 90-х рр. минулого століття. Вони дозволяли описувати процедури перетворення даних на мовах програмування, наприклад, на COBOL.

Середина 90-х рр. XX ст. знаменна появою нових засобів інтеграції даних, заснованих на пропрієтарному ядрі і мові програмування. Протягом наступних десяти років розробники СКБД істотно збільшували можливості SQL і утиліт СКБД.

У 1998 р. Sunopsis випускає унікальний засіб інтеграції даних, що генерує оптимізований SQL-код і що дозволяє задіяти всі можливості і переваги СКБД, які використовуються в проекті.

Розробка методів інтеграції інформаційних ресурсів — одна з найбільш актуальних проблем в області інформаційних систем. Особливо велику увагу вона стала привертати останніми роками. Проте проблема інтеграції даних зовсім не є новою. Перші кроки в цій області відносяться ще до середини 70-х рр., коли почалися розробки розподілених систем баз даних і коли багато в чому завдяки звіту ANSI/X3/SPARC [11] сформувалися чіткіші уявлення про багаторівневу архітектуру систем баз даних, про моделі даних як інструмент моделювання реальності і про відображення моделей даних.

При цьому йшлося, головним чином, про підтримку глобальної схеми для сукупності локальних баз даних, що функціонують в різних вузлах мережі під управлінням СУБД, які підтримують одну і ту ж або, в загальному випадку, різні моделі даних. Пізніше дещо загальніша форма цього завдання була пов'язана із створенням мультибаз і федеральних баз даних, сховищ даних, різних репозиторіїв інформаційних ресурсів, а також веб-додатків. Останніми роками в розробках електронних бібліотек (Digital Libraries), що широко розгорнулися у багатьох країнах проблеми інтеграції неоднорідних даних стали грати ключову роль, причому виникає також завдання інтеграції текстових інформаційних ресурсів з різних незалежних джерел.

2.2.2 Проблеми, що призводять до інтеграції даних

Зупинимося на деяких проблемах реалізації сховища даних, які приводять до виникнення задачі інтеграції даних. Серед них можна виділити:

- неоднорідність програмного середовища;
- розподілений характер організації;
- підвищені вимоги до безпеки даних;
- необхідність наявності багаторівневих довідників метаданих;
- потреба в ефективному зберіганні й опрацюванні дуже великих обсягів інформації.

Неоднорідність програмного середовища.

Сховище даних практично ніколи не створюється на порожньому місці. Майже завжди кінцеве рішення буде різнорідним, тобто в ньому використовуватимуться автономно розроблені програмні засоби. Перш за все це торкається формування інтегрованого узгодженого набору даних, які можуть надходити з різнорідних баз даних, електронних архівів, публічних і комерційних електронних каталогів, довідників, статистичних збірок. При побудові сховища даних доводиться вирішувати завдання побудови єдиної інформаційної системи, що погоджено функціонує *на основі неоднорідних програмних засобів і рішень*. При виборі засобів реалізації сховища даних доводиться враховувати множину чинників, що включають рівень сумісності різних програмних компонентів, легкість їх освоєння і використання, ефективність функціонування та ін.

Розподілений характер організації.

Концепцією сховища даних визначено, що операційне аналітичне опрацювання може виконуватися в будь-якому вузлі мережі незалежно від місця розташування основного сховища. Хоча при аналітичному опрацюванні дані тільки читаються, і потреба в синхронізації відсутня, для досягнення ефективності необхідно підтримувати реплікацію даних в різних вузлах мережі. (Насправді, все не так просто. Однією з вимог до сховищ даних є те, щоб свіжа інформація надходила в сховище якнайшвидше. Тобто потенційно будь-яка модифікація оперативної БД може ініціювати додавання даних до сховища даних, а тоді потрібно буде відновити і всі репліки, для чого синхронізація все-таки потрібна.)

Підвищення вимог до безпеки даних.

Зібрана узгоджена інформація про предметну область дає можливість аналізу минулої і поточної діяльності об’єктів за поведінкою процесів і

побудови прогнозів для майбутнього. Ця інформація настільки цінна для корпорації, що не можна допустити можливості її витоку (насправді, якщо сховище даних однієї корпорації потрапить до рук аналітиків іншої корпорації, то всі аналітичні прогнози першої корпорації відразу стануть невірними). У системах, заснованих на сховищах даних, виявляється недостатнім захист даних в стилі мови SQL, який забезпечують звичні комерційні СКБД (цей рівень захисту відповідає класу С2 відповідно до класифікації *Оранжевої Книги Міністерства оборони США*). Для забезпечення належного рівня захисту доступ до даних повинен контролюватися не тільки на рівні таблиць і їх стовпців, але і на рівні окремих рядків (це вже відповідає класу В1 Оранжевої Книги). Доводиться також вирішувати питання аутентифікації користувачів, захисту даних при їх переміщенні в сховищі даних з оперативних баз даних і зовнішніх джерел, захисту даних при їх передаванні по мережі.

Необхідність наявності багаторівневих довідників метаданих.

Якщо роль метаданих (що зазвичай містяться в таблицях-каталогах) в оперативних інформаційних системах достатньо обмежена, то для сховищ даних наявність розвинених метаданих і засобів їх надання кінцевим користувачам є однією з основних умов успішної реалізації. Наприклад, перш ніж менеджер корпорації задасть системі своє запитання, він повинен зрозуміти, яка це інформація, наскільки вона актуальна, чи можна їй довіряти, скільки часу може зайняти формування відповіді та ін. Детальніше про метадані див. розділ 1.

Потреба в ефективному зберіганні й опрацюванні дуже великих обсягів інформації.

Вже зараз відомі приклади сховищ даних, що містять терабайти інформації. За даними консалтингової компанії Meta Group, близько половини корпорацій, що використовують або планують використовувати сховища даних, припускають довести їх обсяг до сотень гігабайтів. Проблемою таких великих сховищ є те, що накладні витрати на зовнішню пам’ять зростають нелінійно при зростанні обсягу сховища. Дослідження, проведені на основі тестового набору TPC-D, показали, що для баз даних обсягом в 100 гігабайтів буде потрібна зовнішня пам’ять обсягом в 4.87 рази більша ніж потрібно власне для корисних даних. При подальшому зростанні баз даних цей коефіцієнт збільшується.

2.3. Характеристики інтеграції даних

Інтеграція даних – це об’єднання даних, які спочатку вводяться в різні системи. Самі ці системи можуть розташовуватися в одній локальній мережі, але мати різні платформи і внутрішню архітектуру. Така ситуація практично неминуча у всіх підприємствах, що займаються складеним бізнесом. Як правило, один єдиний постачальник не може створити систему, в якій однаково добре вирішені питання бухгалтерського обліку і автоматизації виробничого циклу, керування кадрами і документообігу і так далі. Приклад інтеграції даних з різних джерел подано на рис. 2.1 [18].

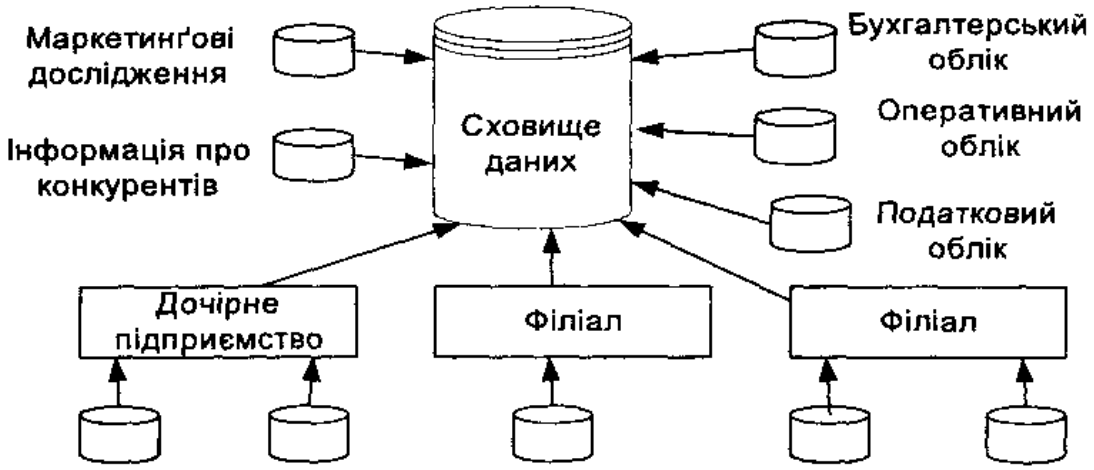


Рис. 2.1 – Інтеграція даних з різних джерел

Крім того, існують завдання, наприклад, маркетингового аналізу, залучення клієнтів, аналізу конкурентного середовища, які за своєю природою вимагають отримання (купівлі) інформації від різних постачальників. Ця інформація постачається у вигляді різноманітних баз даних або електронних таблиць і вимагає завантаження в загальну базу даних для сумісного аналізу.

Метою інтеграції даних є отримання єдиної і цілісної картини корпоративних даних предметної області. Інтеграція даних може бути описана за допомогою моделі, яка включає застосування, продукти, технології та методи:

- технології реалізують один або більше методів інтеграції даних;
- методи – це підходи до інтеграції даних, що не залежать від технологій;
- продукти – це готові комерційні рішення, що підтримують одну або більше технологій інтеграції даних;

- застосування (application) – це рішення, створені постачальниками відповідно до вимог клієнтів, які використовують один або більше продуктів інтеграції даних.

2.4. Методи інтеграції даних

Існує три основні методи інтеграції даних: консолідація, федералізація і розповсюдження (рис.2.2).

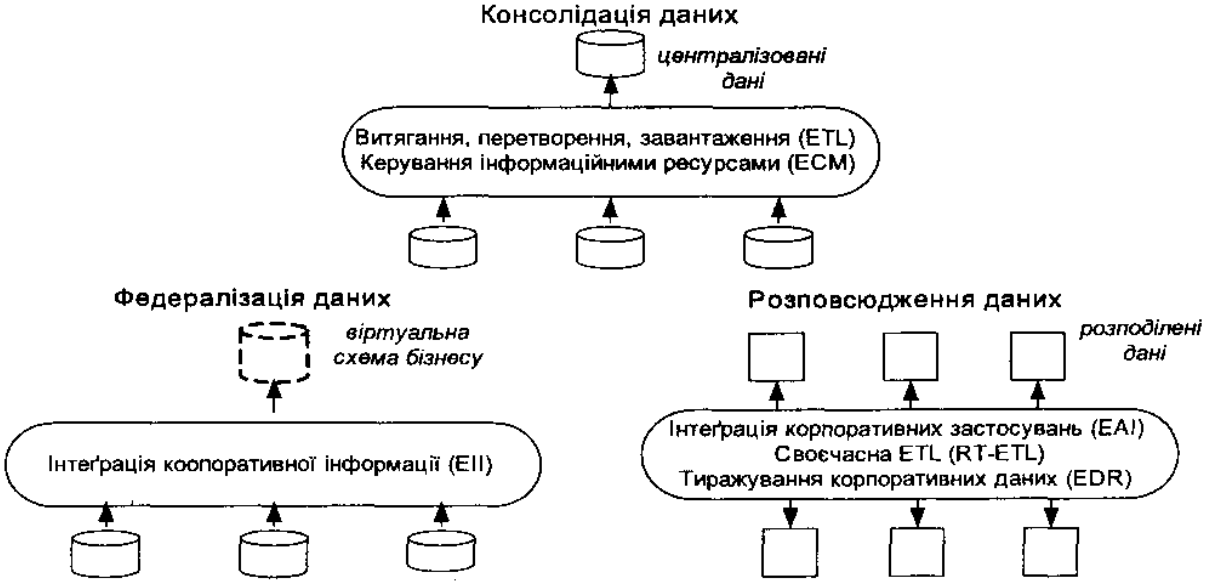


Рис. 2.2. Методи інтеграції даних.

2.4.1. Консолідація даних

Консолідація даних – це збирання даних з територіально віддалених або різноплатформенних джерел даних в єдине сховище даних з метою їх подальшого опрацювання та аналізу. Консолідовані дані необхідні центральному керівництву для того, щоб здійснювати глобальне керування бізнесом, впроваджувати єдину політику у філіалах і здійснювати контроль над їх діяльністю.

Завдання консолідації пояснюється тим, що часто розподілені структури створюються шляхом злиття підприємств, які вже мають деякий рівень автоматизації, навчений певним системам персонал. Тому у багатьох випадках у філіалах працюють різні системи автоматизації. Як правило, підприємства, що входять в холдинг, займаються принципово різними видами діяльності. Наприклад, в деякі холдинги входять банки, страхові й інвестиційні компанії,

підприємства виробництва і транспортні підприємства. Це робить неможливим роботу в єдиному сховищі даних, застосування однотипних систем автоматизації, або однакових налаштувань цих систем. Єдиним способом консолідації даних в цих умовах є застосування розрізнених програм збирання показників звітності або єдиного сховища даних (рис. 2.3).

При використанні цього методу дані збираються з декількох первинних систем і інтегруються в одне постійне місце зберігання. Таке місце зберігання може бути використане для підготовки звітності і здійснення аналізу, як у разі сховища даних, або як джерело даних для інших застосувань, як у разі операційного сховища даних.

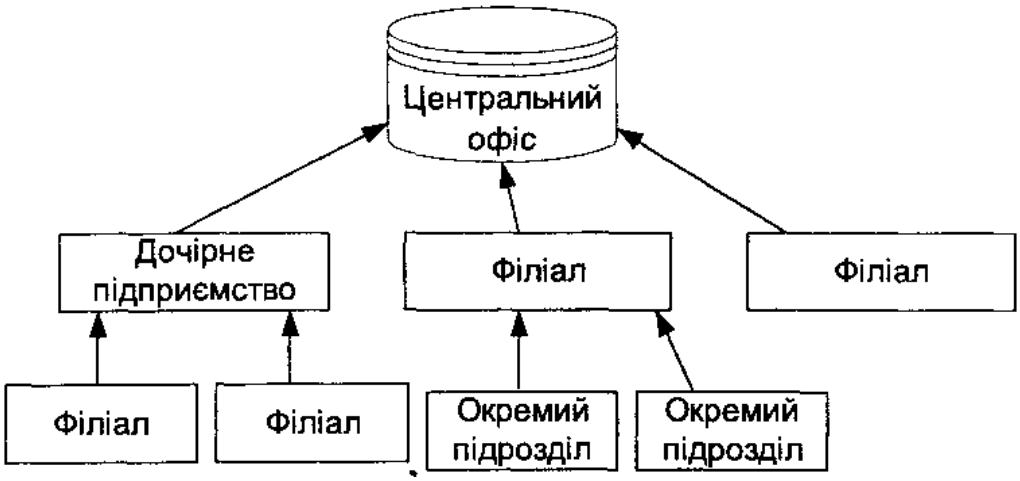


Рис.2.3. Консолідація даних багаторівневого підприємства.

При використанні цього методу зазвичай існує деяка затримка між моментом оновлення інформації в первинних системах і часом, коли такі зміни з'являються в кінцевому місці зберігання. Залежно від потреб бізнесу таке відставання може становити декілька секунд, годин або багато днів. Термін «режим, наближений до реального часу» часто використовується для опису кінцевих даних, оновлення яких відстає від джерела на декілька секунд, хвилин або годин. Дані, що не відстають від джерела, вважаються даними в режимі реального часу, але цього важко досягнути при використанні методу консолідації даних.

