Міністерство освіти та науки
Національний університет "Львівська політехніка"
кафедра "Інформаційні системи та мережі"

Моделювання процесів і технологій сховищ даних

Методичні вказівки

до виконання лабораторних робіт
з курсу "Технології сховищ даних"

для студентів, що навчаються за спеціальністю

8.080405 **"**Інтелектуальні системи прийняття рішень**"**

Уклала
Шаховська Н.Б.,
доц. каф. ІСМ, к.т.н.

Затверджені на засіданні
кафедри ІСМ
протокол № 1
від „ „ 200_ р.

Львів 2008

Лабораторна робота № 1. Проектування логічної структури сховища даних з архітектурою Корпоративна фабрика.

Мета роботи: Вивчення порядку, методів та засобів проектування і побудови сховища даних з корпоративною архітектурою та оцінка часу виконання запитів.

Теоретичні відомості

Парадигма для реляційних даних в сховищі даних (парадигма ***корпоративної інформаційної фабрики*** КІФ – Corporate Information Factory, CIF) розроблена Інмоном і передбачає, що дані повинні перебувати на низькому рівні ступені деталізації і в третій нормальній формі (3НФ, 3NF).

Після того, як дані такі формалізовані, важливо налаштувати їх для кожної групи користувачів та частково денормалізувати для пришвидшення виконання запиту.

Інмон вважає, що реляційна основа є достатньо гнучкою, щоб підтримувати багатовимірні вітрини даних і інші структури даних, як наприклад, сховища дослідження, бази даних, видобування даних.

Білл Інмон підтримує повторний або спіральний підхід до розвитку великого сховища даних. За цим підходом розвиток сховища відбувається ітераційно, тобто у разі виникнення потреби додається одна таблиця за один раз, що забезпечує лише незначну зміну схеми даних. Тому такий підхід до проектування сховища ще називають ***спіральним підходом.***.

У моделі Інмона, використовуючи повторний метод, помилки і внесення змін можуть стосуватись невеликої кількості даних, без необхідності повторно програмувати великі кількості даних у сховищі даних.

3НФ підхід дозволяє ґранулювати деталізовані та інтеґровані дані, забезпечивши максимальну гнучкість для опрацювання даних.

Білл Інмон також радить, щоб сховище даних містило корпоративно найвищий ґранульований рівень даних. Тоді структура і вміст сховища даних не підпорядковуватимуться вимогам будь-якого відділу, але натомість обслуговуватимуть вимоги корпорації.

На рис. 1 поданий підхід, що використовується у сховищах даних з архітектурою CIF.

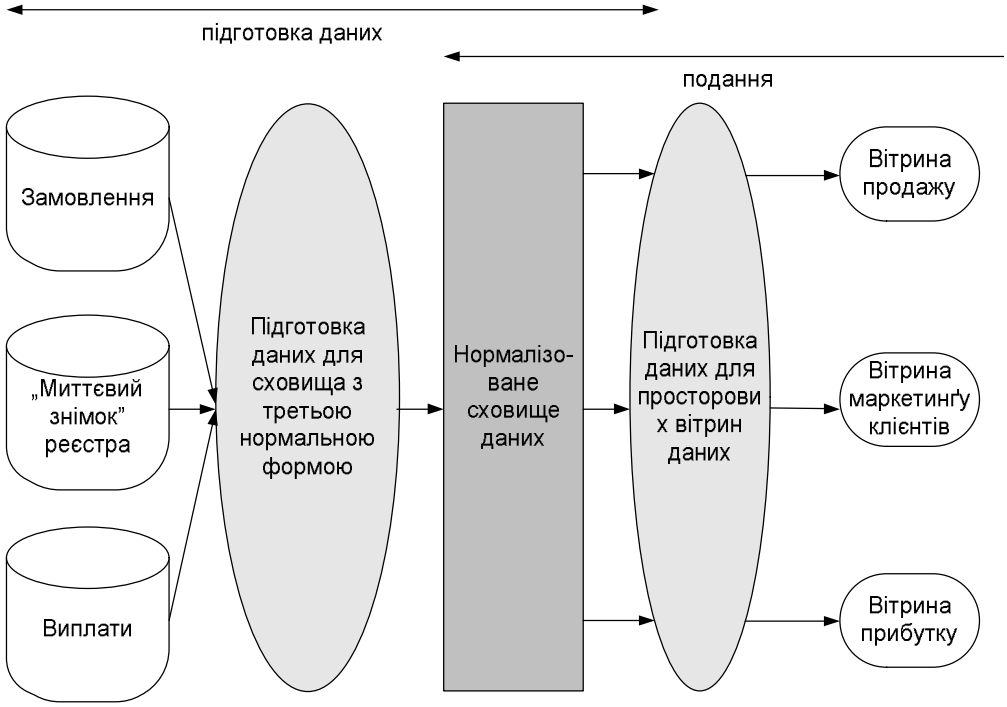


Рис 1. Нормалізоване сховище даних із просторовими вітринами підсумкових даних (CIF).

Колись цей підхід був відомий за назвою *корпоративного сховища даних* (КСД) (enterprise data warehouse, EDW) або *проектування сховища даних знизу вверх*. Робота такого сховища починається зі скоординованого отримання даних із джерел. Після цього завантажується реляційна база даних із третьою нормальною формою, що містить атомарні дані. Нормалізоване сховище використовується для того, щоб наповнити інформацією додаткові репозиторії презентаційних даних, тобто даних,

підготовлених для аналізу. Ці репозиторії, зокрема, включають спеціалізовані сховища даних для вивчення і видобування даних (Data Mining), а також вітрини даних.

При такому сценарії кінцеві вітрини даних створюються для обслуговування бізнес-відділів або для реалізації бізнес-функцій і використовують просторову модель для структурування сумарних даних. Атомарні дані залишаються доступними через нормалізоване сховище даних. Очевидно, що структура атомарних і сумарних даних при такому підході істотно різниться.

Наведемо приклад корпоративного сховища даних.

Приклад 1. Корпоративне сховище даних предметної області з продажу продукції подано на рис 2. Реалізація продажу подана таблицями *Продаж_головна* та *Продаж_детальна*. *Продаж_головна* містить інформацію про дату операції, підрозділ, з якого здійснено операцію, працівника та партнера (клієнта), а *Продаж_детальна* містить числові дані – кількість, ціну, суму.

