УДК 004.658: 652.3

ФІСУН М.Т., ДВОРЕЦЬКИЙ М.Л., *Миколаївський державний гуманітарний університет ім. П. Могили*

Фісун Микола Тихонович – *д.т.н., проф., зав. кафедри інтелектуальних інформаційних систем МДГУ ім. П. Могили. Коло наукових інтересів – бази даних, автоматизовані системи управління.*

Дворецький Михайло Леонідович – *аспірант Миколаївського державного гуманітарного університету ім. П. Могили, Україна. Коло наукових інтересів: сучасні інформаційні технології, СППР, бази даних та бази знань, OLAP та Data Mining технології.*

СИНХРОНІЗАЦІЯ ОНОВЛЕННЯ ДАНИХ В ГЕТЕРОГЕННИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ

В статті розглянуто різні підходи для синхронізації інформації у гетерогенних інформаційних системах на прикладі супермаркету та запропоновано алгоритм обчислення точки реплікації, заснований на аналізі інформації щодо кількості та якості змін інформації, частоти звертання до даних, дати та часу попередньої реплікації та завантаженості сервера баз даних поточними запитами.

In the article different approaches are considered for synchronization of information in the geterogennikh informative systems on the example of supermarket and the algorithm of calculation of point of replikacii, based on the analysis of information on an amount and quality of changes of information, frequency of address to information, is offered, to give and to time of previous replikacii and work-load of server of bases given by current queries.

Загальна постановка проблеми та її зв'язок з науково-практичними задачами.

Розглянемо основні задачі, які повинна вирішувати система управління торгівельного підприємства.

1. Ведення складського обліку підприємства. Це у спрощеному вигляді довідники товарів, клієнтів, складів, документообіг на рівні прибуткових-видаткових накладних та звіти на рівні залишків товару на складах.
2. Для цієї системи основними вимогами є безперебійна робота та надійність серверної частини.
3. Ведення бухгалтерського обліку. Це документообіг підприємства у вигляді бухгалтерських проводок, кадровий облік, модуль обчислення заробітної плати та набір регламентованих звітів, що необхідні для подання у державні контролюючі органи. Цей модуль не передбачає велику кількість клієнтських місць, одже навіть не обов'язково має бути клієнт-серверним. Головна вимога до програмного забезпечення – своєчасне оновлення схем проводок та регламентованих звітів при змінах у законодавстві або вимогах державних контролюючих органів.

4. Модуль роздрібної торгівлі. Являє собою касове місце. Основні вимоги – безперебійна робота, незалежно від ймовірних проблем на серверній частині, мережі та таке інше, надвисока швидкодія, простота і зручність використання.
5. Кіоски даних. Підтримка прийняття рішення управлінців підприємством серенньої та вищої ланки. Працює на базі сховища даних. Основна вимога – надпотужна серверна частина.

На практиці для вирішення кожної з вищенаведених задач використовуються окремі програмні продукти, які максимально задовольняють основні вимоги кожної з них.

Таким чином вощенаведена інформаційна система є гетерогенною, у зв'язку з чим виникає проблема синхронізації даних та синтезу цих різних систем управління.

Існують різні методи переносу та синхронізації даних [1]:

- Служби трансформації даних
- Програмний інтерфейс массивного копіювання
- Реплікація даних
- Розподілені запити

Реплікація – це сукупність механізмів, що забеспечують відображення змін данних, здійснених на одному сервері, на інші сервера [2]. Реплікація може бути повною, за період або вибірковою. Вона може бути здійснена одноразово, за вимогою, або здійснюватись періодично (наприклад один раз на добу). Наприклад, на початку місяця копіюються дані про товари на складах, ціни та список можливих покупців, а наприкінці дані о проданих товарах [3]. Крім того реплікація може бути примусовою або за запитом користувача. Примусова реплікація – в цьому випадку клієнт відіграє пасивну роль. Він лише чекає, доки буде скопійовано нову порцію даних. Вся робота по оновленню лягає на сервер, який сам має встановлювати з'єднання і виконувати всі необхідні операції. Реплікація по запиту – вся робота по оновленню лягає на клієнта, який під'єднується до сервера і копіює із нього порцію інформації [2].