Кінцеві місця зберігання даних, що містять дані з великими часом відставання (наприклад, понад один день), створюються за допомогою пакетних застосувань інтеграції даних, які витягують дані з первинних систем з певними, наперед заданими, інтервалами. Такий підхід використовує запити до даних, які одержують періодичні «миттєві знімки» первинних даних. Хоча

подібні запити одержують поточні дані, вони не відображають тих змін, які відбулися між двома послідовними запитами. А за цей час дані могли оновлюватися кілька разів.

Кінцеві місця зберігання даних з невеликим відставанням оновлюються за допомогою *оперативних застосувань* інтеграції даних, які постійно відстежують і передають зміни даних з первинних систем в кінцеві місця зберігання. Такий підхід вимагає від застосувань консолідації даних, щоб вони могли ідентифікувати ті зміни даних, які необхідно зафіксувати для консолідації. Для цього, зазвичай, використовуються певні форми методу захоплення змін даних. У поданому випадку в результаті виконання завдання по захопленню змін будуть одержані всі зміни, які відбулися в первинних даних.

Методи витягання і передавання можуть використовуватися разом. Наприклад, оперативне застосування передавання даних може накопичувати зміни даних в якійсь області проміжного зберігання, а пакетне застосування (прикладна програма) витягання даних може звертатися до нього через певні інтервали. При цьому важливо розуміти, що метод передавання залежить від того, чи відбуваються певні події, а метод витягання працює на вимогу.

Бізнес-застосування, які опрацьовують консолідоване сховище даних, можуть генерувати запити до цих даних, створювати звіти на їх підставі і аналізувати дані. Як правило, ці застосування не можуть оновлювати консолідовані дані через проблеми, пов’язані з синхронізацією оновлення з первинними системами даних. Проте, деякі програмні продукти для інтеграції даних все ж пропонують можливості запису, забезпечуючи засоби вирішення конфліктів даних, які можуть виникати між оновленими даними в консолідованому сховищі і первинними системами.

Деякі застосування оновлюють консолідоване сховище даних і передають ці зміни назад в первинні системи. Прикладом такої системи є кінцеве сховище даних, яке використовується для створення щотижневої моделі ціноутворення. Модель може бути оптимізована і оновлена протягом тижня, а потім знову завантажена в первинну систему на початку наступного тижня.

Перевагою консолідації даних є те, що цей підхід дозволяє трансформувати значні обсяги даних (реструктуризація, узгодження, очищення і/або агрегацію) в процесі їх передавання від первинних систем до кінцевих місць зберігання. Деякі складнощі, пов’язані з таким підходом, – це значні обчислювальні ресурси, які потрібні для підтримки процесу консолідації даних, а також істотні ресурси пам’яті, необхідні для підтримки кінцевого місця

зберігання. Але у зв'язку з постійним вдосконаленням апаратних засобів ця проблема перестає бути важливою.

Консолідація даних – це основний підхід, який використовується застосуваннями сховищ даних для побудови і підтримки оперативних сховищ даних і корпоративних сховищ. Консолідація даних також може знайти застосування для створення залежної вітрини даних, але в цьому випадку в процесі консолідації використовується тільки одне джерело даних (наприклад, корпоративне сховище). У середовищі сховищ даних однією з найпоширеніших технологій підтримки консолідації є технологія ETL (*витягання, перетворення і завантаження – extract, transform, and load*). Ще одна поширена технологія консолідації даних ECM — *керування змістом корпорації* (*enterprise content management*). Більшість рішень ECM напрямлені на консолідацію і керування неструктурованими даними, такими як документи, звіти і web-сторінки. Далі ці технології буде описано детальніше.

2.4.2. Федералізація даних

Федералізація даних забезпечує єдину віртуальну картину одного або декількох первинних файлів даних. Якщо бізнес-застосування генерує запит до цієї віртуальної картини, то процесор федералізації даних витягає дані з відповідних первинних сховищ даних, інтегрує їх так, щоб вони відповідали віртуальній картині і вимогам запиту, і відправляє результати застосуванню, від якого прийшов запит. За визначенням, процес федералізації даних завжди полягає у витяганні даних з первинних систем на підставі зовнішніх вимог. Всі необхідні перетворення даних здійснюються при їх витяганні з первинних файлів. *Інтеграція корпоративної інформації* (*Enterprise Information Integration,* EII) – це приклад технології, яка підтримує федеральний підхід до інтеграції даних [20].

Один з ключових елементів федеральної системи є метадані, які використовуються процесором федералізації даних для доступу до первинних даних. У деяких випадках ці метадані можуть складатися виключно з визначень віртуальної картини, які ставляться у відповідність первинним файлам. У більш передових рішеннях метадані також можуть містити детальну інформацію про кількість даних, що міститься в первинних системах, а також про шляхи доступу до них. Така розширена інформація може допомогти федеральному рішенню оптимізувати доступ до первинних систем.

Деякі федеральні рішення можуть забезпечувати додаткові метадані проблемної області, які відображають семантичні зв’язки між елементами даних в первинних системах. Прикладом таких даних є дані про споживачів. Метадані можуть містити загальний індикатор споживача, який ставиться у відповідність різним ключовим елементам даних про споживача в первинних системах.

Вважається, що основна перевага федерального підходу – той факт, що він забезпечує доступ до поточних даних і позбавляє від необхідності консолідувати первинні дані в новому сховищі даних. Але потрібно пам’ятати, що федералізація даних не дуже добре підходить для витягання і узгодження великих масивів даних або для тих застосувань, де існують серйозні проблеми з якістю даних в первинних системах. Ще один істотний чинник – потенційний вплив на продуктивність і додаткові витрати на доступ до численних джерел даних під час виконання програми.

Федералізацію даних можливо використовувати в тих випадках, коли вартість консолідації даних перевищує переваги для користувачів предметної області, які вона надає. Оперативне опрацювання запитів і підготовка звітів могла б служити прикладом подібної ситуації. Федералізація даних також, ймовірно, виявилася б корисною в тих випадках, коли політика безпеки даних і ліцензійні обмеження забороняють копіювання даних первинних систем. Зазвичай в цю категорію потрапляють синдикати даних. Крім цього, федералізація могла б використовуватися як короткочасне рішення для інтеграції даних пісня придбання або злиття компаній.. Але в цілому, як показано у статті «Интеграция данных и хранилищ» [20], навіть в названих вище ситуаціях консолідація даних часто виявляється більш прийнятним рішенням, ніж федералізація.

Вивчення і профілізація первинних даних, необхідні для федералізації, несильно відрізняються від аналогічних процедур, що вимагаються для консолідації. Отже, організаціям варто використовувати такі продукти для інтеграції даних, які підтримують як федералізацію, так і консолідацію, або, принаймні, продукти, які можуть забезпечувати сумісне використання метаданих, необхідних для обох підходів.

Федеральна архітектура дуже корисна для великих транснаціональних корпорацій і є вельми зручним підходом для підтримки балансу між необхідністю автономії місцевих підрозділів компанії і їх гнучкості, з одного боку, і стандартизації і централізованого контролю, які здійснює центральний офіс, – з іншого. При цьому під федеральним сховищем може розумітися як

єдине фізичне федеральне сховище, так і федерація дрібніших спеціалізованих сховищ даних.

Відзначимо, що в англомовній літературі термін *federated data warehouse* зараз використовується в двох різних значеннях. Частина фахівців має на увазі під федеральним сховищем створення віртуальної структури, що оперує з вибірками даних. Інші називають федеральним сховищем єдиний фізичний репозиторій, що працює з копіями даних, яке іншими словами можна назвати розподіленим сховищем.

Існує багато архітектурних можливостей для інтеграції за допомогою федерального підходу. Але у будь-якому випадку загальний дизайн повинен ґрунтуватися на вимогах до організації і процесів предметної області, як це розглянуто вище.

Існує три основних типи федералізації:

1) за географічною ознакою;
2) за частинами предметної області;
3) функціональна федералізація.

У першому випадку члени федерації виражають вимоги бізнесу кожного географічного регіону, наприклад, Північної Америки, Азії, Європи або окремих країн.

Федералізація за складовими предметної області означає, що члени федерації відображають окремі сфери цієї області.

Функціональний підхід до федералізації будується на основі різних функцій бізнесу, таких як продаж, фінанси, маркетинг і ланцюги постачань.

Концептуально всі три підходи схожі, але реальний дизайн може істотно відрізнятися, оскільки надання повноважень працює по-різному в кожному з цих сценаріїв. Крім цього, можливе і використання єдиного фізичного сховища даних.

2.4.3. Розповсюдження даних

Застосування розповсюдження даних здійснюють копіювання даних з одного місця в інше. Ці застосування зазвичай працюють в оперативному режимі і здійснюють переміщення даних до місць призначення, тобто залежать від певних подій. Оновлення в первинній системі можуть передаватися в кінцеву систему синхронно або асинхронно. Синхронне передавання вимагає, щоб оновлення в обох системах відбувалися під час однієї і тієї ж фізичної транзакції. Незалежно від використовуваного типу синхронізації, метод

розповсюдження гарантує доставку даних в систему призначення. Така гарантія - це ключова ознака розповсюдження даних. Більшість технологій синхронного розповсюдження даних підтримують двосторонній обмін даними між первинними і кінцевими системами. Прикладами технологій, що підтримують розповсюдження даних, є інтеграція корпоративних застосувань (*Enterprise application integration*, EAI) і тиражування корпоративних даних (*Enterprise data replication*, EDR) [18 - 20].

Великою перевагою методу розповсюдження даних є те, що він може використовуватися для переміщення даних в режимі реального часу або близькому до нього. Інші переваги включають Гарантовану доставку даних і двостороннє розповсюдження даних. Доступність багатьох з цих зручностей залежить від конкретного продукту. Метод розповсюдження даних може також використовуватися для урівноваження робочого навантаження, створення резервних копій і відновлення даних, зокрема у разі надзвичайних ситуацій.

Практичне застосування цього методу відрізняється достатньо великою різноманітністю як в плані продуктивності, так і щодо можливостей реструктуризації і очищення даних. Деякі корпоративні продукти розповсюдження даних можуть підтримувати переміщення і реструктуризацію великих масивів даних, тоді як продукти EAI часто мають обмежені можливості переміщення великої кількості даних і їх реструктуризації. Одна з причин подібної відмінності – той факт, що в центрі архітектури тиражування корпоративних даних лежать дані, а в центрі технології EAI – повідомлення або транзакції.

2.4.4. Гібридний підхід

Методи, що використовуються застосуваннями інтеграції даних, залежать як від потреб бізнесу, так і від технологічних вимог. Достатньо часто застосування інтеграції даних використовує так званий гібридний підхід, який включає декілька методів інтеграції. Хороший приклад такого підходу – інтеграція даних про клієнтів (*customer data integration*, CDI), метою якої є забезпечення узгодженої картини інформації про клієнтів.

Найпростіший підхід до CDI – це створення консолідованого сховища даних про клієнтів, яке містить дані, одержані з первинних систем. Відставання інформації в консолідованому сховищі залежатиме від режиму консолідації даних (оперативний або пакетний) і від частоти оновлення цієї інформації.

Інший підхід до CDI – це федералізація даних, коли визначаються віртуальні бізнес-подання даних про клієнтів в первинних системах. Ці подання

використовуються прикладними програмами для доступу до поточної інформації про клієнтів в первинних системах. При федеральному підході також може використовуватися довідковий файл метаданих для зв'язку інформації про клієнтів на основі загальних ключових елементів

2.5. Технології інтеграції

Гібридний підхід, що використовує як консолідацію, так і федералізацію даних, також може мати місце. Загальні дані про клієнтів (ім'я, адреса та ін.) можуть бути консолідовані в одному сховищі, а дані, які відносяться до певного первинного застосування (наприклад, замовлення), можуть бути федералізовані. Такий гібридний підхід може бути розширений за рахунок розповсюдження даних. Якщо клієнт обновлює своє ім'я і адресу під час транзакції в Інтернет-магазині, то ці зміни можуть бути відправлені до консолідованого сховища даних, а звідти поширені в інші первинні системи, такі як база даних про клієнтів роздрібного магазину.

Проблема інтеграції корпоративної інформації, даних і застосувань залишається актуальною для багатьох організацій. Із зростанням обсягу інформації завдання об'єднання розрізнених структур, таких як вітрини, бази або сховища даних, стає життєво важливою для багатьох компаній. Є три технології, які можуть допомогти в цьому – три «I» (або три «E» в англійському варіанті). Це інтеграція корпоративних застосувань (*Enterprise Application Integration*, EAI), інтеграція корпоративної інформації (*Enterprise Information Integration*, EII) і програмне забезпечення для витягання, перетворення і завантаження даних (*Extract, Transform and Load,* ETL).

Ці технології можуть використовуватись для широкого кола завдань: від інтеграції в режимі реального часу до пакетної інтеграції і від інтеграції даних до інтеграції застосувань. Для інтеграції даних в режимі реального часу найкраще підходить технологія EII. Для пакетної інтеграції даних – ETL. А для інтеграції застосувань – консолідація в режимі реального часу або пакетна, найкращим інструментом є технологія EAI.

У статті «Интеграция данных и Хранилища» [18] використані наступні визначення:

EAI – це технологія, за допомогою якої організація добивається централізації і оптимізації інтеграції корпоративних застосувань, зазвичай використовуючи ті або інші форми технології оперативної доставки інформації (push technology), яка керується зовнішніми подіями (event-driven);

ETL – це технологія, яка перетворить дані (зазвичай за допомогою їх пакетного опрацювання) з операційного середовища, що включає гетерогенні технології, в інтегровані дані, що узгоджуються між собою, придатні для використання в процесі підтримки прийняття рішень; ETL-технологія орієнтована на бази даних, наприклад, сховище, вітрину або операційне сховище даних;

EII – це технологія для інтеграції в режимі реального часу незіставних типів даних з численних джерел як всередині, так і за межами корпорації; інструменти EII забезпечують універсальний рівень доступу до даних і використовують технологію пошуку інформації (pull technology) або можливості роботи за запитами; технологія EII орієнтована на конкретних співробітників, які одержують інформацію через інструментальну панель або звіт.

Технологія EAI інтегрує транзакції двох або більше застосувань, технологія ETL інтегрує дані операційних систем і компонентів підтримки прийняття рішень, а технологія ЕП здійснює віртуальну інтеграцію даних з різних джерел. EAI найбільш функціональна тоді, коли необхідно зв’язати застосування у реальному часі для автоматизації процесів предметної області. Другий випадок застосування EAI – це ситуація, коли необхідно, щоб зміни, внесені в одне застосування (зазвичай це невеликий набір записів), були відображені у всіх інших. Ця технологія дуже добре справляється із завданням фіксації змін і їх перенесення у відповідні застосування або системи.

Технологія ETL виявляється найкориснішою в тих випадках, коли необхідно створити сховище даних, що містить добре документовані і надійні дані для історичного аналізу, наприклад, для аналізу часових рядів або багатовимірних запитів. Ця технологія також використовується для інтеграції ключових довідкових даних. Технологія ETL незамінна для таких завдань, як видалення даних, що дублюються, здійснення процесів перевірки якості даних і т.ін. Ці інструменти також використовуються для створення окремих вітрин даних, що обслуговують конкретний відділ або бізнес-процес, або призначених для яких-небудь довгострокових цілей. Інструменти ETL дають користувачу можливість запустити процеси, що повторюються, для більшої злагодженості дій і можливості їх багатократного використання. Такі процеси включають створення точних технічних метаданих, що підтримують загальну цілісність середовища business intelligence (BI).