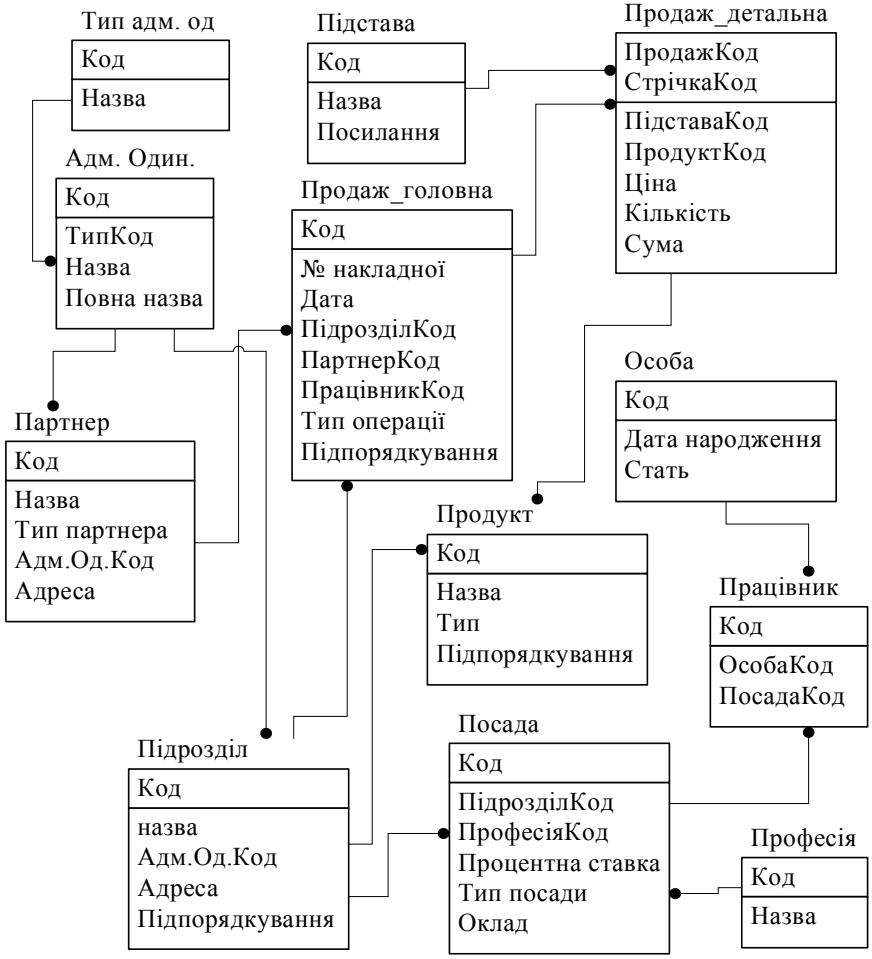


Рис. 2. Сховище даних з архітектурою корпоративної фабрики.

Як відмітні характеристики підходу Білла Інмона до архітектури сховищ даних можна назвати наступні.

1. Використання реляційної моделі організації атомарних даних і просторової - для організації сумарних даних.
2. Використання ітеративного або «спірального» підходу при створенні більших сховищ даних, тобто «будівництво» сховища даних не відразу, а по частинах (рис. 8.3). Це дозволяє, при необхідності, вносити зміни в невеликі блоки даних або програмних кодів і рятує від необхідності перепрограмовувати значні обсяги даних у сховищі. Те ж саме можна сказати й про потенційні помилки: вони також будуть локалізовані в межах порівняно невеликого масиву без ризику зіпсувати все сховище.

3. Використання третьої нормальної форми для організації атомарних даних, що забезпечує високий ступінь детальності інтегрованих даних і, відповідно, надає корпораціям широкі можливості для маніпулювання ними і зміни формату і способу подання даних у міру необхідності.
4. Сховище даних – це проект корпоративного масштабу, що охоплює всі відділи й обслуговує потреби всіх користувачів корпорації.
5. Сховище даних – це не механічна колекція вітрин даних, а фізично цілісний об'єкт.

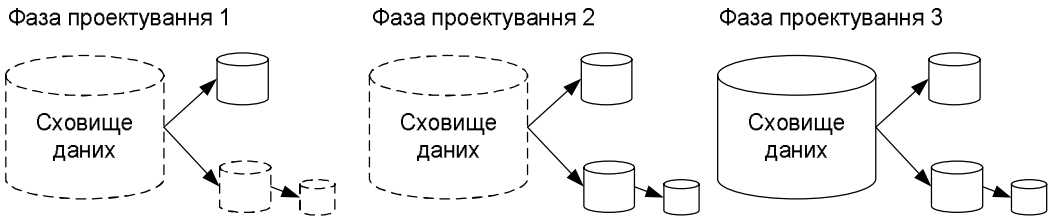


Рис. 3. Ітераційне розроблення сховища даних за методом знизу вверх.

Порядок виконання роботи

2. Створення схеми сховища даних з архітектурою КІФ.

Зміст завдання: розробити ERD та схему сховища даних (не менше 6-ти таблиць).

1. Розробка методів автоматичного заповнення таблиць.

Зміст завдання: оскільки перевірити якість схеми даних можна на великих обсягах даних, то розробити процедуру автоматичного заповнення таблиць, що будуть у подальшому використовуватись для прийняття рішень.

