МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Магістерської підготовки

Кафедра Інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: Частина 2. Розробка корпоративного сховища даних

Виконав студент 2 курсу групи
МІС-19 спеціальності 122
Комп'ютерні науки

Постолова Ірина Василівна

Керівник к.ф.-м.н., доцент
Козловська Валентина Петрівна

Рецензент д.т.н, професов
Великодний Станислав Сергійович

Одеса 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Магістерської підготовки
Кафедра інформаційних технологій

Комплексна магістерська кваліфікаційна робота

на тему: Моделювання та розробка корпоративного сховища даних

Склад:

Частина 1 Моделювання та проектування корпоративного сховища даних

Виконавець: ПостоловЮрій Іванович
Керівник к.ф.-м.н., доцент
Козловська Валентина Петрівна

Частина 2 Розробка корпоративного сховища даних

Виконавець: Постолова Ірина Василівна
Керівник к.ф.-м.н., доцент
Козловська Валентина Петрівна

Староста роботи: ПостоловВіктор Іванович
Провідний керівник: к.ф.-м.н., доцент Козловська Валентина Петрівна
Рецензент: д.т.н., професор Ведикодний Станіслав Сергійович

ОДЕСА 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИУ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Магістерської підготовки
Кафедра інформаційних технологій
Рівень вищої освіти магістр
напрямок 122 Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри _____
_____
" 26 " жовтня 2020 р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Постоловій Ірині Василівні
(прізвище, ім'я, по батькові)

1. Тема роботи «Моделювання та розробка корпоративного сховища даних. Частина 2. Розробка корпоративного сховища даних»
керівник роботи Козловська Валентина Петрівна, к.ф.-м.н, доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від " 16 "жовтня 2020 р. №235-С
2. Строк подання студентом проекту 07.12.2020 р.
3. Вихідні дані до роботи вимоги до корпоративного СД основних користувачів
_____

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
У вступі викладаються мета та задачі кваліфікаційної роботи
У перших розділах проводиться опис та аналіз предметної області, наводиться огляд та вибір існуючих архітектур сховищ даних
Виконується проектування СД
Описується розроблене серверне ПЗ
У висновках підводяться підсумки виконаної роботи
_____
_____

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
_____
_____
_____
_____
_____
_____
_____
_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 26 жовтня 2020р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Термін виконання етапів роботи	Оцінка виконання етапу	
			у %	за 4-х бальною шкалою
1	Огляд та аналіз предметної області	26.10-28.10	90	відм
2	Огляд існуючих архітектур СД	28.10-30.10	90	відм
3	Денормалізація БД СД	31.10-02.11	85	добре
4	Визначення схеми СД	03.11-18.11	90	відм
	Рубіжна атестація	19.11	85	добре
5	Розробка серверного програмного	19.11-30.11	85	добре
	забезпечення			
6	Оформлення пояснювальної записки	30.11-06.12	85	добре
	Подання роботи на кафедру	07.12		
	Перевірка на плагіат	08.12		
	Рецензування	16.12		
	Інтегральна оцінка виконання етапів календарного плану (як середня по етапам)		85	добре

Студент _____ Постолова І.В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Козловська В.П.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

на магістерську кваліфікаційну роботу

Проектування та розробка корпоративного сховища даних

Частина 2. Розробка корпоративного сховища даних,

студентки Постолової Ірини Василівни

Кваліфікаційна магістерська робота: 70 с., 6 рис., 13 джерел.

Ключові слова: РЕЛЯЦІЙНА БАЗА ДАНИХ, СХОВИЩЕ ДАНИХ, ХМАРНІ СХОВИЩА ДАНИХ, SQL, СКБД MS SQL Server.

Мета дослідження – розробка корпоративного сховища даних для компанії з Інтернет-продажів.

Об'єкт дослідження – методики розробки корпоративних сховищ даних.

Предмет дослідження – розробка корпоративного сховища даних.

Задачі дослідження: проаналізувати наявні архітектури сховищ даних для вибору оптимальної архітектури у контексті предметної області, розробити корпоративне сховище даних для заданої предметної області.

Результати, їх новизна, теоретичне та практичне значення: проведений аналіз існуючих архітектур сховищ даних, проведений аналіз відповідності моделі розробленої моделі СД вимогам ефективної роботи СД, проведена часткова денормалізація таблиць БД сховища даних для зменшення кількості складних транзакцій, розроблені чотири функції БД та дві збережені процедури.

SUMMARY

for a master's degree

‘Design and Development of a Corporate Data Warehouse’,

Part 2. Development of a Corporate Data Warehouse,

student of Postolova Irina

Qualifying master's thesis: 70 pages., 6 figures., 13 sources.

Keywords: RELATIONAL DATABASE, DATA WAREHOUSE,

CLOUD STORAGE, SQL, MS SQL DBMS.

The purpose of the research is to develop a corporate data warehouse for an Internet sales company.

The object of the research is the methods of developing corporate data warehouses.

The subject of research is the development of a corporate data warehouse.

Research objectives: to analyze the existing data warehouse architectures to select the optimal architecture in the context of the subject area, to develop a corporate data warehouse for a given subject area.

Results, their novelty, theoretical and practical significance: an analysis of the existing architectures of data warehouses was carried out, an analysis was made of the compliance of the model of the developed DWH model with the requirements of efficient operation of the DWH, partial denormalization of the database tables of the data warehouse was carried out to reduce the number of complex transactions, four database functions and two stored procedures were developed.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ І ТЕРМІНІВ

CS – cloud storage, хмарне сховище даних

DSS – системи підтримки прийняття рішення (ecision Support System)

VPN – Virtual Private Network, віртуальна приватна мережа

БД база даних

БСД – багатовимірні сховища даних

ІС – інформаційна система

ПЗ – програмне забезпечення

РБД – реляційна база даних

РСД – реляційне СД, СД, що базується на реляційній моделі даних

СД – сховище даних

СКБД – система керування базами даних

Хайп – (англ. hype) агрессивная и навязчивая реклама, целью которой является формирование предпочтений потребителя

ЦОД – центр (зберігання та) обробки даних

ВСТУП

У наш час продаж товарів через мережу Інтернет стає звичним. Епідеміологічна ситуація останнього року ще більше підвищила привабливість покупок через Інтернет.

Компанії, що використовують Інтернет для продажу своїх товарів, звичайно мають бази даних цих товарів. Але для оцінки ефективності бізнесу та перспектив розвитку компанії недостатньо мати лише транзакційну базу даних, яка зберігає інформацію про товари, їх ціну, поточні дані про їх продаж. Потрібно мати сховище даних (СД), в якому містяться як поточні, так і історичні дані, міститься вся інформація, яка дозволяє оцінювати ефективність бізнесу та перспектив його розвитку.

Зібрана в базах даних підрозділів компанії 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.

Ці завдання вирішуються у системах підтримки прийняття рішень, основою яких є корпоративне сховище даних.

Сховище даних – це різновид системи управління даними, яка забезпечує підтримку бізнес-аналітики. Сховища даних призначені тільки для виконання запитів і аналізу і зазвичай містять великі обсяги історичних даних. Дані зазвичай надходять в сховище з найрізноманітніших джерел, таких як журнали додатків і додатки транзакцій.

Сховище даних служить для централізації і консолідації великих обсягів даних з різних джерел. Аналітичні інструменти дають можливість організаціям отримувати від власних даних цінні для бізнесу відомості і підвищувати ефективність прийнятих рішень. Згодом в сховище накопичуються записи за минулі періоди, які становлять велику цінність для фахівців з вивчення даних і бізнес-аналітиків. Ці можливості роблять сховища даних "єдиним джерелом перевірених даних компанії."

Метою даної комплексної магістерської роботи є моделювання, проектування та розробка сховища даних компанії по продажу товарів через мережу Інтернет.

Метою даної частини кваліфікаційної роботи є розробка корпоративного сховища даних.

1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ

Проектування сховища даних для компанії слід почати з визначення конкретних бізнес-потреб, погодження сфери застосування та розробки концепції проекту [1] [1)].

Моделювання та проектування сховища даних компанії з Інтернет-продажів було виконано у першій частині даної комплексної кваліфікаційної роботи.

Розроблена модель сховища даних є об'єднанням моделей баз даних декількох підрозділів компанії. В корпоративну модель сховища даних увійшли предметні області підрозділів, що займаються:

- роботою з постачальниками товарів,
- обліком клієнтів і їх вимог;
- маркетинговими дослідженнями з просування товарів.

Модель сховища даних містить 4 таблиці-виміри, таблицю існування, таблицю фактів. Також у моделі є таблиці, що створюють ієрархію виміру «Товар».

Основні групи користувачів сховища даних є аналітики та керівники різних ланок. Для цих груп користувачів не важливими є дані о поточних продажах товарів окремим покупцям. Для проведення фінансового аналізу, маркетингових досліджень, аналізу клієнтської бази, та інших досліджень, що можуть вплинути на політику компанії, цим групам користувачів потрібна узагальнена інформація по продажу товарів.

Для підвищення якості аналітичних досліджень потрібні зрізи даних по декільком вимірам з різним гранулюванням часу.

Кількість характеристик товарів може значно покращити аналіз привабливості різних товарів однієї групи, дозволить виявити ті характеристики, які

[1)] [1] Эрик Спирли. Корпоративные хранилища данных. Планирование, разработка и реализация. Т.1. М.: Вильямс, 2001. – 400 стр.

найчастіше значущі для покупців, та виявити закономірності формування у покупців критерію «ціна-якість» для різних товарів та груп товарів[2], [3] [1)].

Деякі атрибути товарів можуть бути значущі в різний мірі для різних груп товарів. Тобто, для деяких груп товарів зберігання цих характеристик товарів призведе до зайвих витрат – витрат часу персоналу для внесення зайвих даних в базу даних, та витрат простору на диску для зберігання зайвої інформації.

Розроблене сховище даних потрібне надавати всім основним групам користувачів можливість отримати відповіді на аналітичні запити у максимально короткий час у повному обсязі.

Для експлуатації сховища даних потрібно вибрати архітектуру СД.

Метою даної комплексної магістерської роботи є моделювання, проектування та розробка сховища даних компанії по продажу товарів через мережу Інтернет.

Завданням даної частини кваліфікаційної роботи є розробка корпоративного сховища даних.

[1)] [2] Корпоративные хранилища данных. Интеграция систем. Проектная документация. URL: https://www.prj-exp.ru/dwh/what_is_dwh.php

[3] Архипенко С., Голубев Д., Хранилища данных. От концепции до внедрения. М: Диалог-Мифи, 2002. – 528с.

2 АРХІТЕКТУРИ СХОВИЩ ДАНИХ

2.1 Хмарні рішення

В останні роки активно розвиваються хмарні технології. Їх розвитку не заважає навіть економічна криза. Маркетологи бачать за цими технологіями майбутнє і багато в чому вони мають рацію – завдяки публічним хмар приватні користувачі й невеликі компанії отримали доступ до рішень, які раніше були по кишені лише корпораціям з величезними IT-бюджетами.

Корпоративні системи теж еволюціонують з монолітних сховищ в розподілені додатки, але великий бізнес мігрувати в хмари поки не поспішає. Почасти з міркувань безпеки, але в основному через необхідність інтеграції успадкованих додатків для роботи в новому середовищі. Пристосовувати існуючі інформаційні системи доводиться в режимі реального часу і без права на помилку – це відштовхує багатьох IT-директорів від переходу на хмарні технології. Проте, процес йде і в недалекому майбутньому хмарні рішення для сховищ даних стануть звичайною архітектурою.

У 2016 році компанія Zoom виявила наступні тенденції у розвитку хмарних технологій [4] [1)]:

1. Інтеграція як сервіс (iPaaS). Інтеграція в масштабах підприємства стає величезною проблемою не тільки для бізнес-користувачів, але і для провайдерів SaaS і хмарних сервісів. Один із способів її рішення – використовувати інтеграційну платформу як сервіс (iPaaS). Рішення iPaaS не обмежуються перенесенням в хмару локальної системи інтеграції. Мова йде про створення хмарного середовища типу PaaS, що дозволяє застосовувати різні сценарії інтеграції бізнес-додатків, в тому числі локальних продуктів з SaaS. Подібних сервісів вже досить багато: Microsoft Azure, Dell Boomi, CloudHub

---

[1)] [4] Пять главных тенденций облачной интеграции данных в 2016 году. URL: https://zoom.cnews.ru/publication/item/55766

компанії MuleSoft, Tibco Cloud Bus, Red Hat OpenShift і т. Д. Якщо вірити звіту Gartner, в 2016 році не менше ніж в 35% великих і середніх компаній буде експлуатуватися хоча б одне iPaaS рішення.

2. Великі дані (BigData). Обсяг відомостей про клієнтів в корпоративних системах постійно зростає. Інформація надходить з різних джерел: структуровані бази даних, соцопитування, рекламні мережі, соціальні медіа і т.д. Привести такий гігантський обсяг даних до спільного знаменника, упорядкувати їх і обробити надзвичайно складно – для цього потрібні великі обчислювальні потужності. Компанії все частіше переносять пов'язані з BigData завдання в публічні або приватні хмари і цей процес буде йти по наростаючій.

3. Контейнерні хостинг-платформи (CaaS). Однією з головних тенденцій минулого року став активний розвиток платформ контейнерної віртуалізації, таких як Docker. Вони засновані на віртуалізації рівня операційної системи і відокремлюють додаток від інфраструктури, виконуючи його в ізольованому контейнері, який можна розгорнути практично миттєво. Контейнерна віртуалізація не вимагає запуску окремих екземплярів ядра ОС і її використання в публічних і приватних хмарах дає масу переваг крім швидкості розгортання додатків. Подібні рішення регулярно зазнають критики через потенційних проблем з безпекою, але плюси переважують мінуси і кількість що пропонують послуги CaaS провайдерів постійно зростає.

4. Хмарні ERP-системи. Інша область, в якій активно впроваджуються хмарні рішення – системи управління бізнес-процесами. Індивідуальне впровадження дуже цим систем потребує великих витрат, і багато компаній від нього відмовляються. Згідно зі звітом Gartner, з 2016 року почнеться суттєве збільшення організацій, які будуть використовувати хмарні і гібридні ERP-системи.