З активним розповсюдженням Internet процеси ETL стали все ширше застосовуватися і для підтримки Web-застосувань. Наприклад, постачальник

може використовувати засоби ETL для «закачування» у Web-систему даних, необхідних для перевірки стану опрацювання замовлень, з внутрішніх програм. Програми, що реалізують підхід ETL, стають важливими компонентами багатьох ініціатив, пов’язаних з електронною комерцією, зокрема застосувань, що підтримують взаємодію між партнерами бізнесу, а також між виробником і споживачами.

2.6 Технологія ETL

2.6.1. Структура процесу перевантаження даних

Процес.

У загальному випадку, програміст ETL може уявляти собі архітектуру СД у вигляді сукупності трьох областей: джерела даних (сукупність таблиць оперативної системи і додаткових довідників (класифікаторів, таблиць узгодження), що дозволяє створити багатовимірну модель даних з необхідними вимірами), проміжної області (сукупність таблиць, що використовуються виключно як проміжні при завантаженні СД) і приймача даних. Рух даних від джерела до приймача називають *потоком даних*. Необхідні потоки даних формує і описує аналітик (див. рис. 2.4).

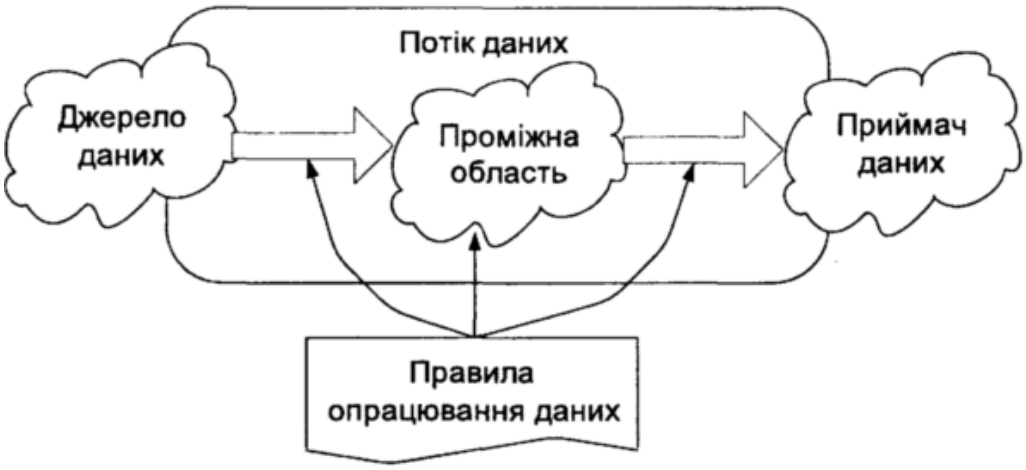


Рис. 2.4. Процедура ETL

Процес перевантаження даних – це реалізація потоку даних від єдиного набору даних джерела до одного або декількох наборів даних СД. Розрізняють наступні класи процесів.

1. За характером завантаження:
- процес початкового завантаження (Initial load);
- процес оновлювального завантаження (Refreshing load).

2. За виглядом джерела даних:
- SCF (джерело даних – стандартний класифікатор Datary, найчастіше – структурований текстовий файл);
- UCF (джерело даних – стандартний класифікатор оперативної системи, призначений для користувача класифікатор);
- MLR (джерело даних – RDS або таблиця фактів оперативної системи),
- DWH (джерело даних – сховище даних).

Процес перевантаження даних включає одну або декілька фаз, які виконуються по черзі, залежно від типу фази.

Фаза.

Фаза процесу перевантаження даних (підпроцес, що забезпечує розв'язування певної задачі в рамках ETL-процесу) відповідає стадії завантаження джерела даних, тобто кількість використовуваних фаз обмежена стадіями, які повинен пройти набір даних джерела, щоб бути завантаженим в СД. Фаза складається з кроків і може включати операції керування виконанням перевантаження. Керування виконанням полягає в аналізі кількості записів в ключових таблицях і прапорів стану, і реалізується за допомогою мови скриптів утиліти SQLExecutor.

Крок.

Кроки – це окремі SQL-запити, які виконують одиничні дії по перевантаженню, перетворенню і вибірці даних. Кожен запит (як і скрипти фаз і процесів) оформляється в окремому файлі відповідно до «Стандарту на оформлення технологічних документів».

Група процесів.

Групи процесів введені для організації періодичних перевантажень даних і вказують чітку послідовність виконання процесів усередині групи.

Стадії завантаження джерела даних.

У процесі завантаження, дані проходять наступні основні стадії, кожна з яких реалізується у вигляді окремої фази процесу перевантаження даних:

№	Назва	Позначення	Опис	Підроцеси
1	Отримання	IDC	Стадія отримання даних з джерела і завантаження їх у проміжну область	Download, Structuring, Refinement, Transfer, Upload
2	Пошук помилок	LOADSTER	Дані перевіряються на відповідність специфікаціям і потенційну можливість завантаження в СД	STER, STAC
3	Перетворення	MLRCONV	Дані групуються і приводяться до вигляду, придатному для СД	STCF
4	Розподіл	MLRDISTR	Дані поділяються на кілька потоків, залежно від способу завантаження в СД	STIN, STUP
5	Додавання	SQLLOAD	Підготовлені дані надходять в СД	INSERT, UPDATE, BACKUP(*), DELETE(**)

* Операція резервування BACKUP в основній масі проектів зі створення дворівневих СД не застосовується. Потрібен подальший розгляд доцільності введення цієї операції взагалі, оскільки вона може бути замінена застосуванням SCD рівня 2 і вище.

** Операція видалення записів з СД в загальному випадку не реалізується через відсутність потреби в ній. Рішення про її створення ухвалюється кожного разу, якщо того вимагає технічне завдання.

Розглянемо основні стадії процесу завантаження даних докладніше [20].

Витягання даних (IDC).

Результатом роботи, на цій стадії, є перевантаження даних з джерела в реляційну СКБД, в проміжну область. Для цієї мети під кожне джерело даних в проміжній області створюється своя таблиця. Всі таблиці мають префікс «sttm» (від «STaging Table for Middle-layer data»), або «stti» (від «STaging Table for Initial load») (також увійшла до ужитку фраза: «Таблиця належить області STTM (STTI)»).

Таблиці, що беруть участь в завантаженні довідників SCF і UCF, найчастіше відносяться до області STTI, оскільки не вимагають написання процедур оновлювального завантаження; а ті, що беруть участь в завантаженні таблиць фактів і оновлюють RDS – до області STTM, оскільки для них

оновлювальне завантаження (завантаження даних в СД з урахуванням наявності даних в СД) майже завжди розробляється.

Проте, іноді зустрічаються джерела інформації, що надходить у різний час або з різних оперативних систем, але ідентичної за структурою. Наприклад, це можуть бути однотипні відомості з різних філіалів. Такі джерела найкраще вважати не різними, а одним розподіленим.

Витягання даних зі всіх частин розподіленого джерела здійснюється в одну таблицю проміжної області. Для збереження інформації, звідки надійшли дані, в структуру цієї таблиці додається поле з позначенням початкової оперативної системи або філіалу.

Отримання (вивантаження) даних (Download).

Найперший етап перевантаження даних - вивантаження інформації з джерела даних до програми-обробника, аналітику або на сервер перевантаження даних. Вивантаження може здійснюватися наступними шляхами, залежно від характеру джерела даних, вимог до організації доступу, інформаційної безпеки, та ін.

Окремим пунктом необхідно виділити керування вивантаженням даних, а саме вказівка глибини вибірки даних за часом. Як правило, досить забезпечити 2 режими роботи процедури вивантаження: вивантаження всієї інформації, без урахування часу її надходження, і вивантаження за деякий останній період (наприклад, за останній закритий день). Універсальним засобом рішення є можливість завдання як параметр-процедури дати, починаючи з якої вибиратимуться дані.

№	Метод організації вивантаження даних	Коментарі
1	Вивантаження з СКБД	Зазвичай не викликає труднощів. Можна використати як «рідні» утиліти, так і Pump.
2	Вивантаження зі структурованого джерела даних	Для вивантаження використовують ODBC, JDBC або відповідні утиліти
3	Вивантаження з не структурованого джерела даних	Якщо джерел неструктурованої інформації не уникнути, необхідне використання додаткових утиліт. У період експорту дані організовуються в жорстку структуру
4	Підготовка структурованого файла з даними людиною або іншою програмою	Подібний до ***2.***

Структуризація даних (Structuring).

Джерела даних, що надходять на вхід ETL-процесів, потенційно мають довільну внутрішню структуру, в загальному випадку далеку від табличної. У зв'язку з цим дані, що містяться в джерелі вимагається спочатку упорядкувати і привести до вигляду, придатного до завантаження в реляційну таблицю за принципом «один до одного». Це може бути один з наступних типів файлів: DBF, TXT (TSV, CSV або файл з вирівняними полями), XML .

Як вже сказано вище, структуризації піддаються тільки дані, які вивантажуються з неструктурованих джерел даних.

Опрацювання даних (Refinement).

Структуровані дані можуть зажадати додаткового опрацювання (очищення, фільтрації, узгодження та ін.) з метою підвищення якості інформації.

В опрацюванні даних на цьому етапі можуть бути задіяні різні інструменти: від ручного виправлення текстового файла аналітиком до утиліт тих, що застосовують новітні методи аналізу даних (нейронні мережі, тезаурус, кластеризацію тощо). Проте, на практиці (можна сказати – на жаль), застосовуються наразі лише прості методи перетворення типів даних, через відсутність потреби проектів в складних і дорогих методах підвищення якості даних.

Пересилання даних (Transfer).

У ході проектування процедур витягання даних необхідно врахувати місцезнаходження джерел даних і умови забезпечення безпеки при пересиланні й опрацюванні даних.

У разі розташування джерел даних на сервері, який розміщений окремо від сервера СКБД проміжної області СД, необхідно забезпечити пересилання даних, вже підготовлених до завантаження в реляційну СКБД, захищеними (довіреними) каналами зв'язку на місце, звідки вони зможуть бути завантажені у відповідну таблицю проміжної області. У зв'язку з широкими можливостями сучасних СКБД по роботі з віддаленими даними, ця проблема є не стільки складною в програмному сенсі, скільки вимагає грамотного адміністрування. У такому разі етап пересилання даних об'єднується з етапом імпорту даних в СКБД (Upload).

І навпаки, дані джерела не завжди можуть бути опрацьовані (підпроцес Refinement) на сервері постачальника даних. Тоді їх необхідно переслати для опрацювання на сервер консолідації даних, що також вимагає наявності

захищених каналів зв'язку і адміністративних ресурсів. У цьому випадку етап пересилання виконується раніше за структуризацію даних.

Імпорт даних в СКБД (Upload).

Для подальшого опрацювання структуровані дані необхідно підвантажити у відповідну таблицю СКБД (яку потрібно заздалегідь очистити).

При цьому існує вірогідність, що окремі записи не зможуть, через фізичні обмеження або несумісність типів даних, бути вставлені. По можливості, такі «невідповідні» записи потрібно зберігати в окремий файл тієї ж структури, що й імпортовані, з метою подальшого аналізу і підвищення якості даних. Такий файл має назву «Файл виключень» і повинен опрацьовуватись додатково.

Опрацювання помилок на стадії IDC.

Помилки можуть з'являтися в будь-якому з підпроцесів стадії витягання даних. Відстежити їх виникнення – завдання слабо формалізоване. Крім того, часто вся відповідальність за забезпечення коректної роботи процедур витягання даних покладається на програмістів, що є неправильним і може бути виправдане тільки для жорстко структурованих джерел даних (наприклад, коли завантаження здійснюється безпосередньо з СКБД).

Вимоги до опрацювання помилок на стадії IDC повинні визначатися залежно від типу джерела даних, але, у будь-якому випадку, обов'язковим є опрацювання фатальних для подальших стадій помилок.

Приклади фатальних помилок:

- відсутність файлів джерела даних;
- помилка доступу до даних;
- виникнення системної помилки ОС.

У разі витягання даних з ODBC-, JDBC-джерела даних або СКБД, застосовується функція Pump програми SQLExecutor. Вона логічно розділяє пересилання даних на дві частини – витягання і додавання даних – і між цими частинами може застосувати програмне опрацювання даних через механізм трансформаторів.

Очищення даних.

Очищення даних полягає у фільтрації тих даних, які, в якому-небудь сенсі, не задовольняють чинні фізичні обмеження або правила предметної області. При цьому дані з таблиці області STTM повністю розділяються на 2 частини (потоки), які згодом опрацьовуються окремо: дані, які не пройшли перевірку, надходять в таблицю з префіксом «ster», а дані, придатні до подальшого опрацювання – в таблицю з префіксом «stac» (рис. 2.5).

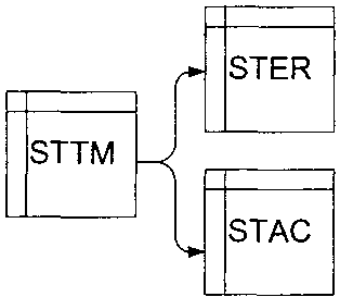


Рис. 2.5. Процедура очищення даних

2.6.2. Підпроцес STER

У потік STER, як вже сказано, потрапляють дані, не придатні до завантаження і СД за якимсь наперед заданим критерієм. Ці критерії визначаються на етапі інформаційного дослідження і перетворяться в SQL-запити розробником ETL.

Категорії критеріїв оцінки якості даних можна звести до наступних класів:

A. За критичністю.
   1. Критичні помилки в даних (дані, які не відповідають цьому критерію, не можуть бути завантажені в СД). Приклад: числовий вираз, що містить букву.
   2. Некритичні помилки в даних (дані, які можуть бути завантажені в СД, але не є якісними). Приклад: порожнє (NULL) значення в полі імені.
   3. Якісні дані.

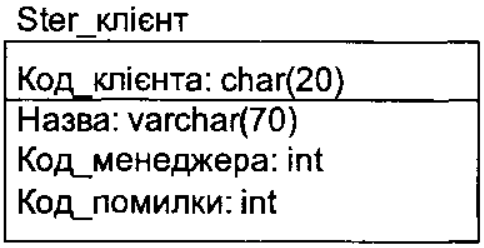
B. За об’єктами, що перевіряються.
   1. Коректність форматів і подань даних.
   2. Унікальність первинних і альтернативних ключів.
   3. Повнота даних.
   4. Повнота зв’язків.
   5. Відповідність даних аналітичним обмеженням.

Можливі декілька варіантів реалізації процедур фільтрації потоку STER:

1. Записи таблиці області STTM опрацьовуються за принципом пріоритету: якщо запис не задовольняє критерій з вищим пріоритетом, то він не перевірятиметься на відповідність решті критеріїв з нижчим пріоритетом, і заноситься в набір даних області STER (це може бути як таблиця, так і

логічний файл-перегляд (view), але його фізична структура повністю включає всі поля початкової таблиці STTM).