Приклад процедури, що заповнює таблиці Продаж_головна (dc_order) та Продаж_детальна (dt_order) (рис. 2):

```
Sub add_in_main()
Dim s As String
Dim rs As Recordset
Dim i As Integer
Dim count1 As Integer
Dim count2 As Integer
Dim count3 As Integer
For i = 1 To 1000
      'визначаємо кількість записів в таблиці ob_partner
      Set rs = CurrentDb.Execute("Select count(id) as c from ob_partner")
      count1 = rs!c
       'визначаємо кількість записів в таблиці ob_employee
      Set rs = CurrentDb.Execute("Select count(id) as c from ob_employee")
      count2 = rs!c
      'визначаємо кількість записів в таблиці ob_dept
      Set rs = CurrentDb.Execute("Select count(id) as c from ob_dept")
      count3 = rs!c
      ' вставляємо в таблицю dc_order випадкові значення (в межах кількості записів в таблиці)
      ' в поля partner_id,employee_id,
      ' dept_id, дату проставляємо сьогоднішню, а тип операції - прихід (1)
      s = "Insert into dc_order(evdate,partner_id,employee_id, dept_id, optype) VALUES " _
      & " (Now(), " & Int((count1 * Rnd) + 1) & ", " & Int((count2 * Rnd) + 1) & ", " _
      & Int((count3 * Rnd) + 1) & ",  1;"
      Set rs = CurrentDb.Execute(str)

      'визначаємо максимальне значення id з таблиці dc_order - те, що ми тільки-що створили
      Set rs = CurrentDb.Execute("Select max(id) as c from dc_order")
      count1 = rs!c
      'визначаємо кількість записів в таблиці ob_reason
       Set rs = CurrentDb.Execute("Select count(id) as c from ob_reason")
      count2 = rs!c
      'визначаємо кількість записів в таблиці ob_product
      Set rs = CurrentDb.Execute("Select count(id) as c from ob_product")
      count3 = rs!c
      ' таким самим чином вставляємо рядок в таблицю dt_order
      s = "Insert into dt_order(dcorder_id, reason_id, product_id, price, amount, suma) VALUES " _
      & " (" & Int((count1 * Rnd) + 1) & ", " & Int((count2 * Rnd) + 1) & ", " _
      & Int((count3 * Rnd) + 1) & ",  20, 30, 600;"
      Set rs = CurrentDb.Execute(str)
Next i
End Sub
```

2. Розробка методів визначення часу виконання запитів

Зміст завдання: на основі визначення часу виконання запиту, що використовує таблиці для підтримки прийняття рішень, проаналізувати якість сховища даних зі схемою КІФ.

Приклад модуля для визначення часу виконання запиту:

```
Public Sub test()
Dim MyTime1, MyTime2
Dim s As String
Dim rs As Recordset
' визначили час початку виконання запиту
MyTime1 = Time
str = "SELECT dc_order.*, dt_order.* " _
    & "FROM dc_order INNER JOIN dt_order ON dc_order.id = dt_order.dcorder_id;"
Set rs = CurrentDb.Execute(str)
rs.FindLast
s = rs!id
MyTime2 = Time
' визначаємо час виконання в секундах
Dim Msg
Msg = "Час виконання (с): " & DateDiff("s", MyTime2, MyTime1)
MsgBox Msg

End Sub
```

Зміст звіту по роботі:

1. Тема лабораторної роботи.
2. Завдання та постановка задачі лабораторної роботи.
3. Теоретична підготовка.
4. Опис виконаної роботи та отриманих результатів по кожному з пунктів завдання:

– перелік, опис та обґрунтування таблиць в сховищі даних;
– приклади формування і заповнення таблиць значеннями;
– опис зв’язків та їх властивостей між таблицями;
– визначений час виконання запиту;

5. Висновки.

Лабораторна робота № 2. Проектування логічної структури сховища даних з архітектурою шини.

Мета роботи: Вивчення порядку, методів та засобів проектування і побудови сховища даних з архітектурою шини та оцінка часу виконання запитів.

Теоретичні відомості

Рис. 1 подає альтернативний підхід до архітектури сховищ даних, відомий як ***Сховище даних з архітектурою шини,*** або ***підхід Ральфа Кімбола***, або ***Просторове Сховище*** [61].

У цій моделі первинні дані перетворяться в інформацію, придатну для використання, на етапі підготовки даних. При цьому обов'язково приймаються до уваги вимоги до швидкості опрацювання інформації і якості даних. Як і в моделі Білла Інмона, підготовка даних починається зі скоординованого добування даних із джерел. Ряд операцій відбувається централізовано, наприклад, підтримка і зберігання загальних довідкових даних, інші дії можуть бути розподіленими.

Область подання просторово структурована, при цьому вона може бути централізованою або розподіленою. Просторова модель сховища даних містить ту ж атомарну інформацію, що й нормалізована модель, але інформація структурована по-іншому, щоб полегшити її використання й виконання запитів. Ця модель включає як атомарні дані, так і узагальнювальну інформацію (агрегати у зв'язаних таблицях або багатомірних кубах) відповідно до вимог продуктивності або просторового розподілу даних. Запити в процесі виконання звертаються до усе нижчого рівня деталізації без додаткового перепрограмування з боку користувачів або розроблювачів застосування.

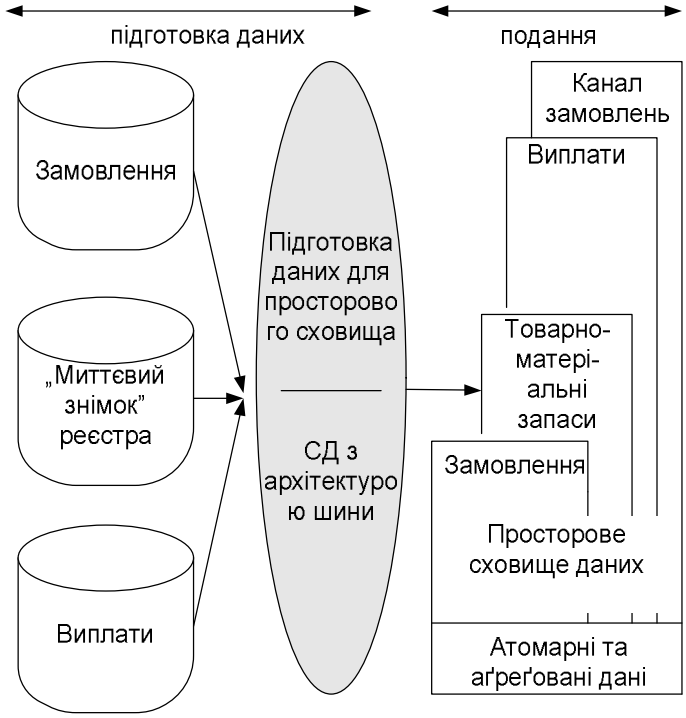


Рис 1. Просторове сховище даних.

На відміну від підходу Білла Інмона, просторові моделі будуються для обслуговування бізнес-процесів (які, у свою чергу, пов'язані з бізнес-показниками або бізнес-подіями), а не бізнес-відділів. Наприклад, дані про замовлення, які повинні бути доступні для загалькорпоративного використання, вносяться в просторове сховище даних тільки один раз, на відміну від КІФ-підходу, у якому їх довелося б тричі копіювати у вітрини даних відділів маркетингу, продажів і фінансів. Після того, як у сховищі появляється інформація про основні бізнес-процеси, консолідовані просторові моделі можуть видавати їхні перехресні характеристики. Матриця корпоративного сховища даних з архітектурою шини виявляє й підсилює зв'язок між показниками бізнес-процесів (фактами) і описовими атрибутами (вимірами).