5. Хмарні брокери. У міру збільшення кількості використовуваних компанією хмарних сервісів управління ними істотно ускладнюється. Завдання їх інтеграції між собою та з внутрішніми ресурсами може стати логіс-

тичним кошмаром, що призвело до появи на ринку брокерів хмарних послуг, що грають роль своєрідних системних інтеграторів. Провайдери зазвичай пропонують замовникам рішення, що не враховують індивідуальних потреб конкретного бізнесу, а брокери забезпечують інтеграцію і управління. Їх клієнти не будуть прив'язані до одного або декількох хмарним провайдерам, а крім того, вони легко зможуть поміняти брокера або почати працювати безпосередньо з провайдером при необхідності.

Визначення хмарних обчислень на перший погляд дуже заплутане: це модель надання повсюдного і зручного мережевого доступу до загального пулу обчислювальних ресурсів, які потрібно конфігуруваи (наприклад, сервери, додатки, мережі, системи зберігання та сервіси), які можуть бути швидко надані і звільнені з мінімальними зусиллями з управління і необхідності взаємодії з провайдером [5] [1)].

Для того щоб краще уявити, що таке cloud computing, можна навести простий приклад: раніше користувач для доступу до електронної пошти вдавався до певного ПР (месенджери і програми), встановленому на його персональному комп’ютері, тепер же він просто заходить на сайт тієї компанії, чиї послуги електронної пошти йому подобаються, безпосередньо через браузер, без використання посередників.

Але цей приклад більше підходить для приватних хмар. Більш цікавішим є використання хмарних технологій в бізнесі. Сучасна реалізація них почалася з 2006 року. Тоді компанія Amazon представила свою інфраструктуру веб-сервісів, що не тільки забезпечує хостинг, але і надає клієнту віддалені обчислювальні потужності.

У наш час існує три моделі обслуговування хмарних обчислень:

[1)] [5] Сергей Глазунов. Бизнес в облаках. Чем полезны облачные технологии для предпринимателя. 22 февраля 2013. URL: https://kontur.ru/articles/225

1. Програмне забезпечення як послуга (SaaS, Software as a Servise). Споживачеві надаються програмні засоби – додатки провайдера, що виконуються на хмарної інфраструктурі.

2. Платформа як послуга (PaaS, Platform as a Service). Споживачеві надаються кошти для розгортання на хмарної інфраструктурі створюваних споживачем або придбаних додатків, що розробляються з використанням підтримуваних провайдером інструментів і мов програмування.

3. Інфраструктура як послуга (IaaS, Infrastructure as a Service). Споживачеві надаються кошти обробки даних, зберігання, мереж та інших базових обчислювальних ресурсів, на яких споживач може розгортати і виконувати довільний програмне забезпечення, включаючи операційні системи і додатки.

Сергій Глазунов зауважує, що виділяють кілька переваг, пов'язаних з використанням хмарних технологій.

- Доступність. Доступ до інформації, що зберігається на хмарі, може отримати кожен, хто має комп’ютер, планшет, будь-який мобільний пристрій, підключений до мережі Інтернет. З цього випливає наступна перевага.
- Мобільність. Користувач не має постійної прихильності до одного робочого місця. З будь-якої точки світу менеджери можуть отримувати звітність, а керівники – стежити за виробництвом.
- Економічність. Однією з важливих переваг називають зменшену витратність. Користувачеві не треба купувати дорогі, великі по обчислювальної потужності комп'ютери та комплектуючі, ПЗ, а також він звільняється від необхідності наймати фахівця з обслуговування локальних IT-технологій.
- Відсутність зайвих витрат. Користувач отримує необхідний пакет послуг тільки в той момент, коли він йому потрібен, і платить, власне, тільки за кількість придбаних функцій.

- Гнучкість. Всі необхідні ресурси надаються провайдером автоматично.
- Висока технологічність. Великі обчислювальні потужності, які надаються в розпорядження користувача, які можна використовувати для зберігання, аналізу і обробки даних.
- Надійність. Деякі експерти стверджують, що надійність, яку забезпечують сучасні хмарні обчислення, набагато вище, ніж надійність локальних ресурсів, аргументуючи це тим, що мало підприємств можуть собі дозволити придбати і містити повноцінний ЦОД.

Багато компаній називають витрати, масштабованість і цілісність даних трьома головними рушійними факторами. У більшості з них є довгострокові плани щодо хмарних технологій, при цьому 25% планують використовувати їх для більшості своїх потреб в ринкових даних протягом наступного року.

Проте найбільш вражаюче потенційне перевагу хмари полягає в його здатності надавати ефективні інструменти аналітики і машинного навчання, необхідні для розробки рішень ШІ (штучного інтелекту), а також розкривати потенціал для підвищення прибутковості [6] [1)].

Хмарні технології забезпечують швидкий і вигідний доступ на вимогу до великих обсягів даних. Користувачі хмарної середовища можуть швидко перевіряти результати експериментів, уникаючи тривалих процесів затвердження та впровадження, а також витрат, які раніше стримували темпи інновації.

2.2 Хмарні сховища даних

Хмарне сховище даних, або ж англійською cloud storage (CS) – це спеціальна модель сховища даних, при якій останні зберігаються не на конкретному носії або сервері, а на розподілених серверах в мережі. Як правило, всі

[1)] [6] Облачные вычисления URL: https://www.refinitiv.ru/ru/cloud-computing

ці сервери надаються в оренду якійсь третьою стороною. Власне CS можуть організувати для власних потреб хіба що дуже великі компанії.

Чим же відрізняється хмара від НЕ хмари? Уявімо, що ми орендували хостинг, доступ до якого здійснюється по FTP. Адже ми можемо закачати на FTP-сервер свої файли і отримувати до них доступ. Чим же звичайний FTP відрізняється від хмари?

Принципова відмінність в тому, що структура самого онлайн-сховища клієнтові не видно в принципі. Він не знає, конкретно на якому сервері в Інтернеті зберігаються його дані. Він підключається до певного вузла, скажімо, google.com і отримує доступ до своїх файлів. Хмарне сховище Google Drive складається з безлічі розподілених серверів, а для клієнта все це здається одним великим безрозмірним сервером [7] [1)].

Зберігати дані можна по-різному, тому і типи сховищ існують різні. В основному хмарні сховища розподіляють по типу збережених даних. Виділяють наступні типи хмарних сховищ:

- Файли і папки. Таке сховище називають файловим хмарним сховище. Типові приклади – Google Диск, Яндекс Диск і інші подібні сервіси. Часто, коли говорять про CS, мають на увазі цей тип сховища.
- Блочне сховище. Воно використовується корпоративними додатками, наприклад, базами даних, системами планування ресурсами. Типовий приклад – Amazon Elastic Block Storage (EBS).
- Об'єктне сховище. Цей тип сховища використовується для зберігання властивостей об'єктів. Типовий приклад такого сховища – Amazon Simple Storage Service (S3). Об'єктні сховище використовуються для забезпечення масштабованості додатків. Але при бажанні такі сервіси можна використовувати і для зберігання резервних ко-

[1)] [7] Хранилище в облаке. https://www.xelent.ru/blog/khranilishche-v-oblake/

пій, наприклад, Handy Backup підтримує S3-сховище для створення бекапа (backup – резервна копія).

Вибір типу сховища залежить від поставлених завдань, а саме від того, які дані планується зберігати в хмарі і як саме буде здійснюватися до них доступ.

Якщо обсяг даних невеликий і ставиться завдання забезпечення загального доступу до даних, тоді можна використовувати звичайні файлові хмарні сховища. Це найпростіший варіант. У таблиці 1 наводиться порівняльна характеристика файлових сховищ. Всі файлові сховища надають певний обсяг даних, які користувач може зберігати безкоштовно. Для персонального використання і невеликих проектів цього цілком достатньо. Більше простору є на платній основі. Але для корпоративного застосування часто місця буває замало і далеко не всі сервіси надають пакети саме для бізнесу. Найчастіше, при великих обсягах даних, таке сховище може бути більш вигідним, ніж традиційне файлове.

Таблица 1. Обзор популярных файловых хранилищ

Сервіс	Безкоштовний обсяг, Гб	Мінімальний тарифний план, вартість в міс.	Максимальний тарифний план, вартість в міс.
Google Drive	15	100 Гб/1.99 $	30 Тб / 299.99 $
OneDrive	5	50 Гб/1.99 $	1 Тб / 6.99 $
Dropbox	2	1 Тб/9.99 €	∞ / 10 € за користувача
Mega	50	200 Гб/4.99 €	4 Тб / 29.99 €
Яндекс.Диск	10	10 Гб/30 р.	1 Тб/200 р.

Найвигіднішим з сервісів є Mega, він надає 50 Гб «місця на диску» безкоштовно.

Якщо мова йде не про файлах, а про дані, які будуть зберігатися додатками, то однозначно для забезпечення масштабованості додатків потрібно вибирати S3-сховища.

2.2.1 Приклади використання хмарних сховищ даних

Існує багато прикладів використання хмарних сховищ, розглянемо п'ять найпопулярніших:

1. Створення резервних копій та відновлення. Хмарні ресурси забезпечують високу надійність і низьку вартість зберігання даних. Можливості масштабування практично необмежені.

2. Розробка програмного забезпечення та тестування. Розробка ПЗ і його тестування часто вимагають середовищ зберігання даних, що дублюються основне сховище. Створити такі умови вимагає додаткових витрат коштів та часу, а використання хмари набагато дешевше, а головне, що необхідні ресурси будуть виділені практично миттєво. Використання хмарних ресурсів – звичайна практика серед розробників ПЗ.

3. Міграція даних. Міграція даних в хмару – це дуже серйозне завдання, що вимагає величезного досвіду у системного адміністратора. Однак є сервіси, що полегшують цей процес. Один з таких корисних сервісів може бути AWS Import / Export Snowball.

4. Великі дані. BigData вимагають великих ресурсів – адже ці найбільші дані потрібно десь зберігати. А хмара надає практично необмежені ресурси. Саме тому при роботі з BigData оптимальним є хмарне сховище.

5. Спільний доступ до даних. Часто групі фахівців потрібен загальний доступ до даних, причому не завжди ця група знаходиться в офісі – фахівці можуть перебувати поза офісом (торгові агенти, фрілансери), в філіях або взагалі в інших країнах. Уявімо, що дані зберігаються на сервері всередині мережі підприємства. В цьому випадку потрібно організувати VPN на сервері – інакше як отримати доступ до даних ззовні офісу? Це складно, так навіщо надавати доступ до всіх ресурсів мережі, якщо потрібен доступ тільки до певних загальних файлів. У цьому випадку документи публікуються в хмарі і до них надається доступ всім, хто цього потребує. Функція «розшарювання» є у всіх популярних хмарних сховищах.

2.3 Переваги і недоліки хмарних сховищ даних

У відповідь на питання: «Чи варто зберігати дані у хмарі?» – у більшості випадків буде позитивна відповідь. Великі корпорації вже давно зробили свій вибір на користь хмари в тому чи іншому сенсі – або розгорнули свою хмарну інфраструктуру, або ж користуються сторонніми хмарними сховищами.

Невеликі компанії з побоюванням ставляться до зберігання в хмарі. Логіка наступна – 4 Тб «місця» обійдеться приблизно в 30 євро (що при використанні OneDrive, що при використанні Mega). Мережевий накопичувач WD My Cloud Home 4 Тб коштує приблизно 180 євро (ціна може коливатися в залежності від курсу). Разом покупка накопичувача окупиться вже через півроку експлуатації і у вас буде власне хмара.

Але не потрібно забувати, що дані в хмарі краще захищені насамперед від збою обладнання – ймовірність втрат прагне досягти 0, а ось звичайний фізичний накопичувач може вийти з ладу в будь-який момент, а гарантія поширюється тільки на фізичний накопичувач, а не на дані, що на ньому зберігаються – накопичувач нададуть новий, а дані не повернуть.

До того ж фізичний накопичувач може бути пошкоджений під час надзвичайної ситуації, і дані з нього будуть втрачені. Тому, вибираючи хмару, клієнт платить за свій комфорт, а й за безпеку своїх даних.

Переваги хмарних обчислень та сховищ [8] [1)]:

- Легке впровадження послуг.
- Вартість послуги набагато менше, ніж купувати апаратного і програмного забезпечення, і його налаштування і управління.
- Послуга просто масштабована – можна купити велику ємність, продуктивність обчислень і швидкість інтернет-з'єднання.
- Надійніше, ніж утримувати в роботі своє власне рішення.

[1)] [8] Куда делись мои данные? URL: https://sisu.ut.ee/minuandmed/ru

- Підтримує впровадження інноваційних рішень.

Виходячи з переваг хмарних обчислень, хмарні сервіси пропонуються кінцевому користувачеві безкоштовно або за невелику оплату.

Існує чимало небезпек при використанні хмарних послуг.

1. Витік даних:
   - Оскільки в репозиторіях зберігається багато цінної інформації, вони стають все більш привабливими цілями.
2. Використання незахищеною аутентифікації:
   - Використання слабких паролів.
   - Невикористання двоетапної аутентифікації.
   - Доступність паролів чужим людям.
3. Використання незахищеного комп'ютера або іншого пристрою:
   - Невикористання антивірусних програм.
   - Використання недостатніх заходів безпеки для запобігання доступу до даних.
4. Психологічне маніпулювання користувачем:
   - Використання облікових записів, використовуючи методи обману, так щоб користувач надавав доступ до своїх даних. Проти цього допоможе використання двоетапної аутентифікації і підвищення обізнаності користувачів про методи вивудження інформації.
5. Зміна відносин з партнером:
   - Люди і організації можуть змінюватися, можуть змінюватися їх цілі вони можуть перестати бути сумісними з попередніми угодами.
6. Видалення або збій даних:
   - І у випадку з хмарними сховищами треба робити резервні копії і перевірити їх відновлення.
7. Недбале ставлення користувача до даних:

- Помилка, що мої і чужі дані не є важливими для інших і їх не обов'язково захищати.

8. Обмежений контроль над вашими даними:

- Немає фізичного доступу до даних.
- Доступ залежить від наявності Інтернету.
- Нездатність постачальника послуг надавати узгоджений рівень обслуговування.

9. Постачальник послуг призупиняє в односторонньому порядку доступ до даних клієнта:

- В умовах користування деяких постачальників хмарних послуг зазначено, що постачальник може в будь-який час, заборонити доступ до ваших даних.

Потенційні збитки при використання хмарних сервісів

- Збиток, пов'язаний з втратою даних.
- Вартість відновлення даних.
- Збиток, пов'язаний з витоком даних.
- Збиток, пов'язаний з втратою доступу до даних.