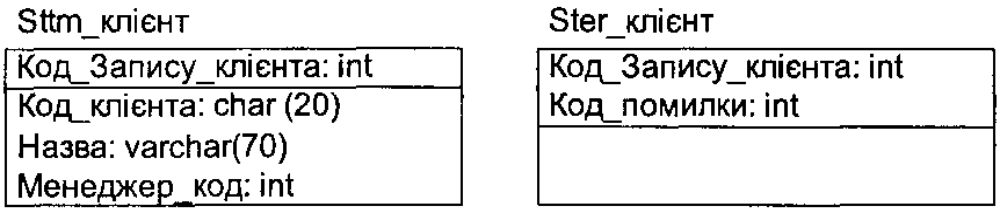
Приклад 2.1 Відношення STER для клієнта подано нижче:



Тут, таблиця *Ster_клієнт* містить в собі поля початкової таблиці *Sttm_клієнт*(*Код_клієнта, Назва, Код_менеджера*) і код помилки якості – *код_ помилки*.

2. Записи таблиці області STTM одержують унікальний ідентифікатор, і таблиця STER формується за принципом «тестового листа» – як карта відповідності критерію перевірки якості даних і ідентифікатора запису.

Приклад 2.2 Відношення STTM для клієнта подано нижче:



Тут, початкова таблиця *Sttm_клієнт* містить додаткове поле *Код_Запису_клієнта*, яке містить унікальний номер запису в цій таблиці. У таблицю *Ster_клієнт*, в результаті виконання запитів якості, потрапляє лише цей номер і номер виявленої в цьому записі помилки.

Переваги другого методу у тому, що він дає повне подання про всі аспекти якості вхідних даних, і використовує простішу форму таблиць області STER. Проте, він вимагає рішення задачі заповнення *Код_запису* для кожної таблиці STTM.

Для проектів, де якість даних не є складовою основних вимог, потік STER просто не виводиться, і процедури перевірки даних не виконуються.

2.6.3. Підпроцес STAC

Таблиця області STAC містить дані, що повторюють вхідні по сховищах поля, але які вже пройшли всі фільтри на етапі заповнення STER, і, отже, визнані якісними. Розроблення подальших процедур здійснюється з урахуванням того, що дані області STAC не «пропадуть» і не «почнуть» двоїтися при об'єднанні, задовольнятимуть правила функціонування предметної області і обмеження на формат даних.

Формування таблиці STAC може здійснюватися за принципом «STTM мінус STER», або накладенням всіх фільтрів одночасно. Перший спосіб використовують переважно через очевидний виграш у швидкості (рис. 2.6).

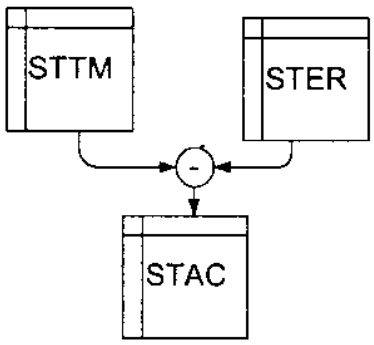


Рис. 2.6. Формування таблиці STAC. Спосіб 1

Перетворення даних.

Не секрет, що фізична модель СД часто не співпадає зі структурою оперативних джерел даних. Це викликано підвищеними вимогами до якості структуризації інформації й особливостями моделювання СД. Тому виникає потреба в перетворенні даних, які надходять з оперативних джерел в структури, відповідні таблицям СД.

Перетворення даних зводиться до декількох елементарних операцій:

- обчислення;
- агрегація;
- отримання статистичних даних (статистика);
- узгодження ключів;
- генерація сурогатних ключів.

Обчислення – це операція, результат якої функціонально залежить від її параметрів, і сама операція може бути реалізована на рівні скалярних функцій.

Агрегація – це операція, яка реалізується як функціональна залежність на рівні агрегатних функцій (функції ряду).

Статистика – операція отримання даних на основі кількісної і/або історичної інформації; результат статистичної функції безпосередньо не залежить від конкретних значень полів в таблиці.

Генерація сурогатних ключів – операція зіставлення первинного ключа (найчастіше – складного) і унікального сурогатного ключа – ідентифікатора набору даних СД. Рішення про способи реалізації приймаються в кожному випадку окремо. Найчастіше застосовуються наступні способи: послідовна нумерація (сурогатний ключ кожного нового природного ключа отримується збільшенням того, що існує, максимального з відомих сурогатних ключів на одиницю) і кодування природного ключа (сурогатний ключ обчислюється з природного за допомогою функціональної залежності).

Узгодження ключів – операція приведення ідентифікаторів набору даних джерела до вигляду, що використовується ідентифікаторам СД. Узгодження ключів виконується найчастіше за допомогою карт відповідності ідентифікаторів (IDMAP).

Результат всіх перетворень надходить в таблиці області STCF, структура яких повторює структуру цільових таблиць СД, за винятком службових полів, існування яких виправдане тільки в СД. Обмеження (constraints) в таблицях області STCF не створюються, оскільки якість інформації повинна бути перевірена повністю на етапі очищення даних.

Очевидно, що таблиць STCF повинно бути використано стільки ж, скільки цільових таблиць у певного процесу перевантаження.

Розподіл даних.

Розподіл даних на декілька потоків перед вставлянням в СД потрібний для того, щоб розділити нові дані і записи, які повинні відновити або доповнити інформацію, що раніше надійшла в СД.

Записи можуть розподілятися з STCF на «нові» (область STIN) і «повторні» (область STUP) різними методами.

Найпростіший, але, в той же час, і найбільш ресурсоємний спосіб – повне порівняння записів з STCF з даними, які вже містяться в СД, при цьому як параметри порівняння використовується наявна ключова інформація (первинні і альтернативні ключі).

Альтернативою є використання ознак модифікованих даних або полів дати-часу останньої модифікації запису, якщо, звичайно, такі поля є в джерелі даних і містять інформацію високої якості.

Фізична структура таблиць областей STIN і STUP, також як і STCF, повністю повторює структуру цільових таблиць СД.

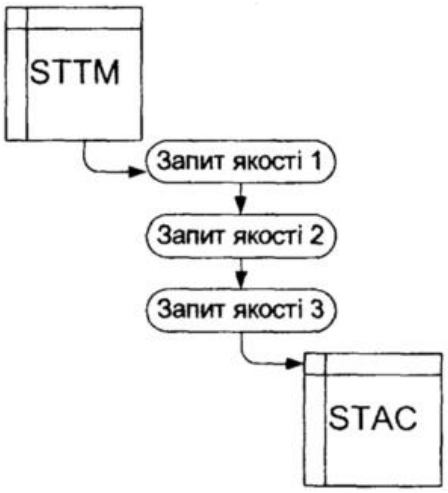


Рис. 2.7. Формування таблиці STAC. Спосіб 2

Додавання та оновлення даних.

Завдяки тому, що дані розподілені за потоками вставляння та оновлення, завантаження даних в СД проходить для всіх таблиць обох потоків простими запитами, без додаткової фільтрації. Цільовими таблицями для цієї стадії є таблиці СД.

Додавання даних здійснюється простим копіюванням записів з таблиці STIN в таблицю СД. Оновлення даних СД з таблиці області STUP здійснюється відповідно до вимог ведення історії даних. Безпосередньо сама операція оновлення може виконуватися або запитом UPDATE, або парою запитів DELETE і INSERT.

Оптимізація перевантаження даних.

Процес перевантаження даних джерел в сховищі даних з технічної точки зору є послідовністю SQL-запитів до СКБД над досить великими обсягами даних (від 1 до 100 мегабайтів за один сеанс). Тому виконання неоптимізованих процесів перевантаження може на порядки збільшити час виконання за рахунок зайвих, повторних опрацювань або пересилання даних. Оптимізація ETL повинна виконуватися строго після відлагодження.

Оптимізація очищення даних.

При очищенні даних здійснюється перевірка кожного запису на відповідність ряду наперед вибраних критеріїв і правил.

Спосіб 1.

Оскільки перевірка одних критеріїв може залежати від результатів перевірки інших, то рекомендується за результатами перевірки критеріїв з вищим пріоритетом (у наведеному прикладі перевірка, чи є значення поля числом, має вищий пріоритет) формувати проміжні (тимчасові) таблиці, які потім перевірятимуться на відповідність іншим критеріям. Як результат, запит на перевірку кожного подальшого критерію опрацьовуватиме все менший обсяг даних.

Цей спосіб має і недоліки: відсутність виграшу під час надходження якісних даних, і навіть програш у швидкості за рахунок потреби в очищенні проміжних таблиць; збільшення кількості об'єктів проміжної області; збільшення кількості кроків.

Для часткового усунення цього недоліку, формування додаткових проміжних таблиць необхідно вводити тільки для перевірки критеріїв, які займають значний час.

Спосіб 2.

Для аналізу критеріїв, час перевірки яких невеликий, можна використовувати логічні файли (подання). Це дозволить зменшити час виконання фази за рахунок відсутності зайвих пересилань даних в проміжні таблиці і скасувати деякі кроки.

Проте, необхідно, щоб подання створювалися прямо у фазі очищення даних, чи ж створювати їх так, щоб вони не накладали обмежень на дані в базовій таблиці.

Спосіб 3.

Третій спосіб полягає у використанні другого варіанту завантаження STER – з перенумерованими записами - і заповненні таблиці відповідності між номерами записів і критеріями, які вони не задовольняють. Такий підхід дозволить реалізувати поданнями як таблиці STER, так і таблиці STAC.

Оптимізація процедури формування сурогатних ключів.

Формування сурогатних ключів за принципом послідовної нумерації у разі використання SQL-запиту для цієї операції – надзвичайно ресурсоємна процедура, оскільки вона створює і опрацьовує тимчасову таблицю, обсяг якої дорівнює половині квадрата від кількості записів початкової таблиці. Тому в деяких випадках, при плановому надходженні великої кількості записів в кожному завантаженні (понад 1E4), необхідно розглянути альтернативні

варіанти процедури генерації сурогатних ключів. Наприклад: кодовані ключі, збережені процедури, автонумерація.

Оптимізація фази вставляння даних в СД. Потік вставляння (STIN).

Додавання даних в СД відбувається через таблицю STIN, дані в яку потрапляють з таблиці області STCF. Проте, часто перевантаження записів з STCF в STIN подається як SQL-запит з об'єднанням з цільовою таблицею, а перевантаження з STIN в СД - просте копіювання. Можна об'єднати ці кроки і заповнювати СД безпосередньо з таблиці області STCF.

Потік оновлення (STUP).

Операція оновлення UPDATE – одна з найповільніших в РСКБД. Один з методів обійти використання UPDATE – заміна її на операції видалення і вставляння.

Можливі два варіанти заміни.

1. Стандартний:
   a) видалення з СД рядків, які є в STUP;
   b) додавання всієї таблиці STUP в СД.
2. Оптимізований:
   a) додавання в STUP рядків, яких немає в STUP, але є в СД;
   b) очищення всієї таблиці СД;
   c) додавання в СД всієї таблиці STUP.

У другому випадку, прискорення може бути досягнуте за рахунок застосування нежурналізованого запиту на видалення даних.

Подібна заміна буде ефективна за великої кількості полів таблиці СД, що оновлюються (понад 10). Проте, ця заміна неможлива для випадків, коли обмеження цілісності (reference constraints) створені фізично в базі .даних. для таких випадків UPDATE – єдиний спосіб оновлення даних.

Оптимізація фази перетворення.

В описі фази перетворення даних мова йшла про те, що вже на цій фазі здійснюється виділення даних з однієї таблиці «на вході» в таблиці, аналогічні сутності сховища даних, «на виході». Проте, легко помітити, що цей метод, відповідаючи традиціям поліпшення супроводжуваності коду, є неоптимальним з погляду кількості таблиць, запитів і, як наслідок, часу завантаження.

Раціональнішим підходом буде заповнення однієї таблиці STCF з вже перетвореними ключами, а розділення даних на різні сутності залишити на етап розподілу даних (STIN і STUP). Потенційно це може дати прискорення як

розроблення, так і виконання процедур завантаження (до декількох разів) без погіршення супроводжуваності коду.

Проте, такий спосіб може застосовуватись не у всіх процесах.

Приклад 2.1. Початкова таблиця з даними продажу (див. рис.2.8) вносить записи в різні за ключами таблиці покупців (клієнтів) і фактів продажу. Для цього вводяться сурогатні ключі покупців і продажу. Організація єдиної таблиці STCF не дозволить працювати стандартній процедурі Генерації сурогатних ключів, оскільки записи про покупців можуть дублюватися. Тому необхідні дві різні таблиці STCF – для кожної з цільових таблиць СД.

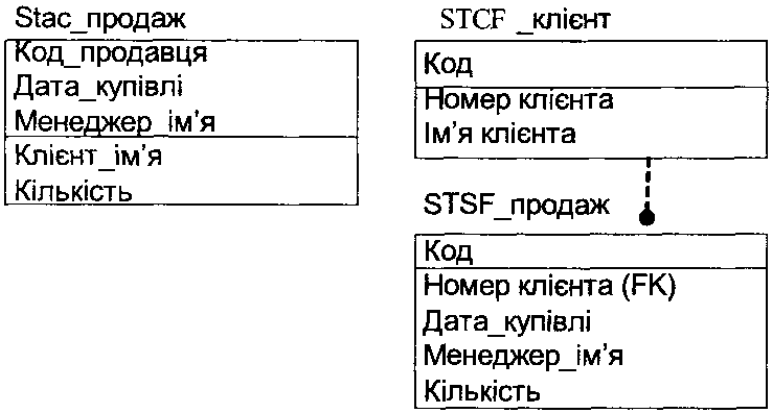


Рис.2.8. Використання двох таблиць STCF

Отже, оптимізація фази перетворення вищеописаним способом можлива тільки тоді, коли первинний або альтернативні ключі таблиці STCF відповідатимуть первинним або альтернативним ключам цільових таблиць сховища даних.

У складаних випадках рекомендується дотримуватися неоптимізованої схеми розроблення для кращої супроводжуваності і зменшення впливу «людського чинника» на якість розроблення.

На закінчення відзначимо, що ETL – процес для розв'язування багатьох завдань є вузьким місцем концепції сховищ даних і за оцінкою InfoWorld, при побудові сховища даних найбільші витрати, як правило, припадають саме на етап ETL. Правильний підхід в реалізації процесів ETL дозволить істотно оптимізувати витрати при побудові сучасного аналітичного інформаційного комплексу і підвищити його ефективність.

2.6.4. Переваги та недоліки ETL

До переваг ETL відносять:

- великий вибір засобів реалізації (Oracle ETL Manager, SQL Server 2005 Integration ServicesActa, CopyManager (Information Builders), DataStage (lnformix/Ardent), Extract (ETI), PowerMart (Informбtica), DecisionBase (CA/Platinum), DataTransformationService (Microsoft), MetaSuite (Minerva/ Carleton), SagentSolutionPlatform (Sagent), WarehouseAdministrator (SAS));
- багато готових інтерфейсів для джерел;
- високу продуктивність;
- ефективне використання устаткування.

Недоліками ETL є:

- небагато засобів мають достатньо високу продуктивність;
- вибухове навантаження на віддалені засоби;
- обмежений вибір засобів;
- не всі перетворення ефективно реалізуються за допомогою SQL.