Підсумовуючи все вищевикладене, можна відзначити типові риси підходу Ральфа Кімбола.

1. Використання просторової моделі організації даних з архітектурою «зірка» (star scheme) – детальніше розглянуто далі.
2. Використання дворівневої архітектури, що включає стадію підготовки даних, недоступну для кінцевих користувачів, і сховище даних з архітектурою шини як таке. До сховища входять кілька вітрин атомарних даних, кілька вітрин аґреґованих даних і персональна вітрина даних, але воно не містить одного фізично цілісного або централізованого сховища даних.
3. Сховище даних з архітектурою шини має наступні характеристики:
   - воно просторове;
   - воно містить як дані про транзакції, так і сумарні дані;
   - воно містить вітрини даних, що описують тільки одну предметну область або мають тільки одну таблицю фактів (fact table);
   - вітрини даних додаються у міру необхідності;
   - воно може містити безліч вітрин даних у межах однієї бази даних.
3. Сховище даних не є єдиним фізичним репозиторієм (на відміну від підходу Білла Інмона). Це «віртуальне» сховище.

Є різні варіанти фізичної реалізації архітектури шини. Простіший варіант моделі – «зірка» (star schema) подає собою радіальну схему, у центрі розміщена головна таблиця фактів, що аналізуються, та пов’язаних з нею таблиць вимірів, що вміщують довідкову інформацію. Така схема оптимізується

під найбільш поширені запити, тому реляційні таблиці вимірів можуть бути ненормалізованими. Якщо таблиці вимірів нормалізовані, то така модель називається «сніжинкою» (snowflake schema).

Модель даних складається з двох типів таблиць: однієї таблиці фактів (fact table) – центр «зірки», і декількох таблиць вимірів (dimension table) за кількістю вимірів в моделі даних – проміння «зірки» (рис. 2).

Рис. 2. Фрагмент схеми сховища даних, виконаного в моделі «зірка», для відображення господарських операцій.

На відміну від корпоративного сховища, в якому для подання господарських операція, а саме продажу, використано 2 таблиці (Продаж_головна та Продаж_детальна), в архітектурі шина використано одну таблицю, яка є таблицею фактів і по якій у подальшому приймаються рішення.

Інші таблиці (таблиці вимірів) є денормалізованими (наприклад, таблиця Менеджер містить інформацію не тільки про менеджера, але й про товарні групи, якими він займається, а окремої таблиці для товарних груп немає).

В архітектурі «сніжинка» (рис. 3) таблиці вимірів також нормалізовані. Так, у порівнянні зі схемою сховища даних, поданою на рис. 2, таблиці вимірів Менеджер та Товар є нормалізованими.

Є різні варіанти фізичної реалізації архітектури шини. Простіший варіант моделі – «зірка» (star schema) подає собою радіальну схему, у центрі розміщена головна таблиця фактів, що аналізуються, та пов'язаних з нею таблиць вимірів, що вміщують довідкову інформацію. Така схема оптимізується під найбільш поширені запити, тому реляційні таблиці вимірів можуть бути ненормалізованими. Якщо таблиці вимірів нормалізовані, то така модель називається «сніжинкою» (snowflake schema).

Модель даних складається з двох типів таблиць: однієї таблиці фактів (fact table) – центр «зірки», і декількох таблиць вимірів (dimension table) за кількістю вимірів в моделі даних – проміння «зірки» (рис. 2.)

Відзначимо, що у таблиці фактів немає ніяких відомостей про те, як групувати записи при обчисленні агрегатних даних. Наприклад, в ній є ідентифікатори продуктів або клієнтів, але відсутня інформація про те, до якої категорії відноситься заданий продукт або у якому населеному пункті

розташовується певний клієнт. Ці відомості, надалі використовувані для побудови ієрархій у вимірюваннях куба, містяться у таблицях вимірів (див. рис. 3).