Найбільші випадки витоку призначених даних користувачів за останні сім років відбулися, як раз, в компаніях, які використовували не хмарну технологію зберігання даних: злом серверів Sony Pictures, сховище даних Ashley Madison.

Компанія Panoply в 2019 році опублікувала звіт, згідно з яким популярність хмарних рішень для зберігання даних приблизно в 10 разів перевершує використання локальних. Третина від 800 опитаних інженерів, вчених і аналітиків заявили, що користуються локальними сховищами, а 23% повідомили, що взагалі не використовують сторонні рішення для зберігання інформації. Така інформація є покажчиком на те, що глобальний «перехід в хмару» ще не відбувся, але станеться в найближчі роки при певних умовах.

2.4 Озера даних

У останні роки з’явився і поширився термін «озеро даних». Концепція набула великого поширення, оскільки обсяги даних збільшилися і продовжують збільшуватися в геометричній прогресії, потокових даних стало більше, а неструктуровані дані продовжують затьмарювати свого структурованого аналога. Чим озеро даних відрізняється від традиційного сховища даних?

Сховище даних є інструментом, який став синонімом процесу вилучення, перетворення і завантаження даних (ETL). На високому рівні сховище даних містить величезні обсяги даних, структурованих строго регламентованими способами. Вони вимагають, щоб перед завантаженням даних була проведена сувора схема. (Це майже завжди схема «зірка» або «сніжинка».) Схема в сховище даних визначається «за записом». Процеси ETL належним чином виводять звіти про помилки, створюють файли реєстрації і відправляють помилкові записи в файли винятків і таблиці, в які можна заглянути з плином часу.

У зв'язку з таким чітким підходом сховища даних підтримують частковий або інкрементний ETL. Іншими словами, в залежності від серйозності проблеми, організація може завантажувати або перезавантажувати частини свого сховища даних, коли щось йде не так.

Організації періодично заповнюють сховища даних. Дані оновлюються за допомогою регулярних циклів. Наприклад, о 3 годині ранку кожен день, коли співробітникам навряд чи знадобиться доступ до даних і зв’язаних систем. А коли вони приходять на роботу, все свіжі дані вже завантажені.

Безумовно, дані, які зберігаються в традиційних сховищах даних, залишаються цінними сьогодні. Проте, компаніям і їх керівникам варто почати переосмислювати підхід до інтеграції даних. Наприклад, це стосується Інтернету речей та аналітики. Датчики на транспортних засобах, сільськогосподарське обладнання, носяться пристроях, термостатах і навіть рослинах безперервно поставляють величезний обсяг даних [9].

У нинішніх умовах спостерігається зростання популярності озера даних. Це не синонім сховищ даних або вітрин даних. Так, всі ці об'єкти зберігають дані, але озеро даних принципово відрізняється. Як пише Девід Лошинь (David Loshin): «Ідея озера даних полягає в тому, щоб зберігати необроблені дані в їх оригінальному форматі доти, поки вони не знадобляться».

При доступі до озер даних користувачі повинні знати [9] [1)]:

- Конкретні типи даних і джерела, в яких вони потребують.
- Скільки даних їм потрібно.
- Коли їм це потрібно.
- Методи аналітики, які будуть застосовуватися до цих даних.

Чи можливо це в сховище даних? Скоріше ні. І навіть якщо б було можливо, малоймовірно, що бізнес-користувачів влаштує період виконання, особливо в сучасних швидко мінливих умовах. Крім того, одна конкретна схема майже напевно не буде відповідати всім потребам бізнесу. Тобто, в кінцевому підсумку, дані можуть виявитися практично марними для цілей співробітника.

Схема озера даних визначається «по читанню». Так, для озера даних все ще потрібно схема, але вона не визначена. Дані використовуються за планом або схемою, коли користувачі отримують їх, а не коли завантажують. Озера даних зберігають дані в незміненому (природному) стані; вони не визначають вимоги до тих пір, поки користувачі не запросять дані.

При правильному використанні озера даних надають бізнес-користувачам і технічним користувачам можливість запитувати менші, більш актуальні і більш гнучкі набори даних. В результаті час запитів може скоротитися до роботи як у вітрині даних, сховище даних або реляційної базі даних.

[1)] [9] Озеро данных и хранилище данных – в чем разница? URL: https://www.sas.com/ru_ua/insights/articles/data-management/data-lake-and-data-warehouse-know-the-difference.html

На даний момент ми знаходимося на ранніх стадіях розвитку концепції озер даних, тому існує безліч точок зору на цю концепцію. Одна думка, що озеро даних – це не тільки важлива, але й обов'язкова умова для компаній, які управляють даними. Прихильники цієї точки зору розуміють обмеження сучасних сховищ даних – вони не були створені для обробки величезних потоків неструктурованих даних. Більш того, різниця між схемами «за записом» і «з читання» не просто питання семантики. Навпаки, остання схема піддається значно більш швидкому часу відгуку і, відповідно, аналітиці.

Інша думка, що озеро даних – це модне слово або хайп (hype) постачальників програмного забезпечення, які серйозно зацікавлені в цьому. Більш того, деякі вважають озеро даних новою назвою для старої концепції з обмеженою можливістю застосування для компаній.

Протягом десятиліть сховища даних обробляли навіть великі обсяги структурованих даних виключно добре: списки співробітників, продажу, транзакції тощо. Вони надають безліч додатків для бізнес-аналітики та корпоративної звітності. Однак недоцільно очікувати, що одні й ті ж сховища даних будуть ефективно обробляти принципово різні обсяги, швидкості і типи даних.

Користувачам все частіше потрібні відповіді швидше, ніж можуть забезпечити традиційні сховища даних і звичні звіти. При правильному використанні озера даних дозволяють користувачам аналізувати невеликі набори даних і швидко відповідати на важливі питання.

2.5 Вибір архітектури сховища даних, що розробляється

Як показав аналіз можливостей та небезпек, переваг та недоліків сховищ даних з різною концепцією зберігання даних, на цей час неможливо однозначно визначити, чи надає хмарне сховище даних дійсні переваги перед локальним сховищем даних.

Навіть прихильники використання хмарних сховищ даних дотримуються думки, що необхідно створювати резервні копії СД у різних місцях зберігання – наприклад, на різних сервісах хмарних сховищ. Але цей підхід є досить простим (хоча й затратним) для файлового сховища. У цьому випадку до сховища не потрібно приєднувати ніякі додатки для обробки та аналізу даних.

Корпоративне сховище даних існує для обслуговування аналітичних запитів від аналітиків та керівників компанії. Резервні копію самих даних не будуть корисними, якщо не буде засобів до аналітичної обробки даних резервної копії. Таким чином, для гарантії безперервної роботи хмарного сховища даних потрібно мати два паралельні сховища на різних хмарах з можливостями їх однакового обслуговування. Таке використання хмарних СД може викликати проблеми.

Також недоліком є фізична відсутність у компанії даних зі сховища. Особливо це відчувається на початкових етапах впровадження та експлуатації сховища даних, коли можливі зміни у логічній моделі даних, а також обов’язково будуть присутні зміни фізичної схеми, викликані необхідністю оптимізувати запити до СД – підвищити швидкість їх виконання.

Тому для першої версії сховища даних компанії вибирається локальна архітектура СД з розташуванням серверу СД у головному офісі компанії.

Озера даних можна використовувати як перехідне сховище неструктурованих даних.

3 РОЗРОБКА ФІЗИЧНОЇ СХЕМИ СХОВИЩА ДАНИХ

У першій частині даної комплексної магістерської роботи було проведене моделювання та проектування сховища даних. В результаті проведеного аналізу даних та можливих моделей даних була вибрана схема «сніжинка» реляційної моделі даних для сховища даних.

Схема «сніжинка» замість більш простої і продуктивної схеми «зірка» була вибрана за декількома критеріями:

- Наявність виміру «Акція» з асоціативною сутністю «Продукти акції»;
- Бажаність для аналізу попиту клієнтів наявності відомостей про категорії та постачальників товарів не у якості атрибутів таблиці-виміру «Товар», а у виді таблиць-вимірів вищого ступеня ієрархії для таблиці «Товар».

Для подальшої розробки фізичної схеми сховища даних потрібно визначити, які транзакції найчастіше будуть виконувати користувачі основних груп «Аналітик» та «Керівник».

3.1 Визначення транзакції основних груп користувачів сховища даних

Як відомо, основними користувачами СД є аналітики та керівники компанії різного рангу.

Аналітики зазвичай використовують слабо агреговані дані, також вони можуть використовувати деталізовані дані.

Керівники нижньої ланки для складання поточних звітів про роботу компанії використовують слабо та середньо агреговані дані.

Керівники середньої ланки для складання звітів, що спрямовані на визначення політики компанії по підвищенню ефективності її роботи, використовують середньо та сильно агреговані дані.

Керівники вищої ланки для розробки стратегії розвитку компанії використовують сильно агреговані дані.

Потрібно визначити які зрізи даних потрібні користувачам різних груп.

Розглянемо варіанти використання СД, які можуть бути потрібними для якоїсь групи користувачів.

Для користувачів груп «Керівник нижньої ланки» та «Керівник середньої ланки» для аналізу клієнтської бази та прибутковості потрібні транзакції, які відповідають на питання:

- Які клієнти забезпечують найбільший прибуток?
- Які продукти дають найбільший прибуток?
- Який набір продуктів купують найприбутковіші клієнти?
- Товари яких фірм найкраще купують клієнти?
- Які властивості характерні клієнтам, що забезпечують найбільший прибуток?
- Які основні причини відмови від продукту (послуги)?
- Які клієнти найближчим часом, можливо, перестануть користуватися послугами компанії?
- Чи мають певні клієнти або продукти неприпустимо довгий термін постачання?
- Наскільки клієнти задоволені якістю товарів (послуг)?
- Як задоволеність клієнтів змінюється з часом?

Для користувачів груп «Керівник верхньої ланки» розробки стратегії розвитку компанії потрібні транзакції, які відповідають на питання:

- Які ключові показники продуктивності компанії в поточному періоді?
- Які тенденції зміни ключових показників продуктивності компанії з часом?
- Яка комбінація товарів і послуг просуває бізнес?

- Які сегменти ринку дають найбільший прибуток?
- Які з партнерів забезпечують найбільший прибуток?
- Яка вартість проведення маркетингової кампанії через заданий канал?
- Який канал проведення маркетингових кампаній є найефективнішим?

В можливих транзакцій користувачів можна виділити варіанти використання СД:

1) Вибрати продукти, що дають найбільший прибуток.
2) Визначити комбінацію товарів, що просуває бізнес.
3) Визначити сегменти ринку, що дають найбільший прибуток.
4) Визначити клієнтів, що забезпечують найбільший прибуток.
5) Визначити набір продуктів, що купують найприбутковіші клієнти.
6) Визначити групи клієнтів, що дають найбільший прибуток.
7) Визначити властивості, які характерні клієнтам, що забезпечують найбільший прибуток.
8) Визначити вартість проведення маркетингової кампанії через заданий канал.
9) Визначити найефективніший канал проведення маркетингових кампаній.
10) Визначити, наскільки клієнти задоволені якістю товарів.
11) Визначити, як задоволеність клієнтів змінюється з часом.
12) Визначити, які продукти постачають вчасно, які - із запізненням.
13) Визначити, чи мають певні клієнти або продукти неприпустимо довгий термін постачання?
14) Визначити основні причини відмови від продукту.

На рис. 1 наведена діаграма використання корпоративного сховища даних користувачами, які є керівниками різних рівнів.

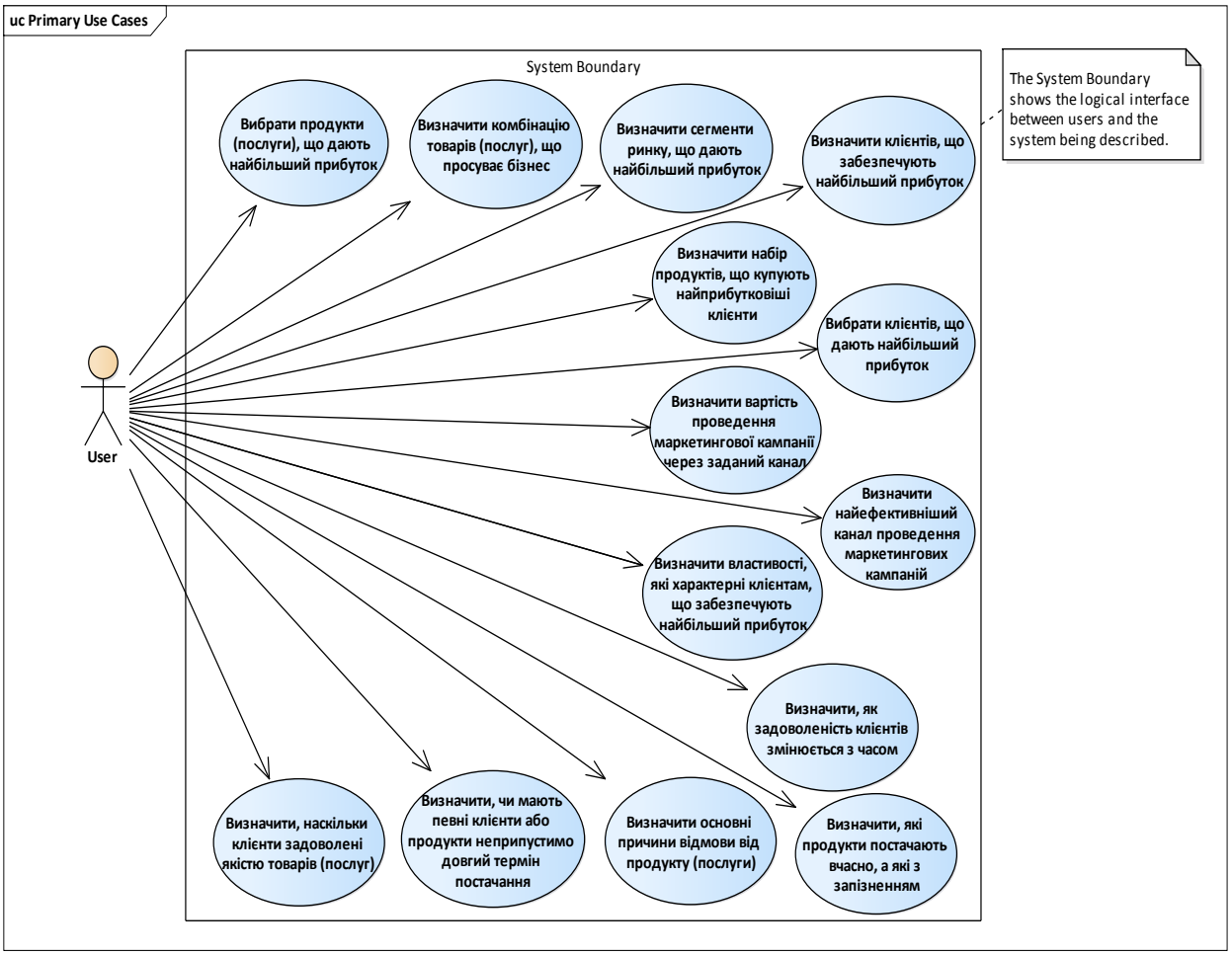

Рисунок 1 – Діаграма варіантів використання корпоративного СД

Як видно з варіантів використання сховища даних менеджерами та аналітиками компанії, для відповіді на їх питання потрібні не деталізовані, а агреговані дані. Агреговані дані у базах даних не зберігаються, їх отримуються на вимогу користувача за допомогою запитів з групуванням даних, тобто вони є віртуальними даними.