Але кожен із цих методів має ряд своїх недоліків. Так, наприклад, повна реплікація неприпустима для великих об'ємів баз даних, а реплікація за період не може забезпечити коректність даних при так званому редагуванні «заднім числом».

Тому антуальною є задача розробки та впровадження комбінованого методу синхронізації даних, що максимально задовільняв би вимогам до актуальності даних та мав би якомога кращі часові характеристики.

Огляд публікацій та аналіз невирішених питань. Необхідність обміну інформацією між різними системами керування баз даних не є новою проблемою і існує із часів появи та розвитку розподілених баз даних. Тому існує багато готових рішень, що дозволяють реалізувати реплікацію даних згідно одного із описаних вище методів із різноманітних джерел даних [4,5,6]. Велика кількість утіліт мають вбудовану мову запитів та багато із них мають „агентів", що дозволяють реалізувати періодичну реплікацію даних.

Однак для кожного конкретного випадку актуальним є питання вибору методу реплікації, що максимально забовільняв би всім вимогам щодо оперативності оновлення та апаратних ресурсів системи.

Крім цього, готові рішення не дозволяють динамічно змінювати період реплікації даних. Тому задача визначення найбільш зручного моменту для релікації даних та розробка інструментарію із динамічно змінною датою реплікації є актуальною.

Метою досліджень є досягнення якомога вищіх показників актуальності даних, не допускаючи при цьому перевантаження серверних ресурсів. Для досягнення мети досліджень необхідним постає розв'язання наступних задач:

- вибір оптимального методу реплікації даних у гетерогенній інформаційно-аналітичній системі для кожної із окремих ланок;

- розробка та впровадження алгоритму обчислення дати наступної реплікації, зоснований на аналізі інформації щодо кількісті та якості змін інформації, частоти звернення до даних, дати та часу попередньої реплікації та завантаженості серверу баз даних поточними запитами

Результати досліджень. У супермаркеті, що розглядається у якості прикладу, використовується програма складського обліку Квінт 5, система бухгалтерського обліку 1С 7.7, модуль роздрібної торгівлі EuroPos та сховище даних на базі середовища баз даних MS SQL Server 8.0 (рис. 1).