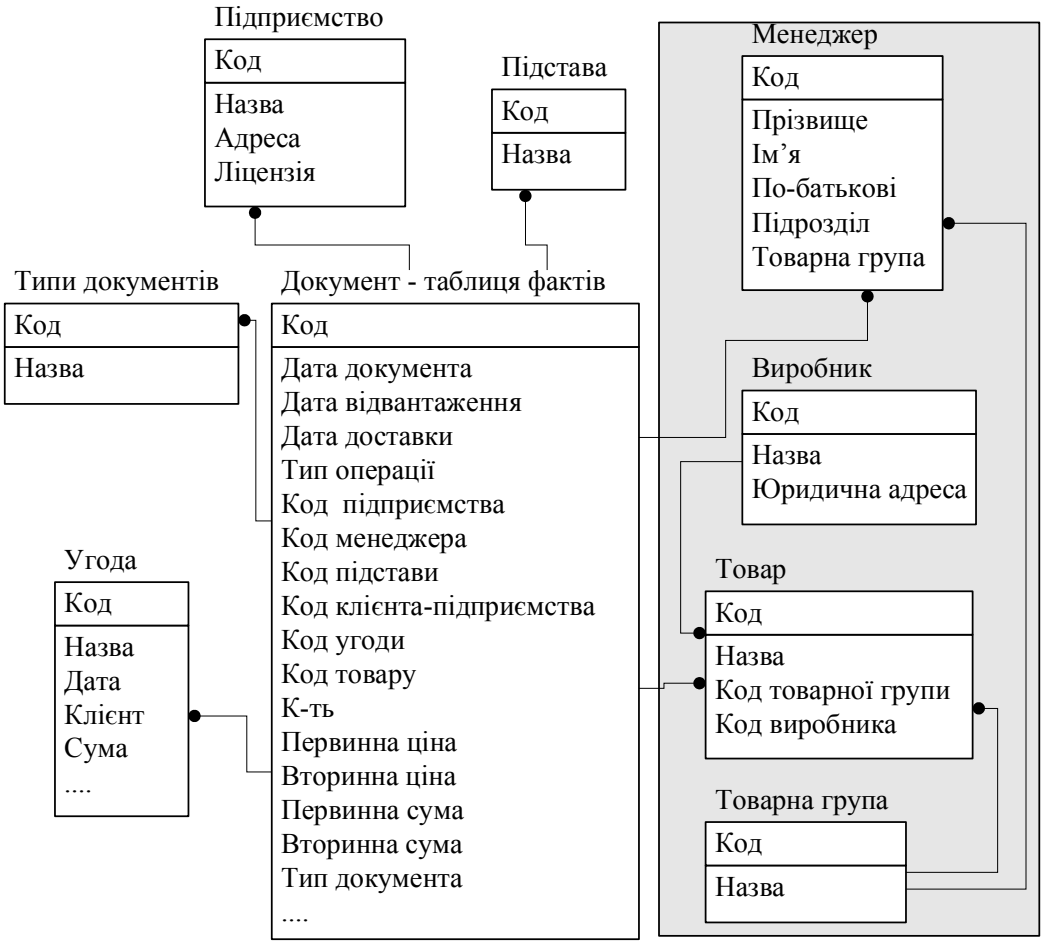


Рис. 3. Фрагмент схеми сховища даних, виконаного в моделі «сніжинка», для відображення господарських операцій.

Таблиці вимірів розшифровують ключі, на які посилається таблиця фактів; наприклад, таблиця виміру «Товар» бази даних автоматизації господарських операцій може містити відомості про назву товару, його виробника, код товарної групи. За рахунок використання спеціальної структури таблиці вимірів реалізується ієрархія вимірів, зокрема гілкування (так звана схема «Сніжинки»).

Зазначимо, що відношення вимірів залишаються сильно нормалізованими, а слабо нормалізоване відношення фактів використовується для збереження агрегованих даних.

SQL-запит до схеми «зірка» зазвичай містить в собі:

- одне або декілька з'єднань таблиці фактів з таблицями вимірів;
- декілька фільтрів (SQL-оператор WHERE), вжитих до таблиці фактів або таблиць вимірів;
- групування і агрегацію за необхідними елементам ієрархії вимірів (dimension elements).

Запит до схеми «сніжинка» міститиме ще з'єднання для таблиць-підвимірів.

Приклад 1. Наведемо SQL-запит для визначення сум продажів по товарних групах для сховища даних, схема даних якого подана на рис. 3.

```
SELECT [Товарна група].[Назва], [Підстава].[Назва], SUM([Документ].[Первинна сума])
FROM [Підстава] INNER JOIN ([Товарна група] INNER JOIN ([Товар] INNER JOIN [Документ]
ON [Товар].[Код] = [Документ].[код товару])
ON [Товарна гурпа].[код] = [Товар].[код товарної групи])
ON [Підстава].[код] = [Документ].[код підстави]
GROUP BY [Товарна група].[Назва], [Підстава].[Назва];
```

Багатовимірні моделі розглядають дані або як факти з відповідними чисельними параметрами, або як текстові виміри, які характеризують ці факти. У роздрібній торгівлі, наприклад, купівля — це факт, обсяг купівлі і вартість — параметри, а тип придбаного продукту, час і місце купівлі — виміру. Запити агрегують значення параметрів по всьому діапазону виміру, і у результаті одержують такі величини, як загальний місячний обсяг продажу певного продукту. Багатовимірні моделі даних мають три важливі області застосування, пов'язані з проблематикою аналізу даних.

- Сховища даних інтеґрують для аналізу інформації з декількох джерел на підприємстві.
- Системи підтримки прийняття рішень дозволяють оперативно одержати відповіді на запити, які охоплюють великі обсяги даних у пошуках загальних тенденцій.
- Застосування видобування даних служать для виявлення знань за рахунок напівавтоматичного пошуку невідомих шаблонів і зв'язків у базах даних.

Детальніше опишемо вимоги до таблиць фактів та вимірів.

Вимір — ключова концепція багатовимірних баз даних. Багатовимірне моделювання передбачає використання вимірів для надання максимально можливого контексту для фактів. На відміну від реляційних баз даних, контрольована надмірність в багатовимірних базах даних, загалом, вважається виправданою, якщо вона збільшує інформаційну цінність. Оскільки дані в багатовимірний куб часто збираються з інших джерел, наприклад, з транзакційної системи, проблеми надмірності, пов'язані з оновленнями, можуть розв'язуватися набагато простіше. Як правило, у фактах немає надмірності, вона є тільки у вимірах.

Виміри використовуються для вибору і агрегації даних на необхідному рівні деталізації. Виміри організовуються в ієрархію, що складається з декількох рівнів, кожний з яких описує рівень деталізації, що вимагається для відповідного аналізу.

Іноді корисно визначати декілька ієрархій для виміру. Наприклад, модель може визначати час як у фінансових роках, так і в календарних. Декілька ієрархій спільно використовують один або декілька загальних, найнижчих рівнів, наприклад, день і місяць, і модель групує їх в декілька вищих рівнів — фінансовий квартал і календарний квартал. Щоб уникнути дублювання визначень, метадані багатовимірної бази даних визначають ієрархію вимірів.

У деяких багатовимірних моделях рівень має декілька зв'язаних властивостей, які містять просту, неієрархічну інформацію. Наприклад, «Розмір пакету» може бути властивістю рівня у вимірі «Продукт». Вимір «Розмір пакету» може також одержувати цю інформацію. Використання механізму властивостей не приводить до збільшення числа вимірів в кубі.

Багатовимірні моделі, як правило, не передбачають функцій впорядковування або відстані для значень виміру. Єдине «впорядковування» полягає у тому, що значення вищого рівня містять значення нижчих рівнів. Проте для деяких вимірів, таких як час, впорядкованість значень розмірності може використовуватися для обчислення сукупної інформації, такий як загальний обсяг продажу за певний період. Більшість моделей вимагають визначення ієрархії вимірів для формування збалансованих дерев — ієрархії повинні мати однакову висоту за всіма гілками, а кожне значення не кореневого рівня — тільки одного батька.

Факти подають суб'єкт — якийсь шаблон або подію, які необхідно проаналізувати. У більшості багатовимірних моделей даних факти однозначно визначаються комбінацією значень вимірів; факт існує тільки тоді, коли комірка для конкретної комбінації значень не порожня. Проте деякі моделі трактують факти як «об'єкти першого класу» з особливими властивостями. Більшість багатовимірних моделей також вимагають, щоб кожному факту відповідало одне значення на нижчому рівні кожного виміру, але в деяких моделях це не є обов'язковою вимогою.