На відміну від транзакційних баз даних користувачам сховищ даних потрібно саме агреговані дані. Тому у сховищах даних поряд з деталізованими даними зберігаються агреговані дані різного ступеня агрегації. Таблиці сховища даних знаходяться у денормалізованому виді.

При проектуванні сховища даних потрібно виявити, які саме агрегати потрібні для основних запитів користувачів, І створити таблиці агрегованих

даних. Отримувати агрегати під час виконання запиту користувача неприпустимо, оскільки у цьому випадку запити будуть виконуватись не достатньо швидко.

Для подальшої розробки фізичної схеми сховища даних потрібно визначити, які транзакції найчастіше будуть виконувати користувачі основних груп «Аналітик» та «Керівник».

3.2 Різновиди транзакцій

У теорії БД, взагалі кажучи, під транзакцією розуміють одну з команд SQL –SELECT, INSERT, UPDATE, DELETE. Однак в залежності від типу додатків термін «транзакція» трактується більш вільно, як елементарна логічно завершена одиниця роботи (так звана бізнес-транзакція), яка може включати кілька команд вставки, видалення або модифікації. Залежно від того, які команди SQL використовуються, транзакції поділяють на транзакції тільки для запису (write-only), тільки для модифікації (modify-only), тільки для читання (read-only), тільки для видалення (delete-only). Транзакції тільки для читання називають запитом.

Виходячи з такого розподілу транзакцій за типами обробки з урахуванням частоти транзакцій кожного типу, можна виділити три основних класичних типу додатків БД.

– OLTP-системи (On-Line Transaction Processing). OLTP-система –це такий додаток, яке містить в основному транзакції вставки, оновлення та видалення, з високою частотою переважно транзакцій поновлення. Класичним прикладом таких систем є системи резервування авіаквитків або обслуговування готелів. Для них характерний високий рівень паралелізму (high concurrency), який в даному випадку означає, що багато користувачів використовують базу даних однаковим чином.

- DSS-системи (Decision Support System). DSS-система –це такий додаток, яке працює з дуже великою базою даних в режимі «тільки читання». Зазвичай використовуються набір фіксованих простих запитів або нерегламентовані запити користувачів. Хорошим прикладом такої системи є корпоративна інформаційна система організації.
- BATCH-системи. BATCH-системи – це такий додаток, яке працює з базою даних в неінтерактивному режимі. Зазвичай воно використовує багато транзакцій вставки, видалення і оновлення і має низький рівень паралелізму, що означає невелике число користувачів, що використовують базу даних однаковим чином. Істотним фактором є ставлення запитів до транзакцій поновлення. Класичним прикладом таких систем є обслуговування БД продукції організації.

Пізніше з’явились ще два види додатків з відмінним набором транзакцій:

- OLAP-системи (On-Line Analytical Processing). OLAP-система - це програма, яка забезпечує аналітичну обробку даних, що включає математичний, статистичний або інший аналіз даних. Такі системи не можна віднести повністю або до OLTP-, або DSS-систем. Вони розташовуються десь між ними. В рамках OLAP-систем виділяють так звані ROLAP-системи (Relational OLAP), тобто OLAP-системи, що використовують реляційні бази даних. Типові OLAP-системи розробляються зазвичай під багатовимірні моделі даних.
- VCDB-системи (Variable Cardinality Database). VCDB-система - це такий додаток обробки даних, для якого база даних зростає або стискається в розмірах періодично, в залежності від характеру обробки даних. Зазвичай розмір цих баз даних стає дедалі більше. Типовим прикладом такої системи є БД по забезпеченню безпеки (security authorization database), для якої характерна коротка за часом активність записів в таблиці.

Для аналізу транзакцій необхідно оцінити тип додатків, для яких розробляється база даних. Це дозволить оцінити:

- тип транзакцій (які);
- частоту транзакцій кожного типу (скільки);
- кількість одночасно працюючих з БД користувачів.

Специфікація транзакцій.

Оскільки будь-яка операція зміни даних в БД несе в собі потенційну можливість порушення цілісності даних, необхідно строго визначати транзакції і ідентифікувати інформацію, яка зазвичай включається в визначення транзакції [10] [1)].

Визначення транзакції може мати різні форми. Іноді для визначення транзакцій використовується репозиторій даних CASE-засобів проектування бази даних. Дуже часто визначення транзакцій виконується за допомогою текстових описів. Незалежно від обраного підходу, будь-яке гарне визначення транзакції включає кілька важливих елементів. До таких елементів відносяться:

- ім'я транзакції;
- номер транзакції ;
- опис транзакції;
- характер транзакції і її складність;
- обсяг транзакції;
- вимоги до продуктивності транзакції;
- відносний пріоритет;
- час виконання транзакції.

[1)] [10] Создание физической модели базы данных: проектирование производительности. Лекции НОУ ИНТУИТ URL: https://intuit.ru/studies/courses/599/455/lecture/10182

Для кожної транзакції може бути визначений характер транзакції (онлайнова транзакція або пакетна транзакція), а також вказана її складність. Зазвичай складність вказується в термінах «висока», «середня», «низька». Ця інформація потрібна для оцінки транзакцій бази даних в цілому. Кількість транзакцій тієї чи іншої складності впливає на час проектування фізичної моделі бази даних: чим більше в базі даних транзакцій високої складності, тим більше час проектування фізичної моделі. Складність транзакції є умовною мірою трудомісткості при досягненні вимог продуктивності бази даних.

Висока складність приписується зазвичай транзакції, яка має дві з наступних характеристик:

- містить від 8-ми до 10-ти команд SQL;
- містить пропозицію WHERE з великою кількістю предикатів;
- містить пропозицію WHERE з більш ніж трьома з'єднаннями або підзапитами;
- транзакція обробляє більш ніж 100 рядків.

Низька складність приписується транзакції з наступними характеристиками:

- містить до 3-х команд SQL;
- містить пропозицію WHERE з одним або двома предикатами;
- транзакція обробляє менш ніж 25 рядків.

Транзакція з середньою складністю має характеристики між низькою і високою складністю.

Інформація про частотах транзакцій включає зазвичай два параметра – середню частоту транзакції (наприклад, 50 тр/год) і пікову частоту транзакції (наприклад, 70 тр/год). Оцінка частотних характеристик БД дуже важлива для проектування фізичної моделі даних СД: настройка фізичної структури БД для транзакцій з високою частотою істотно відрізняється від настройки її для транзакції з низькою частотою використання.

Якщо для модернізації існуючих систем інформація про частоти транзакцій може бути представлена у вхідній документації на основі аналізу статистики експлуатації системи, то для створюваних СД таку інформацію отримати практично неможливо, і доводиться користуватися приблизними оцінками.

Специфікація транзакції повинна включати вимоги по продуктивності. Продуктивність БД в цілому складається з продуктивності кожної транзакції окремо і їх розподіл в часі. Це дуже важлива інформація для розв'язання задачі настройки фізичної структури СД. Вимоги по продуктивності зазвичай задаються у вигляді параметра час реакції, тобто кількості секунд, необхідних для виконання транзакції. Наприклад, у багатьох банківських системах транзакція для зняття грошей з рахунку клієнта не повинна займати більше 5 секунд.

Іншою формою завдання вимоги на продуктивність транзакцій БД є вказівка їх характеру і складності. наприклад,

- онлайнові транзакції високої складності повинні виконуватися не більше 15 с;
- онлайнові транзакції середньої складності повинні виконуватися не більше 7 с;
- онлайнові транзакції низької складності повинні виконуватися не більше 4 с;
- пакетні транзакції високої складності повинні виконуватися не більше 1 години;
- пакетні транзакції середньої складності повинні виконуватися не більше 0.5 години;
- пакетні транзакції низької складності повинні виконуватися не більше 15 хв.

Отримання вимог на продуктивність транзакцій є одним з найбільш вузьких місць як у вихідній документації, так і в проектуванні фізичної моделі

СД. На практиці отримати часові характеристики виконання транзакцій часто вдається тільки на етапі дослідної експлуатації, а то і промислової експлуатації СД.

3.3 Транзакції в сховищах даних

Транзакції, що використовують вкладені запити (підзапити) відносяться до запитів з високою складністю. Використання заздалегідь сформованого запиту – представлення (view) – замість SQL-коду підзапиту не вирішує проблеми, оскільки існування у коді запиту звернення до представлення сприймається транслятором запитів як використання вкладеного запиту.

Але всі запити, які потребують даних більшого ступеня агрегації, ніж ті, що фізично наявні в таблицях фактів сховища даних, потребують використання підзапитів або представлень з групуванням даних. Таким чином, для зменшення кількості транзакцій високої складності необхідно йти на виправдану надмірність даних у сховищі даних. Потрібно розробити всі можливі агрегати, які можуть знадобитися у виконанні транзакцій для читання від всіх основних груп користувачів. У цьому випадку не потрібно йти на економію дискового простору, тому що отримання високої швидкості виконання запитів є більш пріоритетним завданням.

Фізичне зберігання таблиць агрегатів призводить до появі у сховищі даних крім основної таблиці фактів «Продаж» чималої кількості таблиць похідних фактів, які мають меншу кількість атрибутів, наприклад, підсумовуються вартість всіх товарів, куплених усіма покупцями за день; або меншу кількість записів (рядків), якщо агрегація призводить до збільшення розміру гранулювання – зберігаються не факти покупок клієнтом протягом дня, а факти покупок протягом місяця.

Підсумовування вартості покупок за більший період без зменшення кількості вимірів для таблиці фактів може не дати істотного зменшення кілько-

сті рядків, оскільки протягом місяця людина звичайно купує декілька разів один і той товар лише у випадках покупки продуктів харчування.

3.4 Денормалізація таблиць для оптимізації транзакцій

Денормалізація являє собою набір методів настройки фізичної моделі БД, що використовується для реалізації СД, з метою задоволення вимог до продуктивності СД.

Ці методи являють собою набір рекомендацій та евристичних правил зі зміни фізичної структури БД, яка була отримана в результаті першої ітерації створення фізичної моделі даних СД. Застосування цих методик носить рекомендаційний характер.

Під денормалізації розуміють процес досягнення компромісів в нормалізованих таблицях за допомогою навмисного введення надмірності з метою збільшення продуктивності.

Денормалізація методом «розділяй і володарюй» – це процес розбиття нормалізованої таблиці на дві таблиці і більш і створення між ними відносин «один до одного» з метою усунення додаткових операцій введення-виведення або з технічних причин [11] [1)] .

Використання цього прийому зазвичай носить причини технічного характеру. Як правило, цей метод застосовується для денормалізації таблиць, що містять колонки з даними типу text або varbinary (max), розмір яких може становити 64К і більш. Іноді краще винести таку колонку в окрему таблицю.

Розглянемо таблицю, рядки якої містять на початку ключові колонки, потім неключових колонки, а в кінці – колонку типу varbinary (max). Припустимо, що в більшості рядків колонка типу varbinary (max) містить дані. Якщо немає індексів по неключових стовпцями, то при виконанні запитів за допо-

---

[1)] [11] Физическая модель хранилища данных: учет влияния транзакций, денормализация таблиц. Лекции НОУ ИНТУИТ URL: https://intuit.ru/studies/courses/599/455/lecture/10180

могою одного з цих стовпців СКБД зазвичай буде здійснювати повне сканування таблиці. При цьому через наявність в таблиці колонки типу varbinary (max) знадобляться додаткові операції введення-виведення.

Щоб усунути цю проблему, необхідно розділити таблицю так, як показано на рис. 2.

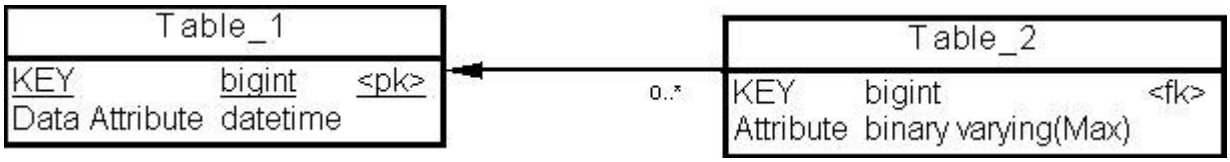

Рисунок 2 – Виділення колонки типу varbinary (max) в окрему таблицю

У деяких СКБД таблиця не може мати більше 254 стовпців, і якщо для представлення даних потрібна таблиця з великим числом стовпців, то також виникне причина для поділу такої таблиці на дві. Як правило, такі таблиці виникають в наступних випадках;

- додаток повністю проектується на основі БД успадкованої системи, і кожна таблиця будується як точна копія файлу успадкованої системи. При цьому успадковується і структура, а всі реляційні властивості в ній відсутні;
- виконується злиття двох таблиць шляхом формування в одній з них групи, що повторюється;
- проектується СД, для якого прийнято рішення виконати масову спадну денормалізацію. В цьому випадку слід створювати таблиці з максимальним для СКБД числом стовпців, так як будь-яке інше рішення, ймовірно, зумовить необхідність масових з'єднань «один до одного».

Відповідно до думки відомого фахівця в галузі проектування реляційних БД Д. Енсор, «хорошою мірою ступеня нормалізації є число стовпців на

таблицю. Емпіричне правило говорить, що далеко не всі первинні ключі мають більше двадцяти дійсно залежних від них атрибутів».