2.7. Технологія EII

Технологія EII найкраще підходить в тих випадках, коли необхідно створити загальний *шлюз* (*gateway*) з єдиними мовою і точкою доступу до неузгоджених джерел даних. Такі інструменти надають застосуванням і кінцевим користувачам можливості гнучкішого, а також незапланованого доступу до даних, при цьому не вимагаючи постійного використання даних або довготривалих цілей для отримання цього доступу. Крім традиційних реляційних баз даних, інструменти EII можуть працювати з XML- і LDAP-файлами, плоскими файлами й іншими нереляційними даними. Ці інструменти також здатні подавати реляційні дані у форматі XML або форматі web-сервісів. Особливо корисні інструменти EII, якщо є необхідність застосування до довідкових даних сховища додаткові деталі, зокрема, детальну інформацію у реальному часі (наприклад, зіставлення історичних даних з поточною ситуацією).

Окрім розуміння того, коли необхідно використовувати ці технології, потрібно також знати і проблеми, які їм властиві. По-перше, впровадження цих

технологій вимагає від IT-персоналу глибокого розуміння тих вимог, які ставляться до даних для прийняття як тактичних, так і стратегічних рішень. Стосовно технології ETL це означає, що необхідні дані витягуються, перетворюються і завантажуються у вигляді, придатному для використання безпосередньо аналітиками або EII-сервером. У разі EII-технології способи подання даних повинні задовольняти звітні вимоги аналітиків, тобто дані повинні бути придатні для використання в аналітичних звітах. У всіх випадках розуміння джерел даних і вимог, що ставляться до даних, є необхідним кроком при впровадженні цих технологій і безумовно виправдовує той час, який доводиться витрачати, щоб досягти цього розуміння.

Крім того, необхідно розуміти, що впровадження цих інструментів в архітектуру, що вже склалася, вимагає від бізнес- і IT-персоналу розроблення такої стратегії керування даними і застосуваннями, яка буде постійно підтримувати цей процес в активному стані. Обов'язковою складовою такої стратегії має бути усвідомлення того, що підвищується важливість механізмів архівації, а також того, що із самого початку повинні бути створені контрольні журнали. Це необхідно для забезпечення злагодженості і надійності інтегрованих даних і застосувань.

І нарешті, дуже важливий постійний моніторинг продуктивності і ефективності цих технологій в умовах конкретної інфраструктури. їх продуктивність в значній мірі залежатиме від швидкості архівації даних, розмірів і детальності даних, а також від ефективності функціонування системи в умовах повного навантаження. При визначенні продуктивності також необхідно оцінити вплив, який ці інструменти можуть мати на операційні застосування і системи. Тому необхідний постійний моніторинг і цього впливу.

2.8. Технологія EAI

EAI – це багатогранна технологія для сховищ даних, яка охоплює всі рівні корпоративної системи – її архітектуру, апаратне і програмне забезпечення і процеси. EAI означає здійснення інтеграції на наступних рівнях:

***Інтеграція процесів бізнесу* (*Business Process Integration, BPI*).** При інтеграції процесів бізнесу компанія повинна визначати, реалізовувати і керувати процесами обміну корпоративною інформацією між різними системами бізнесу. Завдяки цьому організація може спростити операції, скоротити витрати і поліпшити реагування на запити клієнтів. Елементи

включають керування процесами, моделювання процесів і технологічний процес, який охоплює різні завдання, процедури, архітектуру, необхідну вхідну і вихідну інформацію, а також засоби, необхідні для кожного кроку в процесі бізнесу.

***Інтеграція застосувань* (*Application Integration*).** На цьому рівні інтеграції метою є об'єднання даних або функції одного застосування з іншим, завдяки чому забезпечується інтеграція, близька до реального часу. Інтеграція застосувань використовується – це далеко не повний список – для інтеграції B2B, впровадження CRM-систем, які інтегровані з корпоративними серверними застосуваннями, web-інтеграції і побудови web-сайтів, які підтримують численні бізнес системи. Крім того, може бути потрібне проведення спеціальної інтеграції, особливо коли вимагається інтегрувати наявне застосування із знову встановленим ERP-застосуванням.

***Інтеграція даних* (*Data Integration*).** Умовою успішної інтеграції застосувань і процесів бізнесу є інтеграція даних і систем баз даних. Перш ніж приступати до інтеграції, необхідно ідентифікувати (визначити місцезнаходження) і каталогізувати дані, побудувати модель даних. Після закінчення цих трьох кроків дані можна спільно використовувати/поширювати в системах баз даних.

***Стандарти інтеграції* (*Standards of Integration*).** Для забезпечення інтеграції даних необхідно вибрати стандартні формати для даних. Стандартами інтеграції є ті формати, які підтримують використання і розповсюдження інформації та бізнес даних, тобто стандарти є основою для здійснення інтеграції корпоративних застосувань. До них відносяться COM+/DCOM, CORBA, EDI, JAVARMI і XML.

***Інтеграція платформ* (*Platform Integration*).** Щоб завершити інтеграцію систем – базової архітектури, апаратного і програмного забезпечення – необхідно інтегрувати частини гетерогенної мережі, що рознесена. Інтеграція платформ торкається процесів і інструментів, за допомогою яких ці системи можуть здійснювати безпечний і оптимальний обмін інформацією. Як результат, дані можуть безперешкодно передаватися різними застосуваннями. Наприклад, визначення того, як потрібно надійно передавати інформацію з NT-на UNIX-машину, є надзвичайно непростим завданням з інтеграції всієї корпоративної системи.

2.9. Технологія ECM

2.9.1. Поняття ECM

Згідно з визначенням AIIM (www.aiim.org), професійної організації в області керування інформацією, *Enterprise Content Management* (ECM), або керування інформаційними ресурсами підприємства — це набір технологій, інструментів і методів, що використовуються для збирання, керування, накопичення, зберігання і доставки інформації всім споживачам усередині організації. У цілому ECM орієнтується на роботу з неструктурованою інформацією у будь-якому вигляді, включаючи звичні офісні документи у форматі Word або Excel, PDF, а також малюнки, креслення, графіки, скановані зображення і взагалі файли будь-яких форматів, повідомлення електронної пошти, Web-сторінки, відео й іншу інформацію в електронному вигляді. Основне завдання ECM полягає в підтримці повного життєвого циклу інформації (рис. 2.9).

Концепція ECM приваблива тим, що дозволяє швидко здійснити «зшивання» розрізнених інформаційних систем і зв’язати їх на рівні

ECM – не тільки технологія. Це також і методи, регламенти і практики роботи з інформацією, тому процес керування інформацією повністю автоматизувати не можна – в ньому завжди передбачається участь людини. І також не можна звести все різноманіття джерел інформації в одну систему, якою б всеосяжною вона не була. Отже, ECM – це ще і стратегія керування неструктурованою інформацією, яка може бути реалізована тільки при узгодженій і скоординованій роботі різних інформаційних систем, що існують на підприємстві [29].

Рис. 2.9. Структура систем ECM

Концепція ECM приваблива тим, що дозволяє швидко здійснити «зшивання» розрізнених інформаційних систем і зв'язати їх на рівні потоків інформації. Зазвичай, програмісти також можуть зв'язати дві бази даних і автоматизувати операції, але потрібно включити в процес і людину.

Кожен виконавець одержує свої завдання і повідомлення у разі порушення регламенту, а керівники мають можливість контролювати ситуацію. Проте, мало тільки дати завдання людині, потрібна ще інформація для прийняття рішень. Більш того, часто недостатньо тільки оперативної інформації, наприклад, по конкретному замовленню – потрібен доступ до інформаційного сховища, щоб, наприклад, проглянути документи по минулих замовленнях, листування з клієнтом, підняти текст договору і т.ін. Ні аналітики, ні тим більше сама система не зможуть здогадатися, які саме документи знадобляться в конкретний момент користувачеві; їх не можна просто прикріпити до завдання і доставити відразу весь пакет. Ось чому разом із засобами автоматизації процесів бізнесу активно використовуються системи керування документами і інші компоненти загального ECM-рішення (рис. 2.9).

2.9.2. Компоненти ECM-рішення

Integrated Document Management **(IDM)** – керування електронними документами. Це один з базових компонентів, на основі якого виросла сучасна концепція ECM. Чисте IDM-рішення забезпечує зберігання документів і метаданих («карток документа»), версійність, розмежування доступу і ведення історії роботи з документом. IDM-система, за суттю, є електронною бібліотекою. Технології IDM розвиваються понад десять років і вже вийшли на плато стабільності. Істотного прориву більше не очікується і, швидше за все, базова функціональність з керування документами з'явиться на рівні операційних систем, але ще існуватимуть спеціалізовані по галузях IDM-рішення. Лідерами цього ринку є компанії Documentum, Hummingbird, FileNet і OpenText. Рішення Lotus дуже унікальні, щоб їх можна було віднести тільки до класу IDM. До лідерів підтягаються компанія iManage, об'єднана з Interwoven, Stellent, Hyland Software і ще ряд компаній другого ешелону. Microsoft недавно включилася в роботу в цьому секторі, і наразі програмне забезпечення SharePoint ще не досягло серйозних успіхів на цьому терені. SAP пропонує варту уваги систему SAP DMS, але у складі своїх комплексних рішень, тому вона рідко згадується окремо, хоч і заслуговує цього. Також на ринку IDM

працює безліч компаній, що пропонують рішення для середнього і дрібного бізнесу [29].

Web Content Management **(WCM)** – керування інформацією на Web-сайтах. Традиційні ECM-системи, навіть забезпечені Web-інтерфейсом, виявилися важкуваті і надмірні для завдань виробництва і публікації інформації на сайтах. Оскільки вони не зважали на специфіку роботи з Web-вмістом, виник новий клас систем – WCM. Інструменти подібного роду допомагають розподілити обов'язки зі створення змісту між співробітниками і дати їм можливість його публікації. Коли сайт містить навіть декілька сотень сторінок, жоден Web-майстер не в змозі його підтримувати. При зовнішній схожості завдання публікації Web-змісту істотно відрізняються, як, наприклад, оновлення баз юридичних документів Lexis-Nexis або підтримка каталога електронного магазину, тому й інструменти для цих цілей потрібні різні. Наймогутніші корпоративні WCM-платформи пропонують компанії Documentum, FileNet, Interwoven, Stellent і Vignette. Услід йдуть Microsoft зі своїм Content Management Server і OpenText, що купила Gauss. Окремо варто сказати про рішення для середнього ринку, які забезпечують достатню функціональність при помірній ціні: RedDot Solutions, Percussion, Tridon і ін. Також модулі WCM включають в свої пропозиції такі відомі гравці ринку електронної комерції, як BroadVision.

Records Management **(RM)** – керування записами (або керування офіційною документацією). Сьогодні відповідна функціональність є для будь-якого постачальника ECM-рішень у ряді західних країн обов'язковою, завдяки недавно прийнятим актам, які зобов'язують враховувати і зберігати всі електронні документи, що відносяться до ведення бізнесу, навіть листування з клієнтами електронною поштою. У тих країнах, де законодавче регулювання у області електронних документів відстає, RM слід розглядати як корисний засіб для досягнення прозорості бізнесу (в першу чергу, для його власників), а також для проходження сертифікації на відповідність різним стандартам (ISO 9000, CMM і ін.). У нашій країні подібний інструментарій допомагає вести архіви і організовувати діловодство відповідно до рекомендацій вітчизняних стандартів. Більшість вітчизняних систем документообігу, в першу чергу, розв'язують саме задачі категорії RM, забезпечуючи реєстрацію вхідних, вихідних і внутрішніх документів. RM реалізує одну з найважливіших функцій ECM – підтримку повного життєвого циклу документа аж до його списання і знищення. Ще недавно ринок RM-систем існував і розвивався сам по собі, проте, після ухвалення в 2002 році в Сполучених Штатах акту Sarbanes-Oxley,

практично всі незалежні постачальники RM-продуктів скуповувалися провідними гравцями ринку ECM [29]. Тому в списку лідерів можна побачити ті ж самі назви, можливо, дещо в іншому порядку: OpenText, Hummingbird, IBM (купила Tarian Software в 2002 році), Documentum (купила TrueArc в 2002 році) і Vignette (купила Tower Technology в січні 2004 року). У програмних продуктах OpenText і Hummingbird відповідні модулі з'явилися ще в 1999 році. Решті гравців, що тільки відреагували на ініціативу законодавців, ще належить повністю інтегрувати RM-функції в свої рішення.

Business Process Management **(BPM),** ***workflow*** – керування процесами бізнесу і керування потоками робіт. Роль BPM в концепції повного ECM-рішення важко переоцінити. Без інтеграції з процесами бізнесу будь-яка система залишається статичною і служить лише сховищем інформації. Бізнес-логіка «зашита» в головах менеджерів, тому вони витрачають багато часу на виконання рутинних операцій з розміщення замовлень, опрацювання скарг клієнтів, підготовки звітів та ін., тому завдання автоматизації та оптимізації процесів бізнесу відносяться в багатьох проектах до головних і часто пов'язані з проектами із впровадження ERP-систем. Реальний виграш від впровадження ECM-системи можна одержати тільки при її інтеграції з основними бізнес-системами підприємства, тому що головне призначення ECM – активно включити неструктуровану інформацію в процеси бізнесу.

Не можна розглядати BPM і workflow лише як елементи ECM – необхідно вважати ці технології зв'язною ланкою між світом ECM і світом транзакційних систем. Враховуючи граничне положення технологій BPM/workflow, тут декілька груп лідерів. FileNet і Documentum рухалися від ECM, розвинувши свої засоби керування потоками робіт в цілях керування документопотоками. Услід йдуть OpenText і Himimingbird, яка недавно ліцензувала продукти workflow у компанії Drala Software.

Друга група лідерів – компанії SAP і Oracle, у складі ERP-систем яких також є розвинені модулі керування потоками робіт. До третьої, найчисленнішої, групи відносяться незалежні постачальники подібних рішень; серед них необхідно виділити Staffware і Metastorm. Далі йде щільна група компаній, що володіють правильним баченням ринку і добротними технологіями; у їх числі Action Software, Drala Software, Intalio і Ultimus. Є ще велика кількість нішевих гравців, що пропонують власні системи керування потоками робіт. У разі гетерогенних систем, коли є декілька різних ERP-застосувань та ще і документообіг, часто виграшним виявляється використання продуктів від незалежних постачальників.

Collaboration – спільна робота. На відміну від засобів керування потоками робіт, орієнтованих на підтримку формалізованих виробничих процесів, засоби організації спільної роботи дозволяють налагодити взаємодію у випадках, непіддатливих строгій формалізації. Все, що відноситься до творчої роботи, і де присутній переговорний процес, вимагає гнучкого підходу, тому клієнтів, постачальників і партнерів ніхто не включає у внутрішні бізнес-процеси організації. Для взаємодії з ними залишається тільки електронна пошта, що не завжди зручно.

Також важко організувати вільний обмін ідеями й інформацією в командах, зайнятих розробленням нового продукту або послуги (скажімо, підтримати роботу відділу маркетингу). Ось тут і потрібні продукти, які мають достатню функціональність в керуванні документами з одного боку, і можливістю керування процесом – з іншого, забезпечуючи максимальну гнучкість і мінімальне залучення IT-персоналу або зовнішніх консультантів.