Кожен факт має деяку гранулярність, визначеною рівнями, з яких створюється їх комбінація значень вимірів. Наприклад, гранулярність факту в кубі, поданому на рис. 2 — це (Місяць х Продукт х Угода). (Рік х Продукт х Угода) і (День х Продукт х Угода) — відповідно грубша і тонша гранулярності.

Сховища даних, як правило, містять наступні типи фактів.

1. **Події (event),** принаймні, на рівні найбільшої гранулярності, як правило, моделюють події реального світу, при цьому кожен факт описує певний екземпляр явища, що вивчається. Прикладами може служити продаж, клацання мишею на Web-сторінці або рух товарів на складі.

2. **Миттєві знімки (snapshot)** моделюють стан об'єкту в певний момент часу, такі як рівні наявності товарів в магазині або на складі та кількість користувачів Web-сайту. Один і той же екземпляр явища реального миру, наприклад, конкретна банка бобів, може виникати в декількох фактах.
3. **Сукупні миттєві знімки (cumulative snapshot)** містять інформацію про діяльність організації за певний відрізок часу. Наприклад, сукупний обсяг продажу за попередній період, включаючи поточний місяць, можна легко порівняти з показниками за відповідні місяці минулого року.

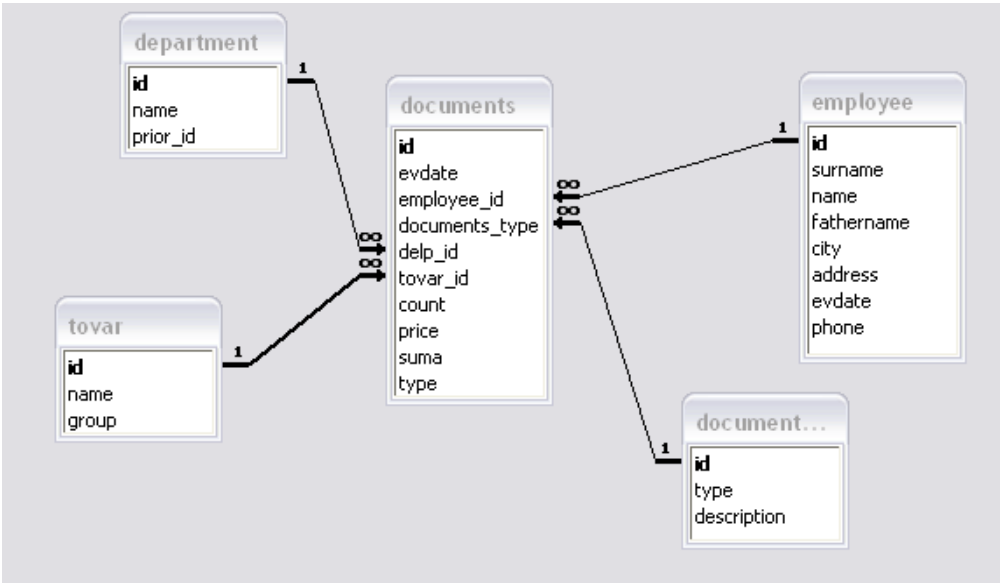
Сховище даних часто містить всі три типи фактів. Одні і ті ж початкові дані, наприклад, рух товарів на складі, можуть міститися в трьох різних типах кубів: потік товарів на складі, список товарів і потік за рік до поточної дати.

Порядок виконання роботи

0. **Створення схеми сховища даних з архітектурою зірка.**

Зміст завдання: розробити ERD та схему сховища даних зі схемою зірка. Автоматично заповнити таблиці та розробити засоби визначення часу виконання запиту, що включають відомості по одному з вимірів та по таблиці фактів.

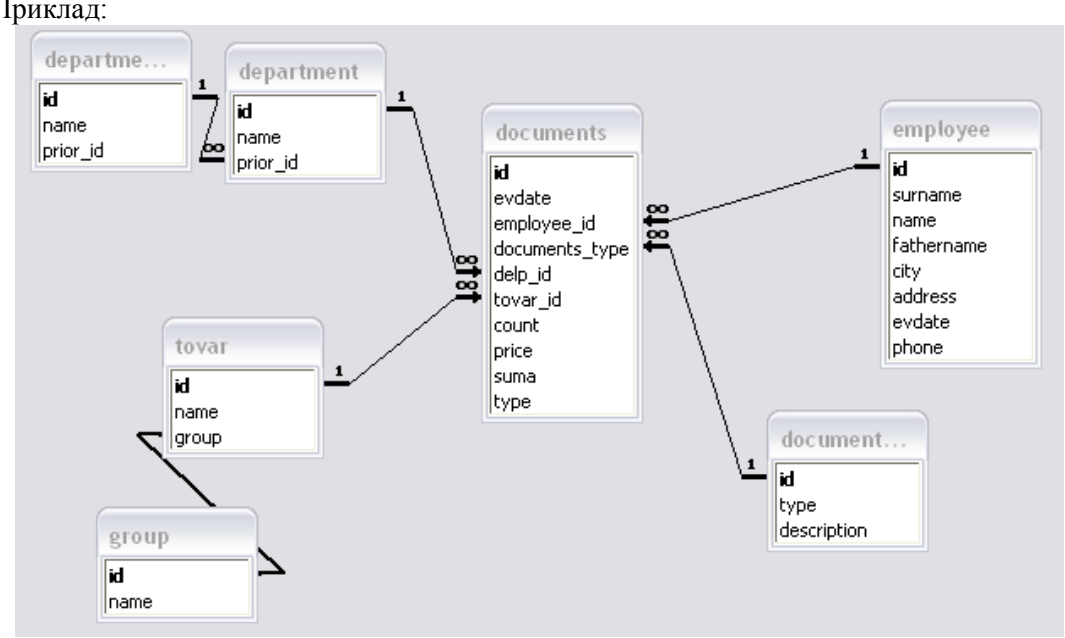
Приклад:



1. **Створення схеми сховища даних з архітектурою сніжинка.**

Зміст завдання: до розробленої вище схеми даних додати ще один під вимір та включити його до запиту. Визначити час виконання запиту.

Приклад:



2. **Порівняння схем сніжинки та зірки**

Зміст завдання: зробити висновки щодо швидкодії запитів на основі вище зазначених схем та вказати переваги та недоліки кожної з схем.