3.4.1 Розбиття таблиць як методи денормалізацію

Розбиття таблиць (splitting partition) є одним із загальних методів денормалізації, який застосовується в фізичному проектуванні СД. Розбиття таблиць буває двох видів – вертикальне розбиття і горизонтальне розбиття.

Вертикальне розбиття (vertically partition) є процесом переміщення деяких колонок таблиці в іншу нову таблицю, яка має той же первинний ключ, що і вихідна таблиця.

Горизонтальне розбиття (horizontally partition) є процесом переміщення деяких рядків однієї таблиці в іншу нову таблицю, яка має таку ж внутрішню структуру, що і вихідна таблиця.

Основні причини розбиття таблиці: або існують певні проблеми з продуктивністю запитів на цій таблиці, або підмножини рядків таблиці мають значну відмінність в продуктивності обробки, тобто підмножини рядків таблиці рідко зустрічаються в одному і тому ж запиті.

3.4.2 Вертикальне розбиття таблиць

Проблеми з продуктивністю виконання запитів на довгих рядках таблиці є найбільш частою причиною для виконання вертикального розбиття таблиці. Критеріями того, що рядок вважається довгою, може бути наступне:

- довжина рядка більше, ніж довжина фізичної сторінки бази даних (> 1 КБ);
- використання так званого індексу хеширования (cluster hashed index).

Розглянемо випадок, коли довжина рядка більше розміру фізичної сторінки БД. Тут або таблиця має занадто багато колонок, або деякі колонки мають велику довжину. В цьому випадку СКБД при вставці рядка в таблицю

буде використовувати одну або більше додаткових фізичних сторінок для збереження рядка. Отже, при вибірці рядки з таблиці потрібно більше операцій введення-виведення на число додаткових фізичних сторінок. Продуктивність запиту при цьому буде погіршуватися. Для збільшення продуктивності вибірки можна розбити таблицю на одну або кілька таблиць з довжиною рядка відповідного розміру.

Метод вертикального розбиття принципово простий, якщо згадати, що розбиття еквівалентно реляційної операції проекції на таблиці. Ясно, що деякі колонки просто переносяться в нову таблицю так, щоб довжина залишилася рядка була придатною (<1 КБ). Розбиття не повинно порушувати функціональних залежностей між колонками. Оскільки ми припускаємо, що вихідна таблиця нормалізована (зокрема, всі неключові колонки функціонально повно залежать від первинного ключа), первинний ключ нової таблиці є точною копією первинного ключа вихідної таблиці.

Критичним питанням є питання, які колонки слід переносити в нову таблицю. При розбитті таблиці збільшення продуктивності буде досягатися тільки при роздільному доступі до отриманих таблиць. При з’єднанні отриманих таблиць продуктивність такого запиту може бути нижче, ніж з тим же запитом до вихідної таблиці, через додаткове введення-виведення. Тому, перш ніж прийняти рішення про розбиття таблиці, необхідно проаналізувати всі транзакції до вихідної таблиці з високим пріоритетом і при рознесенні колонок за таблицями керуватися принципом: частота спільного використання колонок в запитах, розміщених в кожній таблиці розбиття, повинна бути досить висока.

Для прикладу розглянемо вертикальне розбиття таблиці «Службовці» (EMPLOYEE) на таблицю «Службовці» (EMPLOYEE_BASE) та таблицю «Додаткові дані» (ADD_EMPL).

Визначимо, що серед атрибутів таблиці EMPLOYEE є 4 атрибути, які використовуються у запитах на порядок рідше. Це атрибути:

- Дата народження (Age);
- Стаж (HIREDATE);
- Автобіографія (Biog);
- Фотографія (Foto).

Для оптимізації транзакції бажано перенести перераховані атрибути у іншу таблицю, як показане ри рис. 3.

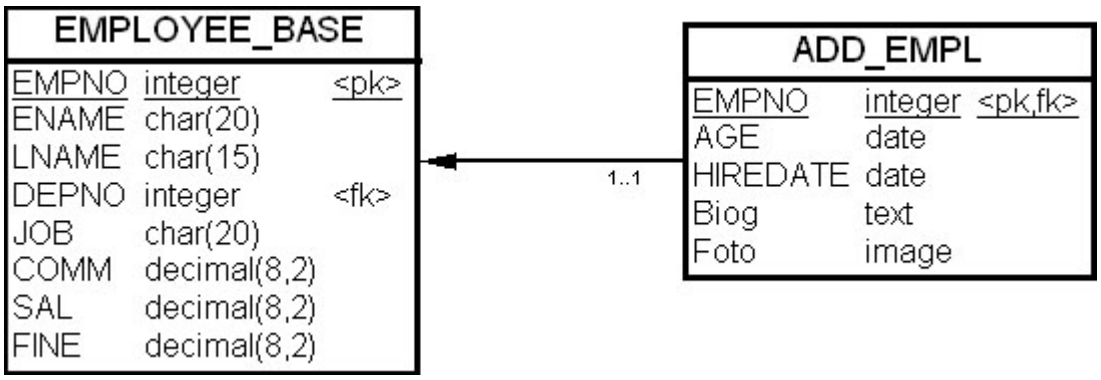

Рисунок 3 – Вертикальне розбиття таблиці EMPLOYEE на дві таблиці: EMPLOYEE_BASE і ADD_EMPL

Крім підвищення продуктивності операцій вибірки в якості обґрунтування вертикального розбиття таблиць слід вказати на підвищення продуктивності при операціях вставки і видалення, а також при операціях резервного копіювання та відновлення БД або СД.

3.4.3 Горизонтальне розбиття таблиць

На практиці горизонтальне розбиття (horizontally partition) застосовується для ізоляції однієї групи рядків таблиці від іншої, коли використання цих груп рядків в транзакціях майже не перетинається. Типовий приклад – ізоляція поточних даних від архівних даних.

Розглянемо систему обробки замовлень. Менеджери і продавці працюють з поточними замовленнями. Обробка виконаних замовлень (архівні дані) виконується при підготовці різного роду звітів (зокрема, шляхом створення

кіоску даних). Навіть якщо готується щоденний звіт зі зверненням до архівних даних, то в організаціях середнього розміру частота використання поточних даних все одно перевищує частоту використання архівних даних на 2-3 порядки, а відношення обсягу поточних даних до архівних даних може становити менше 0.001.

Одним з практичних критеріїв в даному випадку може служити класичне правило 80-20. Якщо активно працюють з 20-ма відсотками даних, то найімовірніше, інші 80% можна перенести в архівну таблицю.

Розбиття таблиць і посилальна цілісність.

Якщо до розбиття таблиця була нормалізована, то первинні ключі будуть ідентичні для таблиць, отриманих в результаті розбиття. Отже, взаємозв'язки нових таблиць з іншими таблицями бази даних можуть залишитися такими ж, як і з початковою таблицею. Однак необхідно розглянути і дію правил видалення для вихідної таблиці, і участь нової таблиці в існуючих взаєминах посилальної цілісності, коли буде вирішуватися питання про визначення підтримки посилальної цілісності в нових таблицях.

У деяких випадках може бути простіше не визначати кількість посилань цілісність зовсім, так як фактичне видалення рядків з нової таблиці буде підтримуватися паралельно з вихідною таблицею будь-яким програмним способом в додатку БД (наприклад, за допомогою тригера).

При прийнятті рішення про розбиття таблиці слід дотримуватися наступного алгоритму:

- визначити, які колонки вихідної таблиці в які нові таблиці будуть переміщені;
- створити нові таблиці з первинним ключем, ідентичним первинному ключу вихідної таблиці;
- якщо СКБД буде управляти декларативною посилальної цілісністю для нових таблиць таким же чином, як і для вихідної таблиці, в разі якщо вона є дочірньою таблицею у взаємозв'язку, слід додати коло-

нку зовнішнього ключа кожної батьківської таблиці у взаємозв'язку в нову таблицю, тобто нова таблиця повинна містити обмеження зовнішнього ключа, ідентичне батьківській таблиці, для кожного взаємозв'язку. Альтернативою цьому рішенню є створення зв'язку «один до одного» між новою таблицею і вихідною таблицею, визначення зовнішнього ключа назад до вихідної таблиці тотожним первинному ключу;

- якщо СКБД буде управляти посилальної цілісністю для нових таблиць таким же чином, як і для вихідної таблиці, в разі якщо вона є батьківської таблицею у взаємозв'язку, то слід додати зовнішній ключ в кожну дочірню таблицю вихідної таблиці, щоб ідентифікувати цю нову таблицю як дочірню. Можна так само вчинити, як в першій частині попереднього пункту, при цьому нові таблиці не слід оголошувати як батьківські в інших взаєминах. Вихідна таблиця підтримує всі взаємозв'язки, в яких вона виступає батьком;
- слід прописати для розробників додатків все команди INSERT для отриманих в результаті розбиття таблиць або вказати правила, яким має слідувати вставка рядків в ці таблиці;
- слід змінити всі представлення, які грунтувалися на вихідній таблиці, і, якщо потрібно, задуматись над створенням нових представлень для доступу до нових таблиць.

У більшості випадків необхідність денормалізації стає очевидною лише на етапі проектування додатків СД або його експлуатації. Іншими словами, не можна прийняти рішення про денормалізацію на підставі однієї тільки моделі даних. Зазвичай намагаються знайти в додатках СД критичні процеси і приймати рішення про денормалізацію в основному на користь цих процесів. Критичні процеси, як правило, визначають по високій частоті, великому обсягу, високої мінливості або явного пріоритету. Якісне опис транзакцій БД дозволяє визначити наявність таких критичних процесів.

3.5 Денормалізація таблиць сховища даних компанії

Для підвищення ефективності роботи сховища даних, що розробляється, також можна вдатися до денормалізацію таблиць.

Вище при розгляді обмежень цілісності таблиць було зауважено, що крім визначення ключів таблиць: первинного, унікального, зовнішніх, – основним обмеженням цілісності у сховище даних є відсутність значень NULL в таблицях. Але завдання обмеження NOT NULL для всіх атрибутів всіх таблиць дуже обмежує бізнес-процеси, що можуть проводитись з використанням корпоративного сховища даних.

Розглянемо це на прикладі таблиці-виміру «Покупець» (Customer).

Відомо, клієнтська база даних є важливим інформаційним ресурсом компанії. Для поповнення клієнтської бази даних компанії проводять соціологічні опитування, акції по залученню нових клієнтів шляхом надання бонусів існуючим клієнтам за приведення нового клієнта.

При реєстрації клієнта існує дуже невелика множина полів картки реєстрації, які клієнт повинен заповнити в обов’язковому порядку. І серед цих обов’язкових атрибутів немає характеристик, по яких можна було провести аналіз залежності купівельного попиту від факту знаходження клієнта у заданій соціальній групі. У більшості Інтернет-магазинів необов’язковими є навіть поля «дата народження» та «стать».

Але цілком зрозуміло, що особи, що належать до різних соціальних та вікових груп споживають товари, відмінні по кількості, набору, вартості. Для ефективної роботи в сфері Інтернет продажів потрібно орієнтуватись у закономірностях формування попиту на товари у різних групах населення: вікових, соціальних, освітніх, тощо.

Таким чином, виникає протиріччя між цінністю зберігання необов’язкових атрибутів клієнтів і вимогою неприпустимості значень NULL для тих клієнтів, що не побажали надати подібні відомості.

У цих випадках стане у нагоді денормалізація таблиці Customer шляхом горизонтального розбиття на декілька таблиць.

Цей процес досить простий, якщо в таблиці Customer один або два необов'язкових. Якщо необов'язковий атрибут один, розбиваємо таблицю Customer на дві за ознакою наявності або відсутності значення необов'язкового атрибуту.

Наприклад, атрибут «день народження» є необов'язковим. Створюємо нову таблицю CustomerNoBirth. У нової таблиці будуть ті самі атрибути, що і у таблиці Customer, але не буде атрибуту Birthdate.

Для перенесення в таблицю CustomerNoBirth записів з таблиці Customer використовується наступний SQL-код транзакції:

```
INSERT INTO CustomerNoBirth
SELECT   FName, LName, EMail, Phone,  Cust_City
FROM Customer
WHERE Birthdate IS NULL
DELETE Customer WHERE Birthdate IS NULL
GO
```

Якщо потрібно буде у звіт включити відомості про всіх клієнтів без даних о даті народження та незалежно від наявності даних про дату народження, потрібно буде об'єднати записи з двох таблиць, використав опцію UNION:

```
SELECT  FName, LName, EMail, Phone,  Cust_City
FROM Customer
UNION
SELECT  FName, LName, EMail, Phone,  Cust_City
FROM CustomerNoBirth
GO
```

Але наявність у сутності декількох необов'язкових атрибутів робить схему більш складною. Так наявність у сутності Customer двох необов'язкових атрибутів і вимога відсутності значень NULL в таблицях СД призведе до розбиття однієї таблиці Customer на чотири за схемою:

- Customer11 – в таблиці є обидва необов'язкових атрибута;
- Customer10 – в таблиці є один перший необов'язковий атрибут;
- Customer01 – в таблиці є один другий необов'язковий атрибут;
- Customer00 – в таблиці немає жодного необов'язкового атрибуту.

Наприклад, необов'язковими атрибутами є дата народження (перший атрибут) та рівень освіти (Education – другий атрибут).

Нехай таблиця Customer10 зберігає дані про тих клієнтів, що вказали свою дату народження, але не вказали рівень освіти.

Тоді для перенесення записів про всіх цих клієнтів з таблиці Customer до таблиці Customer10 потрібно виконати транзакцію з наступним SQL-кодом:

```
 INSERT INTO Customer10
SELECT   FName, LName, EMail, Phone,  Cust_City, Birthdate
FROM Customer
WHERE Birthdate IS NOT NULL AND Education IS NULL
DELETE Customer
WHERE Birthdate IS NOT NULL AND Education IS NULL
GO
```

Аналогічно, таблиця Customer01 зберігає дані про тих клієнтів, що вказали свій рівень освіти, але не вказали дату народження.

Для перенесення записів про всіх цих клієнтів з таблиці Customer до таблиці Customer01 потрібно виконати транзакцію з наступним SQL-кодом:

```
INSERT INTO Customer01
SELECT   FName, LName, EMail, Phone,  Cust_City, Education
FROM Customer
```

```
WHERE Education IS NOT NULL AND Birthdate IS NULL
DELETE Customer
WHERE Education IS NOT NULL AND Birthdate IS NULL
GO
```

Горизонтальне розбиття таблиць оптимізує запити до БД внаслідок того, що для кожної з таблиць, отриманої після розбиття, спрощується предикати пошуку даних у розділі WHERE. Крім того, кожна з результуючих таблиць має менший розмір за кількістю записів, а деякі і за кількістю атрибутів. Для кожної з цих таблиць вибірка даних буде транзакцією низької складності. Об’єднання результатів вибірки з декількох таблиць за допомогою опції UNION не додає транзакції рівня складності.