З незалежними розробниками продуктів для колективної роботи відбулася та ж історія, що і з постачальниками RM-систем - їх скуповували сильні гравці ринку ECM. Спочатку орієнтованою на організацію спільної роботи з Великої четвірки ECM була тільки компанія OpenText. Компанія Documentum купила eRoom, Hummingbird купила PeopleDoc, Vignette придбала Intraspect. Єдиним помітним незалежним виробником наразі залишається SiteScape, якій свого часу дісталися розроблення AltaVista після поглинання DEC компанією Compaq. Першопрохідцем в області організації спільної роботи був продукт Lotus, який і сьогодні перебуває в числі лідерів. Також активно включилися в гру Microsoft і Oracle з продуктами SharePoint і Collaboration Suite відповідно. Xerox зі своїм DocuShare також обіцяє стати помітним гравцем на цьому ринку.

Головна умова при виборі інструментарію спільної роботи полягає у тому, що постачальники включають в це поняття дуже різні засоби (колективна робота з документами, календарне планування, Web-конференції, миттєвий обмін повідомленнями та ін., що частково дезорієнтує клієнтів).

Knowledge Management **(KM)** – керування знаннями. Цей термін можна трактувати як завгодно широко, тому зупинимося тільки на аспектах, що мають пряме відношення до ECM. Основна проблема, для розв'язування якої використовуються окремі технології з арсеналу KM, пов'язана з великим обсягом інформації, що вимагає могутніх пошукових механізмів. Крім того, практика показує, що не вдається звести всю інформацію в єдине сховище, тому вимагається забезпечити прозорий доступ до різнорідних джерел

інформації. І, нарешті, є завдання, яке в принципі не вирішується без застосування КМ; йдеться про автоматичну категоризацію інформації за змістом документів.

Також існує проблема зручної візуалізації великих масивів інформації. Звична ієрархічна модель вкладеності, на зразок каталогів у файловій системі, неприйнятна: людина не сприймає більше п'яти-семи рівнів. Потрібні інші, більш наочні засоби візуалізації. Без зручних засобів навігації по сховищу, велика частина інформації так і залишиться мертвим вантажем, оскільки людині властиво нашвидкуруч проглядати інформацію і тільки потім вибирати те, що потрібно, а зовсім не посилати системі точний запит. Охарактеризуємо рішення, відповідні кожному з напрямів.

Повнотекстовий пошук. Всі ECM-рішення володіють вбудованою пошуковою машиною, власною або ліцензованою у третіх фірм.

Наскрізний пошук за різнорідними джерелами інформації. Швидше за інших подібне рішення запропонувала компанія Hummingbird, якій дістався продукт Fulcrum після поглинання нею компанії PC DOCS в 1999 році. Недавно в цю сторону рушила і Documentum, купивши технологію AskOnce у Xerox в березні 2004 року.

Автоматична категоризація. Лідерами цього напряму є компанії Autonomy, IBM/Lotus, Inxight і Verity. Проте, для побудови ECM-рішення важлива не тільки якість конкретного продукту, але і, перш за все, можливість його прозорої інтеграції в комплексну систему, тому варто приділити увагу і постачальни¬кам повних ECM-рішень, Documentum і Hummingbird. Не варто випустити з уваги компанії Convera (раніше Excalibur), Inktomi (яка нині належить Yahoo), Mohomine (її купила Kofax), а також таких відомих гравців, як Microsoft і SAS, які розвивають власні технології пошуку і категоризації.

Візуалізація інформації. Тут можна використовувати і портальні технології різних виробників і власні розроблення, але найцікавіші рішення на сьогодні пропонують компанії Inxight і Entrieva.

Digital Asset Management **(DAM)** – керування цифровими активами. На перший погляд, ніякої відмінності між DAM і звичним інструментарієм IDM немає. Система керування документами може зберігати файли в будь-якому форматі, наприклад MP3, AVI або JPEG. Кожен файл забезпечується обліковою карткою. Навіщо ж ще DAM? Проте, не варто розглядати DAM як лише розширення функціональності інструментів керування документами засобами роботи з мультимедіа. DAM оперує даними в електронній формі саме як активами в бухгалтерському розумінні цього слова, ставлячи собі за мету

витягання організацією максимальної вигоди з використання цих активів. Області застосування DAM, природно, пов'язані з тими галузями, де матеріальні цінності існують в електронному вигляді – індустрія розваг, реклама, фотографія, музична продукція, електронні книги тощо.

DAM-рішення пропонують Documentum, що купила Bulldog в кінці 2001 року, Interwoven з купівлею MediaBin в 2003 році, а також OpenText з недавнім придбанням (серпень 2004-го) Artesia, одного з кращих таких продуктів. Також варто згадати компанію INSCI, яка недавно поглинула Web Ware і пропонує тепер своє ECM-рішення з функціональністю DAM. У цілому ж аналітики визнають, що ринок DAM ще не сформувався, а тому для нього характерна деяка розмитість термінології, що утруднює порівняння рішень постачальників.

2.10 Засоби інтеграції даних

2.10.1 Основний інструментарій

До числа основних засобів, що використовуються для забезпечення інтеграції інформаційних ресурсів, відносяться конвертори даних, інтегруючі моделі даних, механізми відображення моделей даних, об'єктні адаптери (Wrappers), посередники (Mediators), онтологічні специфікації, засоби інтеграції схем і інтеграції онтологічних специфікацій [12, 14], а також архітектура, що забезпечує взаємодію засобів, використовуваних в конкретній системі інтеграції ресурсів. Приклад розвиненої інфраструктури, що забезпечує семантичну інтеграцію даних з використанням комплексу названих інструментів, обговорюється в роботі [15].

2.10.2 Архітектура систем інтеграції

У системах інтеграції даних найбільше поширення отримала архітектура з посередником. На посередник покладається завдача підтримки єдиного призначеного для користувача інтерфейсу на основі глобального представлення даних, що містяться в джерелах, а також підтримку відображення між глобальним і локальним представленнями даних. Запит користувача, сформульований в термінах єдиного інтерфейсу, декомпозируется на безліч підзапитів, адресованих до відповідних локальних джерел даних. На основі результатів їх обробки синтезується повна відповідь на запит.

Використовуються два різновиди архітектури з посередником — Global as View і Local as View [16, 17]. Перша з них (Global as View) передбачає

визначення глобального представлення інтегрованих даних в термінах заданих представлень локальних джерел. Такий підхід ефективніший у разі, коли множина усіх використовуваних джерел визначена наперед. Якщо система інтеграції призначена для підтримки повного матеріалізованого представлення інтегрованих даних, процеси конверсії даних з джерел в їх єдине глобальне представлення здійснюються одноразово.

При використанні другого різновиду даної архітектури (Local as View) передбачається, що представлення для кожного з локальних джерел даних визначається в термінах заданого інтегруючого глобального представлення. Хоч у цьому випадку ускладнюється відображення запитів користувача в середу локальних джерел даних, такий підхід допускає динамічність складу множини джерел даних. Кожне нове джерело може підключатися до системи як на стадії розробки, так і на стадії функціонування.

Інтегруючі моделі даних

В якості інтегруючих (що називаються також ***глобальними***) моделей даних для підтримки єдиного інтерфейсу користувача в системах інтеграції найчастіше використовуються звичайні широко використовувані моделі даних, наприклад, реляційна або об'єктна. У зв'язку з розширенням розробок веб-додатків як інтегруючої моделі даних стала широко використовуватися модель, заснована на стандартах XML (см наприклад [3]).

При використанні в різних джерелах цих неоднорідних моделей даних часто для підтримки глобального представлення даних створюється спеціальна досить розвинена інтегруюча модель даних. Експериментальні розробки таких моделей почали проводитися ще з середини 70-х років і ведуться до теперішнього часу.

У розробках інтегруючих моделей даних використовується також підхід, заснований на інтеграції моделей даних, що підтримуються різними джерелами. Такі інтегруючі моделі забезпечують одночасно і рішення подвійної задачі — підтримку безлічі різних представлень одних і тих же даних. Відомі проекти такого роду інтеграції моделей, що відносяться ще до початку 80-х років. Як приклад можна привести спробу інтеграції в єдиній моделі даних можливостей мережевої моделі даних CODASYL і реляційної моделі даних.

До цієї ж категорії засобів інтеграції даних прилягає розробка розширення мови SQL, що завершується нині, — компонента нової версії стандарту мови SQL:200n, яка дістала назву SQL/XML. Засоби SQL/XML забезпечують можливості представлення схем баз даних SQL і реляційних

даних у формі XML-документів, а також реляційне представлення інформаційних ресурсів XML в середовищі баз даних SQL.

Нова технологічна платформа Веб, заснована на стандартах XML, останніми роками привертає увагу багатьох фахівців як ефективний інструмент інтеграції інформаційних ресурсів у багатьох важливих на практиці випадках. Великий інтерес до середовища XML пов'язаний не лише з можливостями XML як мови опису даних, але і значною мірою з можливістю використання його для транспорту повідомлень в середовищі Веб.

Конструктивний інтерес до засобів інтеграції інформаційних ресурсів Веб і реляційних баз даних проявляють і розробники нових інформаційних технологій для «Всесвітньої павутини». Стандарт мови запитів XQuery платформи XML, який розробляється, втілює функціональність, властиву інтегруючій моделі даних. Базова модель даних цієї мови підтримує ієрархічні і реляційні структури даних і, таким чином, забезпечує можливості для інтеграції XML-данных і даних, які містяться в реляційних базах даних. Вона дозволяє в той же час явним чином представляти величезні інформаційні ресурси «прихованого» Веб — бази даних SQL, до яких наразі забезпечується доступ в середовищі Веб за допомогою інтерфейсу HTML-форм.

Механізми відображення моделей даних

Невід'ємним функціональним елементом архітектури системи інтеграції даних є механізм відображення моделей даних [9]. В деяких системах, що забезпечують інтеграцію зовнішніх джерел даних в середу систем баз даних, використовується поняття шлюзу, що є по суті механізмом відображення представлення даних джерела в середу системи бази даних.

Стандартизація такого відображення для баз даних SQL забезпечується специфікаціями SQL/MED. При інтеграції даних в середовищі, заснованому на платформі CORBA, використовуються об'єктні адаптери (Wrappers), що підтримують IDL-интерфейс до інкапсульованих інформаційних ресурсів і дозволяють тим самим «об'єктизувати» необ'єктні ресурси, наприклад, успадковані системи баз даних. Завдяки цьому створюється інтегроване інтероперабельне об'єктне середовище неоднорідних інформаційних ресурсів.

2.10.3 Методи реалізації інтеграції даних

Найбільш поширеними методами реалізації інтеграції даних є:

- обмін на основі файлів;
- реплікація даних;

- технологія Web-сервісів;
- сервіс-орієнтована архітектура (SOA);
- інтеграційні сервери.

Обмін на основі файлів

Обмін файлами є найбільш поширеним способом інтеграції. З точки зору реалізації це найпростіший спосіб, але він має недоліки. У разі потреби обміну складними структурами, потрібна розробка спеціалізованих форматів файлів, що призводить до великої залежності систем один від одного. Також обмін за допомогою файлів має на увазі наявність людини, яка здійснює вивантаження і завантаження файлів. Проте у разі неможливості взаємодії через мережу перенесення даних на фізичному носії є єдиним можливим рішенням.

Зокрема обмін даними між підсистемами і з додатками з пакету MS Office для автоматизованої інформаційної системи «Прокуратура-статистика» був використаний саме це метод. Було необхідно реалізувати обмін статистичними звітами між територіально видаленими підсистемами. Для обміну між вузлами був розроблений внутрішній формат представлення даних в двійковому файлі. А для опису структури звіту для додатків MS Word і MS Excel була використана мова XML для опису структури форм звітності. Це дозволило універсально описати спосіб відображення даних у файлі звіту для будь-якої форми.

Мова XML також використовується в системі ―1С для реалізації обміну даними між різними підсистемами.

Реплікація даних

Реплікація — процес приведення даних електронних таблиць двох БД в ідентичний стан. Процес реплікації заснований на поняттях “видавець” і “передплатник”. Видавцем є сервер публікації, тобто сервер, що відправляє інформацію. Передплатником є відповідно приймаючий сервер — сервер підписки.

Реплікації зручно розбити на дві великі категорії: реплікація даних в середовищі між серверами і реплікація даних між сервером і клієнтами. Реплікація даних між серверами виконується для підтримки наступних застосувань і вимог :

Підвищення масштабованості і доступності

Обслуговування безперервно оновлюваних копій даних дозволяє масштабувати операції читання на декілька серверів. Надмірність, яка виникає

в результаті обслуговування декількох копій одних і тих самих даних, критично важлива для планового і непланового обслуговування системи.

Зберігання даних і складання звітів

Сервери сховищ даних і сервери звітів часто використовують дані з інтерактивних обробки транзакцій (OLTP). Використовуйте реплікацію для переміщення даних між серверами OLTP і системами підготовки звітів і підтримки рішень.

Об'єднання даних з декількох вузлів

Дані з віддалених офісів часто накопичуються і об'єднуються в центральному офісі. Аналогічним чином, можна реплікувати дані у віддалені офіси.

Об'єднання різнорідних даних

Деякі застосування залежать від даних, що посилаються з баз даних або у бази даних, відмінні від SQL Server. Використовуйте реплікацію для об'єднання даних з баз даних, відмінних від SQL Server.

Розвантаження пакетної обробки

Пакетні операції часто бувають занадто ресурсоємними для виконання на сервері OLTP. Використовуйте реплікацію для перенесення обробки на виділений сервер пакетної обробки.

В нинішній час використовується ***технологія Web-сервісів***, яка подається як зручний засіб інтеграції додатків, що дозволяє легко реалізувати міжплатформену взаємодію. Сервіси являють собою об'єкти, що знаходяться в певних стосунках і мають вкладену в них функціональність. Будь-який Web-сервис можна розглядати як чорний ящик, що має вхідний і вихідний порти. Але технологія Web-сервісів не може розглядатися як загальний підхід до інтеграції додатків з кількох причин: Web-сервисы непридатні для обробки великих об'ємів даних; у них відсутні засоби підтримки транзакцій; у момент взаємодії інтегровані системи повинні знаходитися в працездатному стані. Непридатність для обробки великих об'ємів інформації пов'язана з тим, що усі дані переводяться у формат XML, що веде до збільшення об'єму даних і навантаження на систему в цілому. Web-сервисы починають «захлинатися», коли за одну взаємодію треба передати сотню кілобайт. Існує стандарт, WS-Transaction, який має розв'язати цю проблему, але доки повноцінні його реалізації відсутні. Так, наприклад, WS-Transaction підтримується IBM у своїх продуктах, заснованих на WebSphere Application Server, але з накладанням маси обмежень, основним з яких якраз є необхідність роботи під управлінням WebSphere Application Server. Те ж саме можна сказати і про .Net 3.0 від

Microsoft. Що ж до необхідності роботоздатності застосувань, що взаємодіють, то можна сказати, що від початку віддалені виклики були придумані не для інтеграції додатків, а для реалізації розподілених систем, коли компоненти однієї системи працюють на різних машинах.