Зміст звіту по роботі:

1. Тема лабораторної роботи.
2. Завдання та постановка задачі лабораторної роботи.
3. Теоретична підготовка.
4. Опис виконаної роботи та отриманих результатів по кожному з пунктів завдання:

– перелік, опис та обґрунтування таблиць в сховищі даних;
– приклади формування і заповнення таблиць значеннями;
– опис зв'язків та їх властивостей між таблицею фактів та таблицями вимірів;
– визначений час виконання запиту;

5. Висновки.

Лабораторна робота № 3. Проектування логічної структури сховища даних з архітектурою зведення даних

Мета роботи: Вивчення порядку, методів та засобів проектування і побудови сховища даних з архітектурою зведення даних та оцінка часу виконання запитів.

Теоретичні відомості

Зведення даних (Data Vault) (ЗД) – предметно орієнтована, історична і унікально зв'язана множина нормалізованих таблиць, які підтримують одну або більше функціональних предметних областей. Це – гібридний підхід, що поєднує кращі особливості 3-ої нормальної форми (3НФ) і схеми «зірка».

У лабораторінй роботі № 1 подано оригінальну 3НФ модель, пристосовану до архітектури сховищ даних. Одна особливо складна проблема очевидна, коли значення часу-дати розміщена в первинному ключ таблиці батька (див рис. 1).

Якщо у таблицю батька додано новий запис, то ця зміна викликає примусове каскадування вниз через всі підлеглі табличні структури. Також, коли новий кортеж вставлений з наявним ключем батька (єдине поле зміни - мітка часу дати), всі кортежі-діти повинні бути переприсвоєні до нового ключа батька. Цей ефект каскадування стає все помітнішим при збільшенні кількості даних. Усе це призводить до неможливості підтримки роботи сховищ даних.

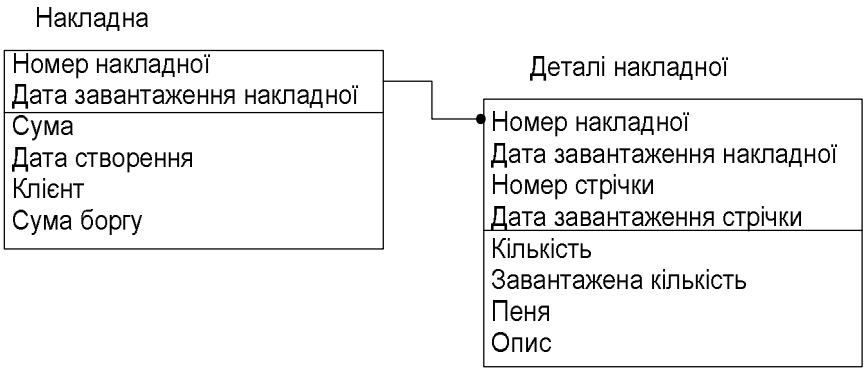


Рис 1. Використання часу у 3НФ.

Тому подальшим розвитком була схема «зірка», яка вимагала денормалізації, але водночас була краще пристосована до виконання аналітичних завдань. Модель «зірка» добре працює для швидкого подання багатовимірної інформації для певних груп кінцевих користувачів. Її перевагами є: багатовимірний аналіз, сумарні звіти (графи, пошук мінімуму, максимуму, зміна рівнів агрегування). Проте, як виявилося, така модель не є гнучкою, оскільки її структура є незмінною, жорстко форматованою. Також вона не здатна забезпечити статистичний аналіз.

Схема «зірка» здатна до виконання операції «slice and dice» для окремого користувача або групи користувачів. Також недоліком є неможливість подання усього масиву інформації у вигляді довільних звітів. Нарешті, модель дуже надмірна і важка для здійснення змін у структурі.

Зведення даних передбачає певну нормалізацію даних, що видозмінена для потреб сховищ даних. Ця архітектура використовує такі модельовані методи: зв’язок багато-до-багатьох, цілісність зв’язків, мінімально надмірні набори даних. Ці методи роблять модель зведення даних гнучкою, поширюваною і послідовною.

На рис. 2 подано модель зв’язків між сутностями предметної області «Торговельна фірма», яка дозволяє перейти до моделі СД зведення даних.

Рис. 2. Схема зведення даних.

Компоненти зведення даних

Є мінімальний ряд компонентів архітектури зведення даних:

- центральна таблиця (Hub) – моделює первинний ключ певної ґранули, тобто є центром зірки,
- таблиця зв’язку (Link entities) – сутність, що є зв’язком між екземплярами Hub,
- таблиці-супутники (Satellite entities) – забезпечують описовий контекст для відповідного екземпляра Hub або Link.

Заданий набір компонентів забезпечує максимальну гнучкість, обмежуючись традиційними підходами до моделювання.

Центральна таблиця являє собою таблицю з множиною унікальних ключів, що описують певну частину проблемної області. Вона складається з наступних атрибутів:

- Business key – первинний ключ об’єкту певної ґранули описуваної зірки,
- Surrogate key – сурогатний ключ, номер стрічки; використовується тоді, коли певний об’єкт описується декількома рядками,
- Load Date time stamp – дата появи запису в контенті сховища даних,
- Record source – запис про систему, звідки прийшов business key.

Таблиця зв’язку моделює зв'язок багато-до-багатьох на фізичному рівні між двома або більше центральними таблицями. Має наступні атрибути:

- $Hub_1 \ldots Hub_N$ key – зовнішні ключі таблиць центрів,
- Surrogate key – сурогатний ключ, номер стрічки; використовується тоді, коли описуються більше ніж два центри і неможливо однозначно визначити унікальний рядок у зв’язку з повторами значень зовнішніх ключів,
- Load Date time stamp – дата, коли зв'язок вперше був відображений у сховищі даних,
- Record source – запис про систему, звідки прийшов зв'язок.

Таблиці-супутники містять описовий контекст екземплярів Hub або Link. Опис може змінюватися з часом і така сутність повинна вміти зберігати нові або змінені дані. Має обов'язкові атрибути:

- первинний ключ Hub або первинний ключ Link – ключі відповідних екземплярів,
- Load Date time stamp – дата, коли з'явився описовий контент,
- Sequence surrogate number – номер стрічки для сутностей, які мають множинні значення або для організації в підгрупи,
- Record source – запис про початкову систему.

Наявність таблиць центів (Hub) та таблиць зв'язків (Link) дозволяють зменшити розрідженість даних, яка виникає внаслідок багатовимірного подання інформації.