Але наведений процес горизонтального розбиття таблиць може призвести до зростання складності транзакцій вибірки даних, якщо кількість атрибутів таблиці, за якими виконується розбиття, перевищує 2.

Таким чином змінюється схема сховища даних. У схему додаються таблиці фактів різного ступеня агрегації. Крім того, у схему додаються таблиці в наслідок процесів вертикального або горизонтального розбиття таблиць. Уточнена схема БД сховища даних наведена у додатку А.

4 ПРОГРАМНІ ЗАСОБИ РОЗРОБКИ СХОВИЩА ДАНИХ

4.1 Мова запитів

Однією з головних переваг реляційних систем керування базами даних (РСКБД) над системами, заснованими на інших моделях даних є мова структурованих запитів SQL (Structured Query Language). Мову SQL називають мовою реляційних баз даних.

Завдяки цій мові можна виконувати вибірки з баз даних як завгодно складної структури. Мова SQL дозволяє знаходити узагальнені (агреговані) дані, які використовуються, наприклад, в корпоративних сховищах даних.

Переваги SQL:

1. Наявність міжнародних стандартів.
2. Незалежність від конкретної СКБД. Незважаючи на наявність діалектів і відмінностей в синтаксисі, в більшості своїй тексти SQL-запитів можуть бути досить легко перенесені з однієї СКБД в іншу.
3. Підтримка архітектури клієнт-сервер.
4. Декларативність. За допомогою SQL програміст описує тільки те, які дані потрібно витягнути або модифікувати. Яким чином це зробити, вирішує СКБД безпосередньо при обробці SQL-запиту.
5. Поширеність.
6. Швидке навчання.

Основні конструкції мови SQL затверджені міжнародними стандартами і однакові в усіх СКБД незалежно від компанії-розробника, так само, як і в відкритих СКБД. До стандартів SQL відносяться і конструкції мови, які дозволяють створювати агрегати – групувати дані за деякими критеріями. Цей факт важливий для розробки сховищ даних, оскільки при експлуатації СД основні групи користувачів – керівники різних рівнів компанії та аналітики – використовують у своїх запитах саме агреговані дані.

Комерційні СКБД мають більше можливостей для розробки сховищ даних та застосувань для бізнес-аналітики. Використання вільних СКБД призведе до необхідності розробляти власні засоби для інтеграції даних та застосування для бізнес-аналітики, або купувати відповідні програмні продукти на ринку програмного забезпечення. Але ці продукти коштують досить дорого.

У останні роки компанія Microsoft випускає безкоштовні редакції своїх комерційних продуктів, наприклад, СКБД SQL Server та середовище швидкої розробки проектів і додатків Visual Studio. Звичайно, безкоштовна редакція має дещо обмежений функціонал у порівнянні з комерційними редакціями, а також призначена для використанні у навчальних закладах та для малого бізнесу. Але якщо компанія входить у ці обмеження і її задовольняє наявний функціонал програмних систем, то вибір цих продуктів для створення СД є цілком виправданим.

Незважаючи на наявність стандартів мови SQL, практично в кожній СКБД застосовується свій діалект мови. Одним зі діалектів SQL є мова T-SQL.

Мова T-SQL – це власний діалект мови структурованих запитів, який застосовується в СКБД SQL Server. Мова T-SQL призначена виключно для роботи з СКБД SQL Server, але основна частина операторів, що застосовується у ній, має більш широке використання.

Основна відмінність між SQL та T-SQL є те, що SQL – це мова, яка використовується для зберігання і управління даними в СRБД, а T-SQL є розширеною версією SQL і призначена для виконання операцій на сервері MS SQL Server.

Додатки можуть зв'язуватися з SQL Server, виконуючи оператори T-SQL. Розробник може писати запити для виконання операцій над таблицями, об'єднання таблиць і додавання обмежень. Він також може виконувати транзакції, писати збережені процедури, подання, індекси і багато іншого. Іс-

нують різні числові, рядкові функції, функції дати. Крім того, існують функції агрегування для виконання операції з набором значень.

При підготовці випуску СКБД SQL Server 2005 мова T-SQL була в значній мірі доопрацьована, і в неї були додані багато нових програмних конструкцій. Крім усього іншого, вона була перетворена в мову, сумісну із загальним середовищем виконання (Common Language Runtime CLR) операційної системи Windows; таким чином, починаючи з цього випуску T-SQL стала однією з мов .NET.

В усіх версіях СКБД SQL Server, починаючи з версії SQL Server 2005, передбачена можливість використовувати для доступу до бази даних будь-яку мову .NET, але в кінцевому підсумку в основі звернення безпосередньо до самих даних завжди лежить якийсь код SQL, тому мова T-SQL залишається базовою мовою при виконанні будь-яких дій в СКБД SQL Server. З кожним випуском SQL Server багато змінюється, але найбільш фундаментальні оператори доступу до даних є практично такими ж, як і раніше.

T-SQL – це процедурне розширення мови SQL компаній Microsoft. SQL була розширена такими додатковими можливостями як:

- керуючі оператори;
- локальні і глобальні змінні;
- різні додаткові функції для обробки рядків, дат, математики тощо;
- підтримка аутентифікації Microsoft Windows.

Мова T-SQL є ключем до використання SQL Server. Всі додатки, які взаємодіють з екземпляром SQL Server, незалежно від їх реалізації і призначеного для користувача інтерфейсу, відправляють серверу інструкції T-SQL.

4.2 Збережені процедури у мові T-SQL

Збережені процедури і функції представляють собою набір SQL-операторів, які можна зберігати на сервері. Якщо сценарій збережений на

сервері, то клієнтам не доведеться повторно ставити одні й ті ж окремі оператори, замість цього вони зможуть звертатися до збереженій процедурі [12] – [13] [1)].

Ситуації, коли збережені процедури особливо корисні:

– Численні клієнтські програми написані на різних мовах або працюють на різних платформах, але повинні виконувати однакові операції з базами даних.
– Безпека грає першорядну роль. Збережені процедури використовуються для всіх стандартних операцій, що забезпечує сумісність і безпеку середовища, а процедури гарантують належну реєстрацію кожної операції. При такому типі установки додатка, і користувачі не отримують безпосередній доступ до таблиць бази даних і можуть виконувати тільки конкретні процедури, що зберігаються.
– Необхідно знизити мережевий трафік між клієнтом і сервером. Обсяг інформації, що пересилається між сервером і клієнтом істотно знижується, але збільшується навантаження на систему сервера баз даних, так як в цьому випадку на стороні сервера виконується велика частина роботи по обробці даних.

[1)] [12] Хранимые процедуры. URL: https://metanit.com/sql/sqlserver/11.1.php

[13] CREATE PROCEDURE (Transact-SQL). Microsoft. Документация по SQL. URL: https://docs.microsoft.com/ru-ru/sql/t-sql/statements/create-proceduretransact-sql?view=sql-server-ver15

5 РОЗРОБКА СЕРВЕРНОГО ПЗ ДЛЯ СД КОМПАНІЇ

Перетворення схеми з розбиттям таблиць виконується на етапі проектування бази даних для СД. Але на етапі експлуатації СД буде виникати задача перенесення даних з однієї таблиці більшого розміру у декілька таблиць більш простої структури. Ця задача виникає через те, що дані у СД будуть поступати з транзакційних систем, що є нормалізованими БД.

Таким чином, однією з задач перетворення даних при запису їх у сховище даних буде саме перенесення даних, що містяться в одній таблиці нормалізованої БД транзакційної системи у декілька таблиць БД сховища даних.

Збережена процедура DivideCustomer виконує перенесення даних з таблиці Customer до 4 таблиць: CustomerBE (є дані про дату народження та рівень освіти), CustomerB (є дані про дату народження, немає даних про рівень освіти), CustomerE (є дані про рівень освіти, немає даних про дату народження) ,CustomerNoBE (немає даних про дату народження та рівень освіти). Процедура без параметрів.

6 СИСТЕМИ БІЗНЕС-АНАЛІТИКИ І СХОВИЩЕ ДАНИХ

У сучасному світі успіх компанії на ринку безпосередньо залежить від того, як швидко менеджмент компанії може розпізнати зміни динаміки ринку і наскільки своєчасно може відреагувати на них з метою збільшення прибутку, виходячи з існуючих реалій ринку. Менеджери компанії повинні відслідковувати тенденції ринку, ідентифікувати конкурентів і загрози, оцінювати ризики, перетворювати стратегію компанії, оцінювати свої ресурси і т.д. Інформація є необхідним виробничим ресурсом для прийняття ефективних управлінських рішень.

Компанії накопичили значні обсяги даних і мають доступ до ще більших обсягів зовнішніх даних. Менеджерам необхідно, щоб ця інформація була перетворена, попередньо оброблена і відповідним чином організована для швидкого доступу, аналізу та прийняття рішень. Такий підхід до даних є, з одного боку, створенням конкурентної переваги, а з іншого боку – вимогою до публікації даних для менеджерів компанії. Публікація даних для менеджерів, що забезпечує швидкий доступ до даних, виконання аналізу даних і інформаційна підтримка процесу прийняття рішень, є основною метою систем бізнес-аналітики. Бізнес-аналітика допомагає компанії створювати знання з усієї доступної інформації для прийняття ефективних управлінських рішень і перетворення цих рішень в дію.

Таким чином, ключову роль в управлінні організацією в цілому та її окремими виробничими функціями грає інформація. Дані, які доступні менеджерам і аналітикам безпосередньо з корпоративних інформаційних систем, не уніфіковані, розрізнені і в загальному випадку не готові для аналізу. Системи ділової обізнаності або бізнес-аналітики є тим класом інформаційних систем, який дозволяє перетворити дані корпоративних інформаційних систем і дані із зовнішніх джерел в корисні для бізнесу інформацію і знання, які використовуються в управлінні, на основі яких можна приймати рішення.

Інформаційним фундаментом для бізнес-аналізу і систем бізнес-аналітики є сховище даних. Основна вимога до сховища даних системи бізнес-аналізу полягає в тому, щоб забезпечити інформаційне середовище – структуроване і організоване для вирішення завдань бізнесу. Як правило, таке середовище лаконічно представляють у вигляді інформаційної піраміди, як показано на рис. 4.

Рисунок 4 – Інформаційна піраміда

Інформаційна піраміда формується з декількох рівнів.

– Рівень оперативної інформації. На цьому рівні ІТ забезпечують роботу з даними на рівні бізнес-процедур компанії. Дані в автоматизованих системах є добре структурованими і детальними. З цими даними працюють фахівці компанії: бухгалтери, менеджери продажів, плановики і т.д.

- Рівень тактичної інформації. На цьому рівні ІТ забезпечують інтеграцію даних на рівні бізнес-процесів оперативного управління виробництвом в рамках підрозділів компанії. З цими даними працюють керівники підрозділів компанії при виконанні щоденних виробничих завдань.
- Рівень стратегічної інформації. На цьому рівні ІТ забезпечують інтеграцію даних на рівні бізнес-процесів за напрямками господарської діяльності компанії. З цими даними працюють аналітики і керівники вищої ланки компанії, які готують стратегічні рішення розвитку і діяльності компанії на ринку.
- Рівень прийняття рішень. На цьому рівні ІТ забезпечують інтеграцію і агрегацію даних на рівні бізнес-процесів компанії для керівників вищої ланки компанії. Цей рівень забезпечує інформаційну підтримку прийняття рішень.

Інформаційна піраміда описує середовище бізнес-аналітики, яке можна описати таким чином. В інформаційне середовище бізнес-аналітики надходить первинний матеріал – дані, які потім переробляються в автоматизованих системах та інформаційних продуктах.

У процесі переробки відбувається перехід від даних до інформації. СД отримує дані з множини транзакційних або оперативних систем, а потім інтегрує і зберігає дані в спеціалізованій БД. Наприклад, в СД можуть приводитися у відповідність і об'єднуватися для користувача записи з чотирьох оперативних систем (додатків для обробки замовлень, обслуговування, продаж і поставок). Такий процес вилучення та інтеграції перетворює дані в новий інформаційний продукт – інформацію.

Потім користувачі, що працюють з аналітичними інструментами (наприклад, для створення запитів, звітів, OLAP-аналізу і виконання операцій інтелектуального аналізу даних), звертаються до даних з СД і аналізують її.

Таким чином, визначаються тенденції, структури і виключення. Аналітичні інструменти допомагають користувачам перетворити інформацію в знання.

Система ділової обізнаності, або бізнес-аналітики (BI System), – це система управління базою знань підприємства, яка надає ряд нових можливостей в існуючій інформаційній системі підприємства для аналізу бізнесу і управління базою знань підприємства.

До основних функцій системи бізнес-аналітики, як правило, відносять такі.

- Управління в реальному часі бізнес-знаннями в рамках всього підприємства.
- Простий доступ до інформації для співробітників компанії різних рівнів.
- Зростання обсягу інформації, що переробляється, підвищення конкурентоспроможності.
- Проведення більш ефективного аналізу доходів і витрат.
- Надання виконавчим директорам комплексної і більш наочної картини підприємства в усіх напрямках бізнесу.
- Відстеження ситуації як на всьому підприємстві в цілому, так і на якихось конкретних проблемних ділянках зокрема.

До основних технологічних засобів реалізації функціональності систем бізнес-аналітики відносять:

- звіти та засоби їх створення;
- спеціалізовані засоби створення звітів;
- генератори звітів, вбудовані в засоби розробки;
- нетрадиційні засоби створення звітів;
- OLAP-засоби;
- клієнтські OLAP-засоби;
- серверні OLAP-средства;
- засоби пошуку закономірностей (Data Mining-засоби) і т.д.

Система бізнес-аналітики є стрижнем, навколо якого формуються потоки стратегічної бізнес-інформації. Даний інструмент допомагає компанії приймати рішення, які будуть засновані на коректної інформації, отриманої вчасно.

В умовах, коли ринок постійно змінюється, а конкуренція стає все жорсткішою, керівникам вкрай необхідно виявляти і аналізувати наявні у підприємства резерви, які можуть істотно розширити можливості бізнесу.