Дуже поширеною є архітектура, орієнтована на сервіси (Service Oriented Architecture, SOA). ***Сервіс-орієнтована архітектура*** (Service-Oriented Architecture або SOA) — підхід до проектування, розгортання і експлуатації розподілених програмних систем. Система, реалізована по принципах SOA, є сукупністю програмних компонентів, а саме сервісів, що мають стандартні інтерфейси для доступу до них за допомогою мережі і використання цих компонентів. Інтерфейси в SOA незалежні від платформ розгортання сервісів і технологій їх реалізації. Інтерфейс — це ключове поняття сервісу. За допомогою їх виконується представлення можливостей сервісу зовнішньому світу і організація взаємодії сервісів. SOA пропонує єдину схему взаємодії сервісів, незалежну від того, де знаходиться сервіс. Сервісом може виступати як ціле застосування, так і окремий його модель. Передбачається, що сервіс виконує яку-небудь бізнес-дію (business concept), якусь окрему функцію. Завершивши свою роботу, сервіс може викликати інший сервіс. У сервісній архітектурі немає жорстких зв’язків між модулями. Їх замінюють, так званою, слабкою зв’язністю компонентів (loose coupling). Використовуючи такий зв’язок можна на ходу збирати з сервісів таку конфігурацію, яка потрібна в даний момент. Так само разом із звичайними даними при зверненні до сервісу модуль, що його викликає, який знає, як його викликати і яку роботу він виконує, передає метадані. Використовуючи сукупність даних і метаданих, сервіс виконує замовлення на обслуговування [5]. SOA – це не якісь окремі продукти, а технологія побудови інформаційних систем, які представляють з себе сервіси, доступні для зовнішнього використання. Отже, SOA базується на наступних принципах:

- слабка зв’язність компонентів, яка дозволяє виконувати зміну усередині сервісу, не зачіпаючи програмні модулі, що його використовують;
- «Крупнозерниста» (coarse-grained) структура сервісів, яка забезпечує, завдяки тому, сервіси — модулі бізнес-логіки високого рівня, які не мають потреби в наявності безлічі низькорівневих викликів, що враховують архітектуру сервісів, понизити навантаження на мережу і підвищити продуктивність;

- Стандартні інтерфейси (standards-based). Завдяки нейтральності по відношенню до використовуваної апаратної платформи, забезпечується універсальність взаємодії сервісів в різнорідному середовищі і зниження витрат на інтеграцію.

Simple Object Access Protocol (SOAP) — для обміну даними.

Технологія SOA активно використовується при реалізації інтеграції даних у банківських системах.

Інтеграційні сервера являють собоє проміжні сервери в інтеграційному середовищі, шлюзи, що здійснюють обробку потоків даних і повідомлень, а також розподіл даних між додатками, що мають різні інтерфейси. У ядрі інтеграційного сервера повинні зберігатися бізнес-правила, на основі яких, а також отриманих даних, виконуються обчислювальні операції, аналіз і ухвалення рішень. Використання брокерів дозволяє винести правила інтеграції з коду конкретних систем в окремі бізнес-правила, що спрощує зміну цих правил, веде до їх відкритості і застосування до усіх взаємодіючих систем. Також на брокери покладаються функції адресації і перетворення даних.

Прикладом програмного забезпечення, реалізованого відповідно до технології серверів, може служити IBM WebSphere Message Broker.

2.10.4 Магічний квадрант Gartner

Для оцінки постачальників будь-якого сегмента ринку інформаційних технологій, Gartner використовує дві лінійні прогресивні експертні шкали:

• повнота бачення (англ. Completeness of vision);
• здатність реалізації (англ. Ability to execute).

Кожен постачальник, який потрапив в рамки розгляду для досліджуваного сегмента ринку, оцінюється за цими двома критеріями. При цьому, повнота бачення відкладається на осі абсцис, здатність реалізації – на осі ординат. Кожен постачальник, таким чином, виявляється в одному з чотирьох квадрантів площині, званих:

- лідери (англ. Leaders) – постачальники з позитивними оцінками як за повнотою бачення, так і за способами реалізації,
- претенденти (англ. Challengers) – постачальники з позитивними оцінками тільки за можливості реалізації,
- провидці (англ. Visionaries) – постачальники з позитивними оцінками тільки за повнотою бачення,

- нішеві гравці (англ. Niche players) – постачальники з негативними оцінками за обома критеріями.

Gartner називає магічним квадрантом (по алюзії на магічний квадрат) конкретний аналіз будь-якого сегмента ринку, з розподілом постачальників за вказаними чвертях; щорічно компанія випускає кілька десятків магічних квадрантів на регулярній основі. Постачальники іноді відзначають навіть сам факт потрапляння в будь-якої магічний квадрант окремим прес-релізом як визнання ринкових досягнень, навіть якщо компанія згадана лише в квадраті нішевих гравців.

Розглянемо для початку, якими критеріями керується Gartner Inc. вибираючи на ринку ту чи іншу компанію і її програмні продукти, пов'язані з ринку інструментів інтеграції даних.

Перше, програмний продукт повинен мати можливість бути застосовний в різних **сценаріях інтеграції даних, таких як:**

- ***Збір Даних для Сховищ Даних*** і рішень бізнес-аналізу (BI);
- ***Консолідація та Доставка Майстер-Даних*** і підтримка рішень з управління майстер-даними (MDM, Master Data Management);
- ***Міграція і Перетворення Даних*** при переході з успадкованих додатків на нові програмні продукти, або при здійсненні консолідації даних, викликаної об'єднанням або поглинанням компаній;
- ***Синхронізація Даних*** між оперативними програмними додатками і забезпечення цілісності та узгодженості даних, як в локальних, так і в хмарних додатках;
- ***Обмін Даними*** з зовнішніми компаніями (партнерами, клієнтами, постачальниками, і ін.);
- ***Надання «Інформаційних Послуг»*** в сервіс-орієнтованих архітектурах (SOA).

Так само Gartner визначає перелік функціональних можливостей, якими інструменти інтеграції даних повинні володіти і надавати користувачам:

- ***Connectivity Capabilities*** –Можливість з'єднуватися з широким спектром типів джерел (*data source*) і приймачів (*data target*) даних, від традиційних і успадкованих реляційних і нереляційних баз даних, файлів різних форматів, XML-даних і черг повідомлень до промислових стандартних форматів повідомлень (таких, наприклад, як Swift і HL7), неструктурованих даних, розподілених файлових систем (таких як Hadoop DFS) і інших NoSQL сховищ;

- ***Різні Режими Взаємодії*** з перерахованими структурами даних;
- ***Data Delivery Capabilities*** – Можливість доставляти дані в різних режимах;
- Можливість задовольняти ***Вимоги до Затримки*** **(*Latency Requirements*)** та здійснювати обробку даних в запланованому пакетному режимі, потоковому близькому до реального часу режим, режим подієвої доставки даних;
- ***Data Transformation Capabilities*** – Можливість, як основних традиційних *трансформацій даних*, так і розробки середніх і дуже складних перетворень даних, вирішальних специфічні потреби конкретного бізнесу.
- ***Metadata and Data Modeling Capabilities*** – Можливості управління метаданими і здійснення моделювання;
- ***Можливості для Розробки – Design and Development Environment Capabilities*** – включаючи графічне представлення об'єктів, моделей і потоків даних, управління робочими процесами і підтримку тестування і налагодження
- ***Можливості Підтримки Управління Даними – Data Governance Support Capabilities*** – включаючи механізми профілювання і поліпшення якості даних;
- ***Deployment Options and Runtime Platform Capabilities*** – *Можливості Розгортання і Виконання*, що включають підтримку різних програмних і апаратних платформ, віртуалізації, моделі SaaS, і підтримку паралельної розподіленої обробки (наприклад, Hadoop, MapReduce).
- ***Можливості Управління та Адміністрування – Operations and Administration Capabilities*** – включаючи обробку помилок, моніторинг і контроль виконання процесів, збір статистики виконання, контроль безпеки і т.п.
- ***Архітектурні Можливості та Можливості Інтеграції – Architecture and Integration Capabilities*** – ступінь спільності (включаючи мінімізацію числа продуктів, в ідеалі один продукт), цілісності та взаємодії між різними компонентами інструментів інтеграції даних.
- ***Service Enablement Capabilities*** – інструменти інтеграції даних повинні володіти ***Сервіс-Орієнтованими Характеристиками*** і забезпечити підтримку розгортання SOA.

Gartner постійно проводить моніторинг програмних продуктів і періодично поновлює Магічний квадрант. На рис. 2.10 наведений Магічний квадрант Gartner засобів інтеграції даних (Data Integration Tools) на липень 2015 р.

Як видно з рис. 2.10, на той час лідерами розробки засобів інтеграції даних були компанії Informatica, IBM, SAP, Oracle, SAS.

Рис.2.10 – Магічний квадрант Gartner засобів інтеграції даних

2.10.5 Засоби інтеграції даних компанії Informatica

Корпорація Informatica (NASDAQ: INFA) – провідний світовий незалежний постачальник програмного забезпечення для промислової інтеграції та забезпечення якості корпоративних даних. Більш 3850 компаній по всьому світу використовують Informatica для забезпечення доступу, інтеграції, візуалізації даних, аудиту інформаційних активів для підвищення ефективності бізнесу, збільшення доходів клієнтів і виконання вимог регулюючих органів.

Відкрите і крос-платформенне програмне забезпечення Informatica дозволяє отримати доступ до даних практично всіх типів і робить їх значущими і корисними для людей і процесів, яким вони необхідні. Використовуючи Informatica, організації скорочують витрати, швидше досягають намічених цілей і реалізовують проекти в області інтеграції даних будь-якого масштабу і будь-якої складності. Єдина спеціалізація Informatica – продукти та послуги, пов'язані з управлінням даними.

Продукти Informatica підтримують широкий спектр корпоративних ініціатив в області інтеграції даних (ETL), серед яких побудова сховищ даних, міграція та консолідація даних, синхронізація даних, керованість даних, управління нормативно-довідковою інформацією (майстер-даними), доступ до неструктурованих даних, забезпечення якості даних. Використання продуктів Informatica забезпечує організаціям наступні переваги:

- Поліпшення прийняття управлінських та інших рішень на основі надання універсального доступу до цілісним, точним і послідовним корпоративних даних;
- Скорочення витрат на створення і підтримку IT-інфраструктури, а також її складності при підвищенні продуктивності IT-підрозділу і його здатності швидко реагувати на потреби бізнес-підрозділів;
- Підвищення з розрахунком на майбутнє гнучкості IT-організації в цілому і мінімізація довгострокових ризиків на основі використання сервісно-орієнтованої архітектури.

Всі програмні продукти платформи Informatica розроблені для допомоги компаніям будь-якого розміру в використанні своєчасних, релевантних і достовірних даних в якості своїх конкурентних переваг.

Data Integration

Доступ, обробка і доставка (Extract, Transform, Load – ETL) даних швидко, легко і економічно вигідно.

Data Quality

Повномасштабне рішення для комплексного управління якістю даних

Master Data Management

Підвищення операційної ефективності з єдиними і надійними даними

Application ILM

Економічно вигідне управління всіма фазами життя даних

Data Masking

Забезпечення безпеки конфіденційних даних

Data Replication

Витяг і швидке завантаження даних з неоднорідних джерел

B2B Data Exchange

Оптимізація процесів інтеграції неструктурованих даних

Complex Event Processing

Проактивний моніторинг даних і подій

Data Services

Пряме управління розрізненими даними з єдиної точки

Ultra Messaging

Забезпечення мінімальної затримки передачі даних по будь-яких мереж.

Application ILM

ILM-рішення від Informatica дозволяють IT-департаментам управляти всіма етапами життєвого циклу даних – від розробки і тестування до архівації та списання. Рішення знижує повну вартість володіння і підвищує повернення інвестицій, пов'язаних з такими бізнес-додатками, як ERP, CRM, HR, SCM і сховищами даних. Це рішення дозволяє IT-департаментам працювати з даними за допомогою найбільш ефективної і відповідної IT-інфраструктури.

Informatica Data Archive

Informatica Data Archive допомагає ефективно управляти зростаючими обсягами даних за допомогою безпечної архівації даних в додатках, надаючи універсальний доступ до архівованих даних, і передаючи їх бізнесу за першим запитом.

Informatica Test Data Management

Informatica Test Data Management є спеціалізованим рішенням для знеособлення (маскування) конфіденційних структурованих даних і створення тестових середовищ для наступних завдань розробки і тестування. Рішення дозволяє проводити маскування по різним типам даних і отримувати дані, максимально наближені до реальних. Informatica Test Data Management також автоматизує процес створення малих, цільових баз даних з великих, комплексних баз даних. Завдяки цьому, значно знижується обсяг часу, ресурсів і дискового простору, необхідних для підтримки невиробничих середовищ.

Data Warehouse Advisor

Рішення Informatica Data Warehouse Advisor дозволяє вести моніторинг реального використання даних різними бізнес-підрозділами та департаментами для забезпечення найкращого управління сховищами даних і процесами інтеграції, оптимізувати сховища даних і наявні процеси інтеграції.

2.11. Семантична інтеграція даних

2.11.1 Способи представлення даних для інтеграції

Проблему інтеграції даних можна розглядати трохи під іншим кутом. Можна виділити два підходи до представлення даних: синтаксичний і семантичний. Більшість сучасних рішень інтеграції заснована саме на синтаксичному представленні даних. При такому підході розробник грунтується на зовнішній схожості даних. Семантичне ж представлення засноване на змістовній схожості.

Якими б не були досконалими з технічної точки зору реляційні і функціональні підходи до консолідації даних, за своєю суттю вони примітивні. Усі вони мають в основі традиційний інженерний світогляд, що склався упродовж усіх років існування IT.

Їх загальна слабкість в тому, що дані розглядаються як набори бітів і байтів, а їх семантика жодним чином не враховується. Більше того, немає навіть виразного визначення, що таке дані, а є лише аморфні висловлювання ніби «дані — це представлення фактів і ідей у формалізованому виді, придатному для передачі і обробки в деякому інформаційному процесі» або «дані, – це те, що знаходиться на нижньому рівні в ієрархії дані – інформація – знання».

Перші серйозні спроби змінити існуюче положення були пов’язані із створенням в 80-і роки систем, що складаються з безлічі баз даних, за ними послідували СУБД з агентами, що виступають «посередниками при доступі до даних» (mediators agent). Ці роботи дозволили зрозуміти, що причина складності при інтеграції даних, що містяться в класичних базах, криється в жорсткій зв’язаності баз і єдиності використовуваної в них схеми зберігання. Ці роботи залишилися на рівні академічних або університетських досліджень і до того ж не запропонували вирішення проблеми інтеграції гетерогенних даних, проблеми, яка загострилася у зв’язку з широким поширенням неструктурованих або квазіструктурованих даних.

Загальна причина невдач полягає в тому, що проблеми інтеграції не є чисто технічними — зовсім не складно об’єднати різні реляційні бази, використовуючи інтерфейси Open Database Connectivity (ODBC) або, наприклад, Java Database Connectivity (JDBC), але складніше інтегрувати дані, що поступають з джерел, що мають різні моделі або, що гірше, що мають різну семантику, тобто що інтерпретують одні і ті ж дані по-різному. Для автоматизації роботи з даними семантика має бути явним чином виражена і

включена в ці дані, тобто дані повинні містити в собі описи власної семантики. Тут можна провести паралель з тим, як людина узагальнює дані у своєму повсякденному житті, грунтуючись на розумінні навколишнього світу, семантика якого вже в нім знаходиться.

Якщо таке вдається реалізувати, то можливий перехід на наступний рівень інтеграції, який можна назвати семантичним.

2.11.2 Семантична інтеграція

Перші спроби створення систем з семантичною інтеграцією проводилися ще на початку 90-х років, проте семантичні методи ще довго не виходили за рамки дослідницьких проектів.