Також одним із підвидів ЗД є **матрична методологія**, яка розширена ще однією таблицею *Малюнок* (Picture) для фіксації історичності даних та історичності зв'язків. Приклад матричної методології до проектування сховищ даних подано на рис. 8.3.

Проектування на основі зведення даних передбачає такі етапи.

- *Моделювання Hub.* Передбачає виявлення сутностей проблемної області.
- *Моделювання Links.* Виявлення можливих взаємозв'язків між сутністю, транзакціями та процесами функціонування, що є в проблемній області.
- *Моделювання Satellites.* Виявлення описового контенту як для сутностей – Hubs, так і для транзакцій – Links, що об'єднують сутності.

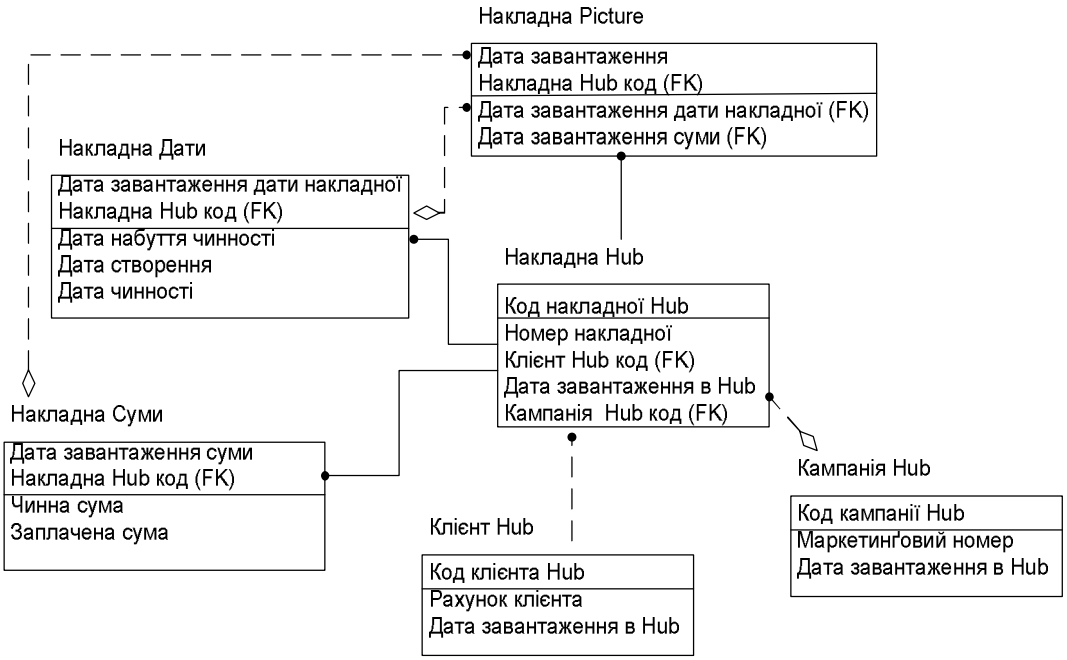


Рис. 8.3. Схема матричної ЗД.

Різниця між Satellite і Link полягає у тому, що Satellite має описовий характер, не містить зовнішніх ключів. Link описує функціональний зміст сутності, що дозволяє обмежувати період дії їх бізнес-процесів, переглядати вміст при зміні бізнесу.

Проектування за методологією ЗД вимагає дотримання таких правил.

- Первинні ключі таблиць-центрів не можуть міститися в інших екземплярах таблиць центів.
- Зв'язок між центрами відображається за допомогою Link.
- Link повинен містити не менше двох зовнішніх ключів таблиць-центрів.
- Link може використовуватися для зв'язку більше двох таблиць-центрів.
- Link може посилатися на інший екземпляр Link.
- Surrogate keys (номери стрічок) можуть бути присутніми в Hub і Link.
- Таблиці-супутники можуть описувати як Hub, так і Link.
- Таблиці-супутники повинні містити дату завантаження інформації або посилання на окрему таблицю дат.
- Таблиці-супутники фіксують тільки зміни, в них немає повторів рядків, дані групуються за вмістом і частотою змін.

Перед перетворенням сховища даних з іншою архітектурою в архітектуру з використанням ЗД необхідно дослідити наявні структури даних:

- незалежні таблиці – не перетворюються, копіюються повністю,
- чи визначені відношення первинних-зовнішніх ключів,
- чи використовуються в моделі номери стрічок – якщо використовуються, необхідно заново наповнити дані,
- чи можна класифікувати інформацію за класом або типом даних,
- наскільки часто змінюються атрибути.

Можливі застосування зведення даних

Зведення даних може застосовуватись для різних предметних областей. Маленький список можливостей поданий нижче.

- Динамічні сховища даних – побудовані на динамічних автоматизованих змінах, що виконуються як до процесу, так і до структури в межах сховища.
- Сховища дослідження – дозволяється споживачам керувати структурами сховища даних без втрати вмісту.
- Застосування «брудних» даних – дозвіл використовувати історичні та неочищені дані для видобування даних (штучного інтелекту).
- Швидке з'єднання зовнішньої інформації – здатність швидко зв'язати і пристосувати структури, щоб внести зовнішню інформацію і відобразити це в межах сховища даних без руйнування наявного вмісту.

Порядок виконання роботи

0. **Модифікація структури сховища даних КІФ до архітектури зведення даних.**
Зміст завдання: сховище даних з лабораторної роботи 1 модифікувати до архітектури зведення даних.
1. **Модифікація структури сховища даних зірка до архітектури зведення даних.**
Зміст завдання: сховище даних з лабораторної роботи 2 модифікувати до архітектури зведення даних.
2. **Висновки щодо ефективності архітектур.**
Зміст завдання: визначити час виконання запитів до сховища даних, розроблених у першому та другому завданнях. Визначити переваги та недоліки архітектури зведення даних на основі аналізу часу виконання запитів.

Зміст звіту по роботі:

1. Тема лабораторної роботи.
2. Завдання та постановка задачі лабораторної роботи.
3. Теоретична підготовка.
4. Опис виконаної роботи та отриманих результатів по кожному з пунктів завдання:
   - перелік, опис та обґрунтування таблиць в сховищі даних;
   - приклади формування і заповнення таблиць значеннями;
   - визначений час виконання запиту;
5. Висновки.