Пропоновані рішення в області бізнес-аналітики повинні надавати можливість оперативно аналізувати тенденції ринку, усвідомлювати рушійні сили бізнесу і, грунтуючись на об'єктивній інформації, швидко реагувати на зміни ринкової ситуації і приймати правильні рішення.

Система бізнес-аналітики повинна:

- мати єдиний графічний інтерфейс користувача (GUI), який забезпечує інтуїтивну навігацію і комфортність в роботі;
- надавати гнучкий аналіз і звітність співробітникам різних підрозділів для перегляду зведених даних, використовуючи одні й ті ж розрізи діяльності в порівнянних показниках;
- мати відкриту архітектуру і організаційну масштабованість, які дають контрольоване, послідовне і швидке розгортання аналітичної системи у всіх підрозділах підприємства;
- мати потужну систему адміністративного контролю, яка звільняє IT-службу від необхідності складання багатьох форм звітності, але в той же час забезпечує контроль доступу до баз даних, конфіденційність і моніторинг змін;
- забезпечувати швидке впровадження, яке сприяє швидкому отриманню практичних результатів і прискорює віддачу від інвестицій в програмне забезпечення.

Таким чином, системи бізнес-аналітики дозволяють:

- краще аналізувати структуру купівельного попиту і потреби ваших клієнтів;
- використовувати гнучку стратегію продажів і цільовий маркетинг;
- по-новому представити на ринку можливості вашої продукції;
- виявити слабкі місця в ланцюжку поставок;
- виключити невиправдані виробничі і комерційні витрати;
- виявити стратегічні тенденції в зміні цінової структури і номенклатури продукції, що випускається.

6.2 Реалізація архітектури системи бізнес-аналітики

Сьогодні на ринку інформаційних технологій представлено широкий спектр інструментальних засобів, призначених для швидкої реалізації компонентів архітектури системи бізнес-аналітики. Застосування таких інструментів дозволяє не розробляти аналітичні додатки заново, а скористатися готовими сучасними технологіями і, отже, скоротити час і витрати на їх створення.

Рішення завдання забезпечення користувачів інформацією в системі бізнес-аналітики визначається в основному правильним підбором інструментів ділового аналізу. Але важливим є і вибір інструментів підтримки процесів вилучення, перетворення, завантаження і зберігання даних.

При реалізації системи бізнес-аналітики підприємства можуть бути використані програмні рішення як різних фірм-виробників (змішані рішення), так і одного виробника (переносних базовані рішення). І в першому, і в другому випадку є свої переваги і недоліки. Тому вибір інструментів для реалізації архітектури системи бізнес-аналітики, незважаючи на їх різноманіття, – завдання не з простих.

На ринку не існує одного виробника, що пропонує кращі рішення всіх необхідних для побудови системи бізнес-аналітики програмних компонентів.

Тому спільне використання найоптимальніших рішень від різних виробників дозволяє підвищити функціональну потужність системи бізнес-аналітики. Критеріями оцінки інструментів можуть виступати як їх технічні і вартісні характеристики, так і швидкість впровадження, а також доречність використання в кожному конкретному випадку.

Однак використання продуктів від різних виробників призводить до значного ускладнення архітектури системи через різнорідності інструментальних рішень. Це ускладнення пояснюється необхідністю інтегрування не пов'язаних один з одним інструментальних рішень. Крім того, адміністрування системи виявляється непростим завданням, враховуючи неузгодженість даних і метаданих, керованих окремими, не пов'язаними один з одним модулями платформ від різних виробників.

При реалізації архітектури системи бізнес-аналітики від одного виробника (якщо користуватися термінологією дослідного центру Gartner, переносних базуватися рішення) рішення необхідно шукати серед фірм-виробників так званих BI-платформ (Business Intelligence Platforms).

Даний сегмент ринку інформаційних технологій представлено більш ніж 20-ма компаніями, такими як (в алфавітному порядку): AlphaBlox, Arcplan, CA, Comshare, Crystal, Hyperion, Info Builders, Microsoft, Micro?trategy, Oracle, PeopleSoft, ProClarity, Sagent, SAP, SAS, Whitelight і ін. Серед них виділяються наступні сім лідерів і претендентів на лідерство в цій галузі: Microsoft, SAS, Oracle, SAP, PeopleSoft, Info Builders, Hyperion

Двоє з перерахованих виробників, Microsoft і Oracle, в стані реалізувати всі рівні системи бізнес-аналітики своїми силами, не вдаючись до інструментів третіх фірм. Вирішальний критерій, що виділяє цих виробників, – наявність власної СКБД.

6.3. Побудова систем бізнес-аналітики: проблеми та рішення

Інформаційні технології забезпечують підтримку технологічного ланцюжка обробки даних:

- збір і отримання даних;
- перетворення даних;
- надання даних.

Отримання даних забезпечується автоматизованими системами оперативної обробки даних або транзакційними системами обробки даних. Основне призначення таких систем – це забезпечення розвиненою форми обліку даних на низькому рівні бізнес-процесів організації. Користувачами цих систем є фахівці.

Щоб використовувати зібрані дані для аналізу, їх потрібно привести до єдиного формату, перетворити, узгодити і попередньо обробити. Це завдання призначені вирішувати системи вилучення, перетворення і завантаження даних (ETL). Це важлива ланка переходу до аналізу даних.

Надання даних забезпечується інформаційно-аналітичними системами обробки даних. Такі системи розробляються з використанням технології СД і методів бізнес-аналітики. Основне призначення таких систем – це забезпечення розвиненою форми публікації даних. Їх користувачами є менеджери.

Публікація даних для менеджерів є першочерговим завданням. Добре відомо, що публікація є успішною, якщо вона задовольняє потреби читачів. Своєчасна і по можливості повна публікація даних є середовищем для підтримки і прийняття рішень.

Для менеджера важливо, щоб публікація була:

- суттєвою для вирішення поточних бізнес-завдань;
- зрозумілою і простою у використанні;
- швидкої;
- ефективною в співвідношенні "ціна / якість".

6.4 Сховища даних і системи бізнес-аналітики

Одним з головних призначень системи бізнес-аналітики є публікація даних в форматі, зручному для прийняття рішень, і надання простих і зручних інструментів керівникам організації для маніпулювання цими даними з метою їх дослідження та аналізу.

Одним з головних призначень СД – накопичення даних з різних джерел для аналітичної обробки і відділення формування звітів і проведення інтелектуального аналізу даних від систем оперативної обробки даних підприємства з метою збільшення продуктивності.

Таким чином, СД представляє собою дуже великі репозиторії історичних даних, а системи бізнес-аналітики представляють собою різною мірою взаємопов’язаний набір додатків для бізнес-аналізу цих даних.

Нижче, на рис. 5, показано, як система бізнес-аналітики взаємодіє зі СД.

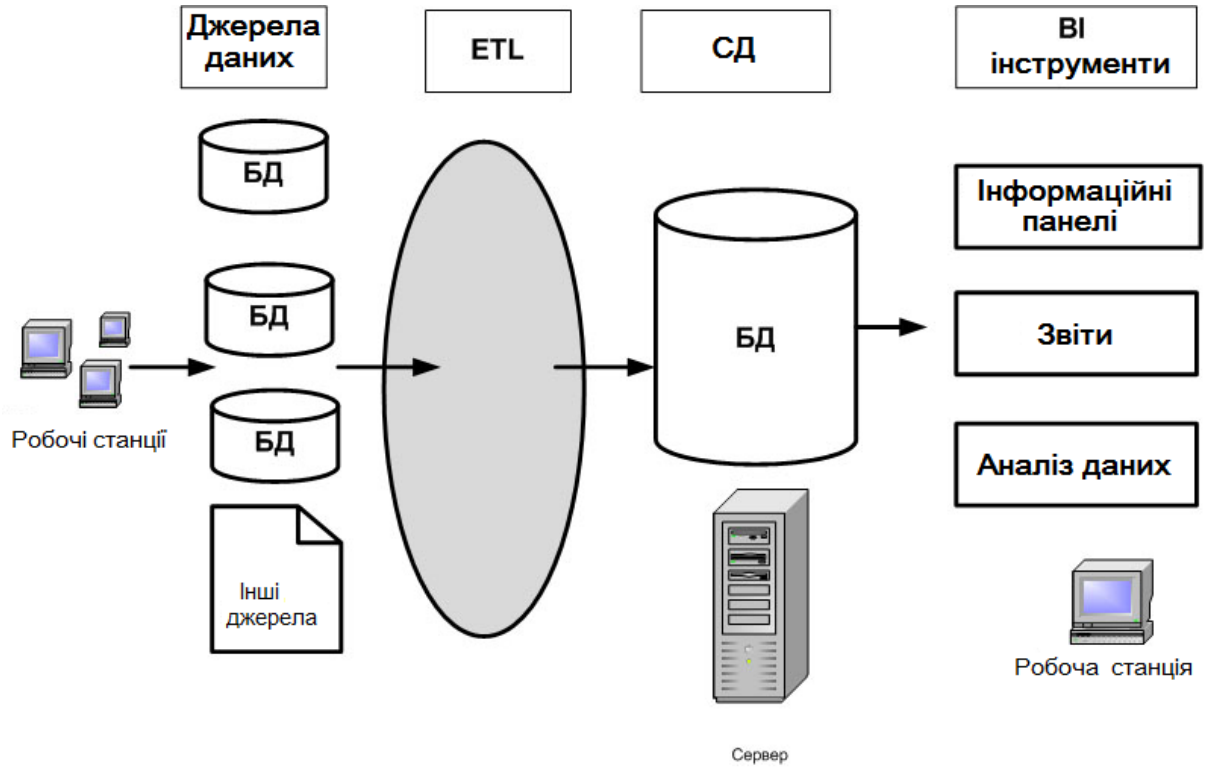

Рисунок 5 – Взаємодія системи бізнес-аналітики та СД

Розробники систем бізнес-аналітики є по суті видавцями. Вони збирають дані з різних джерел, редагують їх для забезпечення якості та узгодженості, забезпечують довіру до опублікованої інформації. Їх успіх оцінюється кінцевими бізнес-користувачами: аналітиками, менеджерами різних рівнів і керівництвом організації.

СД підтримує системи бізнес-аналітики, будучи її інформаційним фундаментом.

Завдання, які вирішують системи бізнес-аналітики:

- Запити та звітність:
  - Публікація правильних даних.
- Пошук винятків:
  - Візуалізація, визначення меж, порівняння, попередження.
- Аналіз причинно-наслідкових зв'язків:
  - Що взаємопов'язане.
- Моделювання потенційних рішень:
  - Що якщо...
- Відстеження результатів прийнятих рішень.

З цієї точки зору СД допомагає вирішувати основні завдання підтримки систем бізнес-аналітики.

- Максимізація цінності інтелектуального капіталу:
  - застосування знань у певній предметній області;
  - демонстрація інтуїції в простих діях;
  - прийняття рішень;
  - досягнення розуміння серед всіх осіб, які приймають рішення;
- і одночасна мінімізація витрат:

розробка;

  - адміністрування (стандартні операції, маленькі сюрпризи, великі сюрпризи);

– очевидні витрати (персонал, апаратне і програмне забезпечення);
– приховані витрати (упущені можливості, відхилення).

На рис. 6 показано, як сховища даних управляють системами бізнес-аналітики.

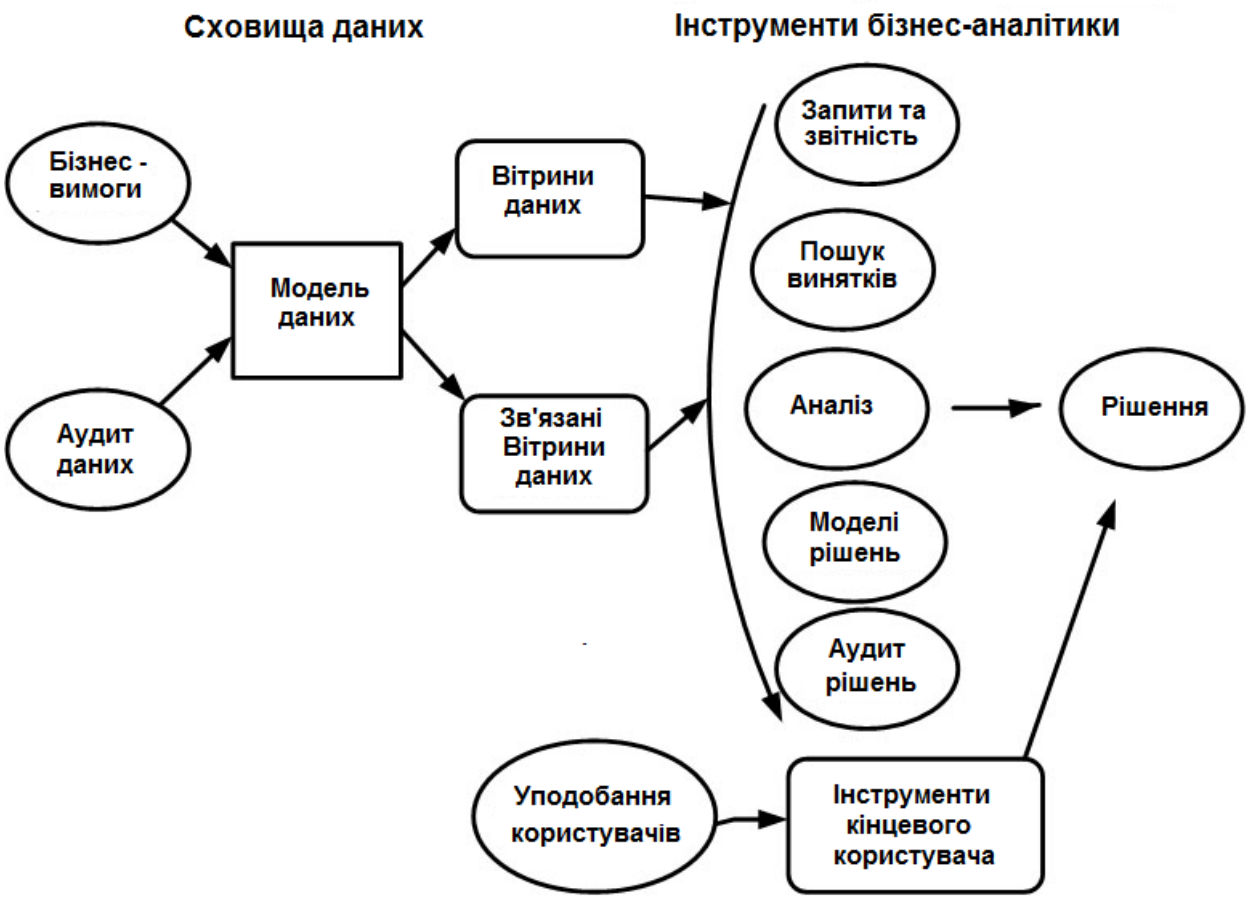

Рисунок 6 – Схема управління сховищем даних
системами бізнес-аналітики

На практиці СД функціонують відповідно до наступних положень:

– децентралізована, інкрементальний розробка;
– різні технології, не сумісні на найпростішому рівні;
– швидка розробка;
– постійно змінюється середовище / постійно мінливі пріоритети;
– потенційно несумісні вітрини даних;

- негайна реакція;
- атомарні дані, дані в режимі реального часу, безперервна історія;
- всебічне уявлення про замовника;
- відстеження, зберігання, передбачення поведінки.