Семантична інтеграція — це скоріш доповнення, а не заміна стандартних методів, але це доповнення заповнює критичний пропуск в забезпеченні необхідної практичності даних і їх взаємозв'язку.

Застосування семантичних технологій — необхідна умова для побудови успішної Сервіс-Орієнтованої Архітектури підприємства (SOA). Без переваги семантичної інтероперабельності даних і федеративних онтологий, розгортана SOA-архитектура підприємства страждатиме від багатьох підводних каменів, від яких страждають традиційні технології EAI -інтеграції: складна і дорога підтримка і неймовірно крихкі правила трансформації даних при обміні, що вимагають переробки при зміні специфіки даних. Застосування Web-сервісів для «обгортання» успадкованих систем не вирішує також проблем якості даних: оскільки, як правило, при введенні даних в такі системи не закладається необхідних перевірок, можуть спостерігатися помилки дублювання і невідповідності даних.

У більшості своїй семантичні моделі будуються на основі одного з напрямів в логіці першого порядку, які являють собою сімейство мов, що дозволяють формально і однозначно описувати поняття в якій-небудь предметній області. Кожен клас («концепт») може бути співвіднесений з іншим подібним до нього концептом шляхом додавання тегів метаданих, що вказують на властивості, загальні риси, відмінності і так далі. Розширення моделей тегами дозволяє створювати такі структури, яких раніше не могло бути. Завдяки тому що дані представляються у вигляді графів, відбувається спрощення їх змін. Так наприклад, злиття двох моделей зводиться до об'єднання їх графів. Інформаційна одиниця може бути представлена ідентифікатором Uniform Resource Identifier (URI), за допомогою якого можуть бути встановлені відношення між двома або великим числом інформаційних

одиниць. Семантичні моделі, вони ж онтології, можуть бути написані з використанням Resource Description Framework (RDF) моделі для представлення даних, розробленою W3C на мові Web Ontology Language (OWL). [1]

Описати представлення моделі можна представити таким чином. На нижньому, атомарному рівні представлення даних розташовується словник, на наступному рівні розташовується *таксономія*, що є аналогом схеми. Кожен елемент контенту має представляти свою приналежність до певної таксономії. Наступний рівень — *онтологія* (словник і таксономія), є структурою, що виражає одночасно ієрархією і набір відношень між елементами словника в цій ієрархії. Семантична множина об'єднує онтології, таксономії і глосарії, що входять до складу корпоративної системи.

У останні декілька років з'явилися ряд специфічних стандартів і засобів для підтримки семантичної інтеграції. Проте одне з важливих досягнень — це можливість створення семантичної абстрактної моделі. Це забезпечує ряд переваг, які раніше не були доступні :

- можливість абстрагувати управління і підтримку системи під нашим контролем в автоматизованому режимі;
- можливість посилатися на різні бізнес-процеси через той самий абстрактний рівень управління.

Підводячи підсумок можна виділити наступні переваги семантичного підходу до інтеграції:

- структура даних орієнтована на відношення між одиницями даних незалежно від схожості форми їх представлення;
- дані зв'язуються на основі визначення в загальних онтологіях;
- менша прив'язка системи до стандартів обміну, внаслідок чого збільшується масштабованість.

Вже сьогодні на ринку є ряд продуктів, в яких, хай і частково, реалізовані методи семантичної інтеграції.

2.11.3 Приклад продуктів сучасного ринку ПЗ для реалізації семантичної інтеграції

Раніше за інших компанія Progress Software випустила семантичний інтегратор ***DataXtend Semantic Integrator. Progress***®. Він надає єдину модель даних, яка робить можливою семантичну інтеграцію даних різних інформаційних структур. Тому якщо уявлення про деяку структуру даних якої-

небудь системи або розробника конфліктує з представленням в іншій структурі, DataXtend SI перевірятиме обмін даними між системами, прагнучи до бізнес-цілісності даних. Він вирішує задачі інтеграції корпоративних даних, забезпечуючи допустимість даних на основі бізнес-правил для користувачів і додатків, яким потрібні ці дані. Використання рішення семантичної інтеграції даних DataXtend дозволяє підприємству обмежитися для додатків тільки слабо пов'язаними інтерфейсами і забезпечує слабку зв'язаність на семантичному рівні. [2]

Семантичний сервер (Semantics Server) компанії Алтимета дозволяє поліпшити інтеграцію складних даних і оптимізувати доступ до ключової інформації в масштабі усієї організації. Він дозволяє в реальному часі перетворити розрізнені дані організації на повноцінну бізнес-інформацію, придатну для ухвалення ефективних рішень. Семантичний сервер може використовуватися спільно з Інтеграційним сервером цієї ж компанії, що дозволяє забезпечити тотальну інтеграцію даних і додатків в масштабі усієї організації. Він забезпечує інтеграцію додатків з використанням архітектури SOA і підходу інтеграційних процесів (BPEL, BPMN). Це традиційний підхід, що використовується усіма найбільшими постачальниками інтеграційних рішень.

Семантичний сервер застосовує семантичні стандарти до SOA-архітектури. Він дозволяє:

- консолідувати не лише довідники і класифікатори, а взагалі усю інформацію, яку необхідно спільно використовувати системам, що інтегруються
- об'єднати, інтегрувати інформацію і гарантувати якість даних
- забезпечити отримання інформації із слабоструктурованих джерел
- робити автоматичну класифікацію даних і логічний висновок.

2.11.4 Засоби семантичної інтеграції даних

Найбільш поширений підхід до семантичної інтеграції даних заснований на використанні семантичних посередників (Mediators). Засобами посередників підтримуються уніфіковані метаописання інтегрованих джерел даних. Як правило, семантичні посередники розробляються для конкретної вузької предметної області. Механізми посередників спираються на онтологічні специфікації джерел. Для посередника створюється інтегрована онтологія використовуваних джерел. У таких системах потрібна також інтегруюча модель даних з розвиненими можливостями моделювання семантики даних.

Останніми роками з’явилися ряд публікацій, присвячених вирішенню проблеми семантичної інтеграції даних з множини джерел, в яких для представлення глобальної схеми в системі інтеграції даних пропонується використати апарат дескриптивних логік, втілений в мові опису онтологий OWL. У цих роботах онтологія предметної області використовується як концептуальна схема. Перевага такого підходу полягає не лише в тому, що основою призначеного для користувача інтерфейсу є при цьому високорівнева семантична модель даних, але і можливість міркувань в термінах онтології, що служить концептуальною моделлю.

Резюме

1. Необхідність інтеграції даних виникає через неоднорідність програмного середовища, розподілений характер організації, підвищені вимоги до безпеки даних, необхідність наявності багаторівневих довідників метаданих, потребу в ефективному зберіганні й опрацюванні дуже великих обсягів інформації.
2. Інтеграція даних – це об’єднання даних, які спочатку вводяться в різні системи. Самі ці системи можуть розташовуватися б одній локальній мережі, але мати різні платформи і внутрішню архітектуру. Метою інтеграції даних є отримання єдиної і цілісної картини корпоративних даних предметної області. Інтеграція даних може бути описана за допомогою моделі, яка включає застосування, продукти, технології та методи.
3. Існує три основні методи інтеграції даних: консолідація, федералізація і розповсюдження
4. Консолідація даних – це збирання даних з територіально віддалених або різноплатформенних джерел даних в єдине сховище даних з метою їх подальшого опрацювання та аналізу.
5. У середовищі сховищ даних однією з найпоширеніших технологій підтримки консолідації є технологія ETL (витягання, перетворення і завантаження-extract, transform, and load). Ще одна поширена технологія консолідації даних ECM – керування змістом корпорації (*Enterprise Content Management*). Більшість рішень ECM напрямлені на консолідацію і керування неструктурованими даними, такими як документи, звіти і web-сторінки.

6. Федералізація даних забезпечує єдину віртуальну картину одного або декількох первинних файлів даних. Процес федералізації даних завжди полягає у витяганні даних з первинних систем на підставі зовнішніх вимог. Всі необхідні перетворення даних здійснюються при їх витяганні з первинних файлів.
7. Прикладом федералізації є інтеграція корпоративної інформації (*Enterprise Information Integration* ЕП).
8. Застосування розповсюдження даних здійснюють копіювання даних з одного місця в інше. Ці застосування зазвичай працюють в оперативному режимі і здійснюють переміщення даних до місць призначення, тобто залежать від певних подій. Оновлення в первинній системі можуть передаватися в кінцеву систему синхронно або асинхронно.
9. Прикладами технологій, що підтримують розповсюдження даних, є інтеграція корпоративних застосувань (*Enterprise Application Integration*, EAI) і тиражування корпоративних даних (*Enterprise Data Replication*, EDR).
10. Методи, що використовуються застосуваннями інтеграції даних, залежать як від потреб бізнесу, так і від технологічних вимог. Достатньо часто застосування інтеграції даних використовує так званий гібридний підхід, який включає декілька методів інтеграції. Приклад такого підходу – інтеграція даних про клієнтів (*customer data integration*, CDI), метою якої є забезпечення узгодженої картини інформації про клієнтів.

СПИСОК ЛІТЕРАТУРИ

1. Dan E. Linstedt. Data Vault Overview: The Next Evolution of Data Modeling [Електронний ресурс] / Dan E. Linstedt. – July 1, 2002. – Режим доступу: http://www.tdan.com/i021hy01.htm
2. Garretts Summary of Principles of Dataspace Systems / [Електронний ресурс] / Arizona State University – Режим доступу: http://aravaipa.eas.asu.edu/wiki/index.php/Garretts_Summary_of_Principles_of_Dataspace_Systems#Overview
3. Матеріали сайту «Открытые системы» – Режим доступу: http://www.osp.ru.
4. Li C. Query Algebra and Optimization for Relational Top-k Queries / [Електронний ресурс] / Li C, Chang K. C.-C, llyas I. R, and Song S.: RankSQL// Proceedings of the 2005 ACM SIGMOD Conference (SIGMOD 2005), Baltimore, Maryland, June. – 2005. – Режим доступу: http://eagle.cs.uiuc.edu/pubs/2005/ranksql-sigmod05-lcis-mar05.pdf
5. Hackathorn D. Reinventing Enterprise Systems Via Data Warehousing. – Washington, DC: The Data Warehousing Institute Annual Conference. – 1995. – 82 p.
6. Linstedt D. Data Vaultm overview the next evolution in data modeling / [Електронний ресурс] / – TDAN. – 2005. – Режим доступу: http://www.tdan.com/i021hy01.htm
7. Data Vault Modeling / [Електронний ресурс] / DanLinstedt. – 2007. – Режим доступу: http://www.danlinstedt.com
8. Data Vault overview: the next evolution in data modeling. / [Електронний ресурс] / TDAN. – 2007. – Режим доступу: http://www.tdan.com/edattl_archive.htm
9. Kossmann D. Personal Data Space / [Електронний ресурс]/ Dittrich J.-P. – Depeartment Informatik. – Режим доступу: http://www.inf.ethz.ch/news/focus/res_focus/feb_2006/index_DE
10. Codd E. F. Providing OLAP (On-Line Analytical Processing) to User-Analysts: An IT Mandate/ Codd E. F., Codd S. B., Salley C. T. – Codd E. F.& Associates. – 1993.
11. Madden S. TinyDB: An Acquisitional Query Processing System for Sensor Networks / [Електронний ресурс] / Madden S., Franklin M. J., Hellerstein J. M., Hong W.. ACM TODS, 30, No. 1. – March 2005. – Режим доступу: http://db.csailjTiit.edu/madden/html/tinydb_tods_nnal.pdf

12. Inmon W. H. Building The Data Warehouse (Second Edition). – NY, NY: John Wiley. – 1993. – 32 p.
13. Гаврилова ТА. Базы знаний интеллектуальных систем / Гаврилова ТА. Хорошевский В.Ф.: Учебник. – СПб., БХВ. – 2000. – 384 с.
14. Буч Г. Язык UML. Руководство пользователя / Буч Г., Рамбо Дж., Джекобсон А. [пер. с англ.]. – М. ДМК, 2000. – 496 с.
15. Дубова Н. Управление надежным хранением / [Електронний ресурс] / Открытые системы, #06/2002. – Режим доступу http://www.citforum.ru/nets/storage/management/
16. Саенко И.Б. Расширение реляционной модели для представления темпоральных данных в реляционных информационных системах / [Електронний ресурс] / Саенко И.Б. – IT сайт. – Военный университет связи, 2001. – Режим доступу:
http.V/www.inftech. webservis.ru/it/conference/sciii/2000/sessionl0/saenko.htm
17. Иванов А.Ю., Саенко И.Б. Основы построения и проектирования реляционных баз данных. – СПб: ВАС. – 1998. – 80 с.
18. Интеграция данных и хранилища / [Електронний ресурс] – TWAN. – 2007. – Режим доступу: http://citcitv.ru/12101/
19. Интеграция корпоративной информации: новое направление/ [Електронний ресурс] – TWAN. – 2005. – Режим доступу: http://citcity.ru/11155/
20. Итеграция данных в хранилищах/ [Електронний ресурс] – TWAN. – 2006. – Режим доступу: http://citcity.ru/11156/
21. Дрюэк К. (Katherine Drewek). Хранилища данных: сходство и различия подходов Билла Инмона и Ральфа Кимболла/ [Електронний ресурс] / Katherine Drewek: [пер. с англ.].. – 2005. – Режим доступу:
http://www.b-eye-network.com/view/743
22. Основные подходы к архитектуре Хранилищ данных/ [Електронний ресурс] /Interface.ru – 2006. — Режим доступу:
http://www.interface.*ru*/chapters/publicat.htm#5
23. Реализации хранилищ данных / [Електронний ресурс] / SPS Consulting. – 2007.– Режим доступу:
http://www.spc-consulting.ru/solution/database4.htm
24. Матеріали сайту компанії Interface Ltd. – Режим доступу: http://www.interface.ru.

25. Спецификация «Общая метамодель Хранилища данных» - Common Warehouse Metamodel (CWM) / [Електронний ресурс] / InterSoft Lab. – 2003. – Режим доступу: http://xmlhack.ru/texts/03/common/warehoyse.model/cwm.html
26. Инмон Б. Производительность систем хранилищ данных // Performance In The Data Warehouse Environment. – 2000. – №4. – С. 41-48.
27. Сходство и различия двух подходов к архитектуре хранилищ данных /[Електронний ресурс] / CitForum. – 2005. – Режим доступу: http://citforum.ru/consulting/Bl/data
28. Чен П. Модель "сущность-связь" – шаг к единому представлению о данных// СУБД. – 1995. – № 3. – С. 67-72.
29. Коротаев А. В., Малков А. С, Халтурина Д. А. Законы истории. Математическое моделирование развития. – Мир-Системы. Демография, экономика, культура. 2-е изд. М.: УРСС. – 1987. – 1082 с.

Навчальне електронне видання

КОЗЛОВСЬКА ВАЛЕНТИНА ПЕТРІВНА

МЕТОДИ ТА ЗАСОБИ ІНТЕГРАЦІЇ ДАНИХ

Конспект лекцій

Видавець і виготовлювач
Одеський державний екологічний університет
вул. Львівська, 15, м. Одеса, 65016
тел./факс: (0482) 32-67-35
E-mail: info@odeku.edu.ua
Свідоцтво суб'єкта видавничої справи
ДК № 5242 від 08.11.2016