Таким чином, СД через збережені в них дані керують системами бізнес-аналітики і впливають на якість і ефективність прийнятих рішень. Щоб дані сприяли прийняттю якісних рішень, вони повинні бути добре організовані.

Як видно з опису функцій системи бізнес-аналітики, вона представляє собою досить складний програмний продукт. Використання неякісної системи бізнес-аналітики призведе до прийняття не правильних рішень керівниками компанії, отже принесе не користь, а шкоду.

Оскільки від якісної системи бізнес-аналітики залежить політика компанії, бажано не розробляти власну дешевий та неякісний продукт, а розумно витратити кошти на придбання надійного і якісного програмного продукту.

ВИСНОВКИ

Продаж товарів через мережу Інтернет в останні роки стає все більш популярним. Для показу товарів можливим покупцям на сайті компанії мають базу даних цих товарів.

Але для оцінки ефективності бізнесу та перспектив розвитку компанії потрібно мати сховище даних (СД), в якому містяться як поточні, так і історичні дані, міститься вся інформація, яка дозволяє оцінювати ефективність бізнесу та перспектив його розвитку: база даних покупців, постачальників товарів, продажів товарів, маркетингових акцій, тощо.

Це сховище даних повинне містити агреговані дані по продажу товарів, які використовуються аналітиками компанії для аналізу попиту на товари та впливу на попит та продаж товарів різноманітних факторів: сезонів року, місце проживання покупця, віку покупця, проведення маркетингових компаній, тощо.

За звітами аналітиків керівництво нижньої та середньої ланки розробляє поточну політику компанію, направлену на підвищення прибутку та розвиток компанії. Керівництво верхньої ланки розробляє стратегію розвитку компанії.

Метою даної комплексної кваліфікаційної роботи було моделювання, проектування та розробка сховища даних для компанії з Інтернет-продаж.

Завдання даної частини комплексної роботи була розробка сховища даних.

При виконанні кваліфікаційної роботи були отримані наступні результати:

- було проведено дослідження сучасних архітектур сховищ даних;
- були проаналізовані переваги та недоліки хмарних сховищ даних та озер даних;

- була досліджена та проаналізована логічна модель сховища даних, отримана у першій частині даної комплексної магістерської роботи;
- було прийняте рішення про доробку схеми БД сховища даних для оптимізації запитів – зменшення часу виконання запиту;
- були проаналізовані можливі рішення денормалізації БД сховища даних для підвищення ефективності виконання запитів;
- для зменшення часу виконання транзакцій змінена схема БД сховища даних – таблиця-вимір Product горизонтально розбита на 2 таблиці;
- для більш швидкого доступу до даних і зменшення часу виконання транзакцій створені додаткові індекси, не з ключових атрибутів .

Задачі, поставлені в технічному завданню на виконання кваліфікаційної роботи повністю виконані.

ПЕРЕЛІК ПОСИЛАНЬ

1. Эрик Спирли. Корпоративные хранилища данных. Планирование, разработка и реализация. Т.1. М.: Вильямс, 2001. – 400 стр.

2. Корпоративные хранилища данных. Интеграция систем. Проектная документация. URL: https://www.prj-exp.ru/dwh/what_is_dwh.php (Дата звернення 25.11.2020)

3. Архипенко С., Голубев Д., Хранилища данных. От концепции до внедрения. М: Диалог-Мифи, 2002. – 528с.

4. Пять главных тенденций облачной интеграции данных в 2016 году. URL: https://zoom.cnews.ru/publication/item/55766 (Дата звернення 21.11.2020)

5. Сергей Глазунов. Бизнес в облаках. Чем полезны облачные технологии для предпринимателя. 22 февраля 2013. URL: https://kontur.ru/articles/225 (Дата звернення 20.11.2020)

6. Облачные вычисления URL: https://www.refinitiv.ru/ru/cloud-computing (Дата звернення 20.11.2020)

7. Хранилище в облаке. https://www.xelent.ru/blog/khranilishche-v-oblake/ (Дата звернення 20.11.2020)

8. Куда делись мои данные? URL: https://sisu.ut.ee/minuandmed/ru (Дата звернення 01.12.2020)

9. Озеро данных и хранилище данных – в чем разница? URL: https://www.sas.com/ru_ua/insights/articles/data-management/data-lake-and-data-warehouse-know-the-difference.html (Дата звернення 01.12.2020)

10. Создание физической модели базы данных: проектирование производительности. Лекции НОУ ИНТУИТ URL: https://intuit.ru/studies/courses/599/455/lecture/10182 (Дата звернення 15.11.2020)

11. Физическая модель хранилища данных: учет влияния транзакций, денормализация таблиц. Лекции НОУ ИНТУИТ URL:

https://intuit.ru/studies/courses/599/455/lecture/10180 (Дата звернення 15.11.2020)

12. Хранимые процедуры. URL: https://metanit.com/sql/sqlserver/11.1.php (Дата звернення 08.12.2019)

13.CREATE PROCEDURE (Transact-SQL). Microsoft. Документация по SQL. URL: https://docs.microsoft.com/ru-ru/sql/t-sql/statements/create-procedure-transact-sql?view=sql-server-ver15 (Дата звернення 08.12.2020)

ДОДАТОК А. ЛОГІЧНА СХЕМА СХОВИЩА ДАНИХ

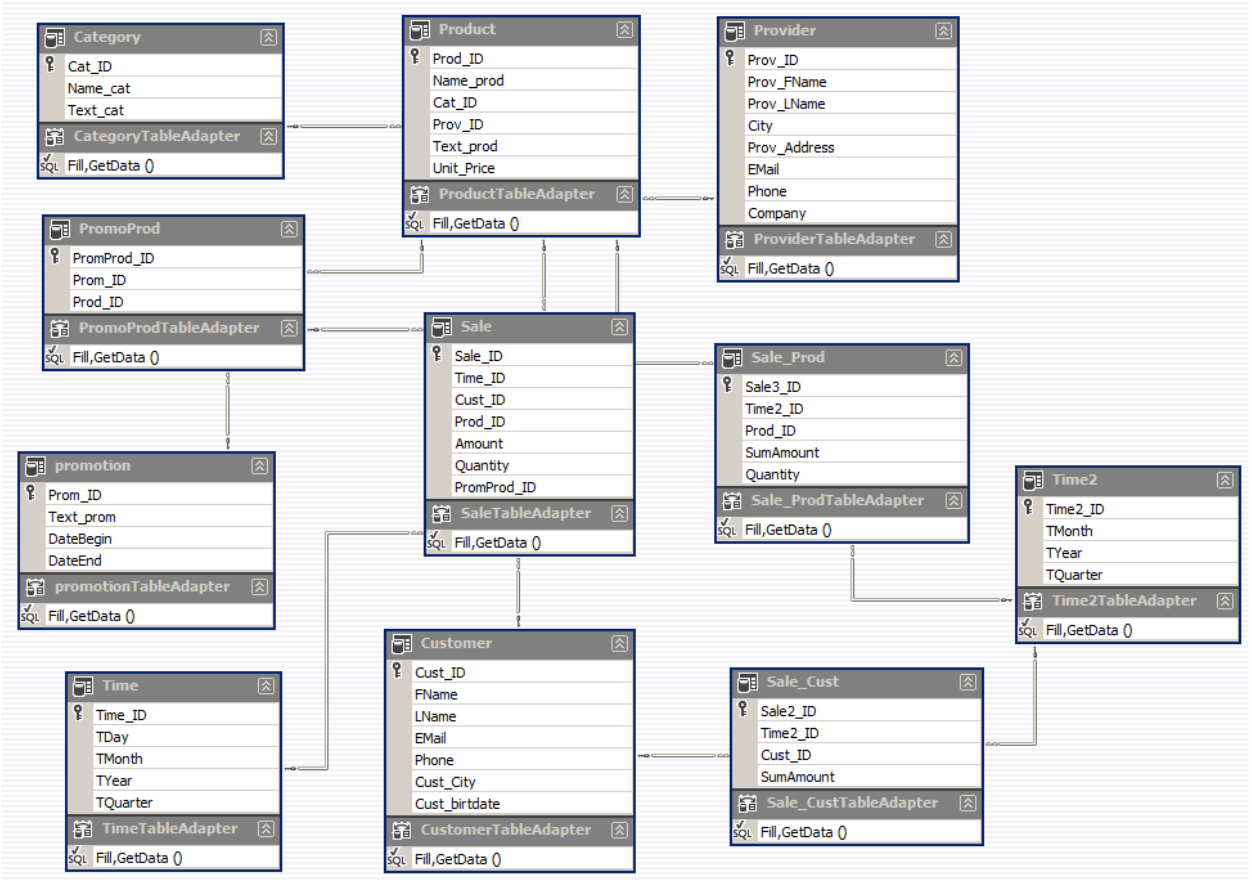

Рисунок А – Уточнена логічна схема сховища даних

Додаток Б. ФІЗИЧНА СХЕМА БАЗИ ДАНИХ

```
create table Customer (
   Cust_ID int identity,
   FName        varchar(20)       not  null,
   LName        varchar(20)      not   null,
  EMail                  varchar(40)  not null unique,
  Phone       varchar(13)     not null unique,
   Cust_City         varchar(20)   not null,
   Cust_birtdate datetime null,
   constraint PK_CUSTOMER primary key  (Cust_ID)
)
GO
Cat_ID smallint identity,
Name_cat varchar(50) not null UNIQUE NONCLUSTERED,
 Text_cat  varchar(300)      not   null,
constraint PK_CATEGORY primary key  (Cat_ID)
)
GO
create table Provider (
   Prov_ID  smallint identity,
   Prov_FName           varchar(20)          not null,
   Prov_LName           varchar(20)          not null,
   City                 varchar(20)          not null,
  Prov_Address         varchar(40)          null,
  EMail                 varchar(40) not null unique,
  Phone       varchar(13)  not null unique,
  Company              varchar(40)      not    null,
   constraint PK_PROVIDER primary key  (Prov_ID)
)
GO

create table Product (
   Prod_ID int identity,
  Name_prod varchar(100) not null UNIQUE NONCLUSTERED,
   Cat_ID smallint REFERENCES Category(Cat_ID),
  Prov_ID  smallint REFERENCES Provider(Prov_ID),
  Text_prod  varchar(300)      not   null,
  Unit_Price           numeric(8,2)         not null,
   constraint PK_PRODUCT primary key  (Prod_ID)
)
GO
```

```
CREATE TABLE Time2 (
   Time2_ID       smallint identity,
   TMonth              tinyint              not null,
   TYear                 smallint              not null,
   TQuarter              tinyint              not null,
   constraint PK_TIME2 primary key  (Time2_ID)
)
GO
CREATE TABLE Sale (
   Sale_ID   int identity,
   Time_ID              smallint   REFERENCES Time(Time_ID),
   Cust_ID              int REFERENCES Customer(Cust_ID),
   Prod_ID              int REFERENCES Product(Prod_ID),
   Amount               numeric(9,2)         not null,
   Quantity             integer              not null,
   PromProd_ID int REFERENCES PromoProd (PromProd_ID),
   constraint PK_SALE primary key  (Sale_ID)
)
GO

CREATE TABLE Sale_Cust (
   Time2_ID              smallint   REFERENCES Time2(Time2_ID),
   Cust_ID              int REFERENCES Customer(Cust_ID),
   SumAmount            numeric(9,2)         not null,
unique (   Time2_ID,  Cust_ID)
)
GO

CREATE TABLE Sale_Prod (
   Time2_ID              smallint   REFERENCES Time2(Time2_ID),
   Prod_ID              int REFERENCES Product(Prod_ID),
   SumAmount               numeric(9,2)         not null,
   Quantity             integer              not null,
unique (   Time2_ID,  Prod_ID)
)
GO

create table promotion(
Prom_ID smallint identity primary key,
Text_prom varchar(200),
DateBegin datetime not null,
DateEnd  datetime not null,
)
```

```
GO

create table PromoProd(
PromProd_ID int identity primary key,
Prom_ID smallint  references promotion,
Prod_ID int references Product,
unique(Prom_ID, Prod_ID )
)
GO
```

ДОДАТОК В. ВИХІДНИЙ КОД ЗБЕРЕЖЕНИХ ПРОЦЕДУР

```
CREATE PROC DivideCustomer
AS
BEGIN

INSERT INTO CustomerB
SELECT   FName, LName, EMail, Phone,  Cust_City, Birthdate
FROM Customer
WHERE Birthdate IS NOT NULL AND Education IS NULL
DELETE Customer
WHERE Birthdate IS NOT NULL AND Education IS NULL

INSERT INTO CustomerE
SELECT   FName, LName, EMail, Phone,  Cust_City, Education
FROM Customer
WHERE Education IS NOT NULL AND Birthdate IS NULL
DELETE Customer
WHERE Education IS NOT NULL AND Birthdate IS NULL

INSERT INTO CustomerBE
SELECT   FName, LName, EMail, Phone,  Cust_City, Birthdate, Education
FROM Customer
WHERE Birthdate IS NOT NULL AND Education IS NOT NULL
DELETE Customer
WHERE Birthdate IS NOT NULL AND Education IS NOT NULL

INSERT INTO CustomerNo_BE
SELECT   FName, LName, EMail, Phone,  Cust_City
FROM Customer
WHERE Education IS NULL AND Birthdate IS NULL
DELETE Customer
WHERE Education IS NULL AND Birthdate IS NULL
END
GO
```