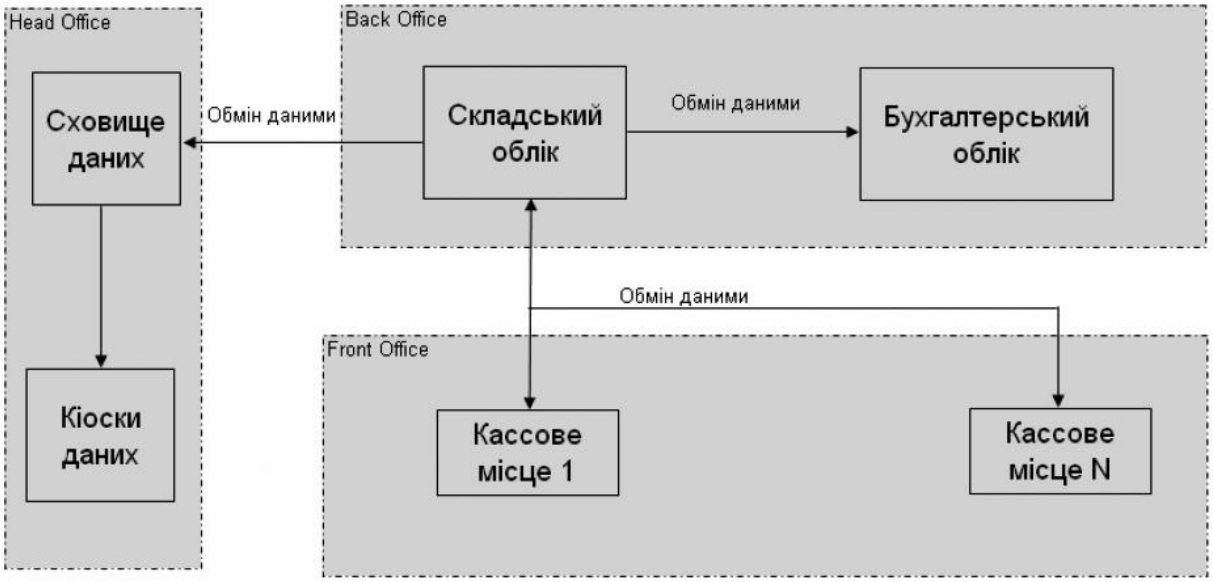

Рис 1. Взаємодія систем управління у супермаркеті

При реплікації даних між різними системами запропоновано використовувати різні методи реплікації, кожен з яких більш оптимальний для окремої ланки.

Так при реплікації даних між складським обліком та бухгалтерським обліком та сховищем даних використовується періодична реплікація із підтримкою реплікації за вимогою із використанням журналу транзакцій документів. Тобто зрепліковано буде не всі дані, і не дані по документообігу за конкретний період, а лише ті дані, що були реально змінені. Крім того, враховуються не всі зміни, а лише ті, що впливають на кінцевий результат. Так наприклад, якщо проводка у видатковій накладній не підтримує аналітику по товару та торговому агенту, через якого було здійснене відвантаження, то при зміні цих реквізитів, якщо це не спричинило зміну загальної суми по накладній, вона не буде розглядатися як змінена.

Алгоритм реплікації даних складської програми із касовим місцем є схожим, відмінність лише у тому, що за вибором можуть бути зрепліковані як усі потрібні довідники цілком (може бути доцільним при відкритті нового касового місця, або при збоях у роботі апаратного чи програмного забеспечення клієнтської програми), так і лише конкретні зміни у цих довідниках. Крім того, внаслідок того, що кожна каса працює незалежно від серверної частини та одна від одної, підтримуються так звані точки актуальності інформації для кожної каси, де зберігаються значення дати та часу останньої реплікації даних із складського обліку.

Завантаження даних із касового місця у складську програму є значно простішим, оскільки це не потребує постійної синхронізації та проводиться зазвичай людиною-користувачем касового місця по завершенню робочої доби.

Як вже зазначалося вище, обмін даними проводиться періодично із підтримкою режиму реплікації за вимогою. Але головна особливість запропонованого алгоритму реплікації полягає у тому, що період обміну не є чітко визначеним. Це обумовлено тим, що у різних сетуаціях необхідна різна актуальність даних [7]. Так, наприклад, інформація по залишкам товару на кожну годину не є актуальною для менеджера у звичайний день, але є актуальною для „ходових" позицій товару напередодні свята. Або ж, наприклад, довідник товарів може бути незмінним (або зміни можуть торкнутися неходових позицій товару) на протязі доби, а потім за лічені хвилини може бути змінено велику кількість позицій.

Саме тому корентне обчислення наступної дати реплікації даних може з одного боку значно підвищити актуальність даних, а з іншого істотно розвантажити ресурси серверів баз даних. Виходячи із цього було запропоновано використовувати спеціально розроблений алгоритм, що враховує такі фактори, як:

- кількість змін у складській програмі;
- якість цих змін (наскільки вони вплинуть на роботу кінцевого замовника інформації);
- частоту звернiння до цих даних;
- дату та час попередньої реплікації;
- завантаженість серверу баз даних поточними запитами.

Схематично алгоритм обчислення точки наступної реплікації зображено на рис. 2.

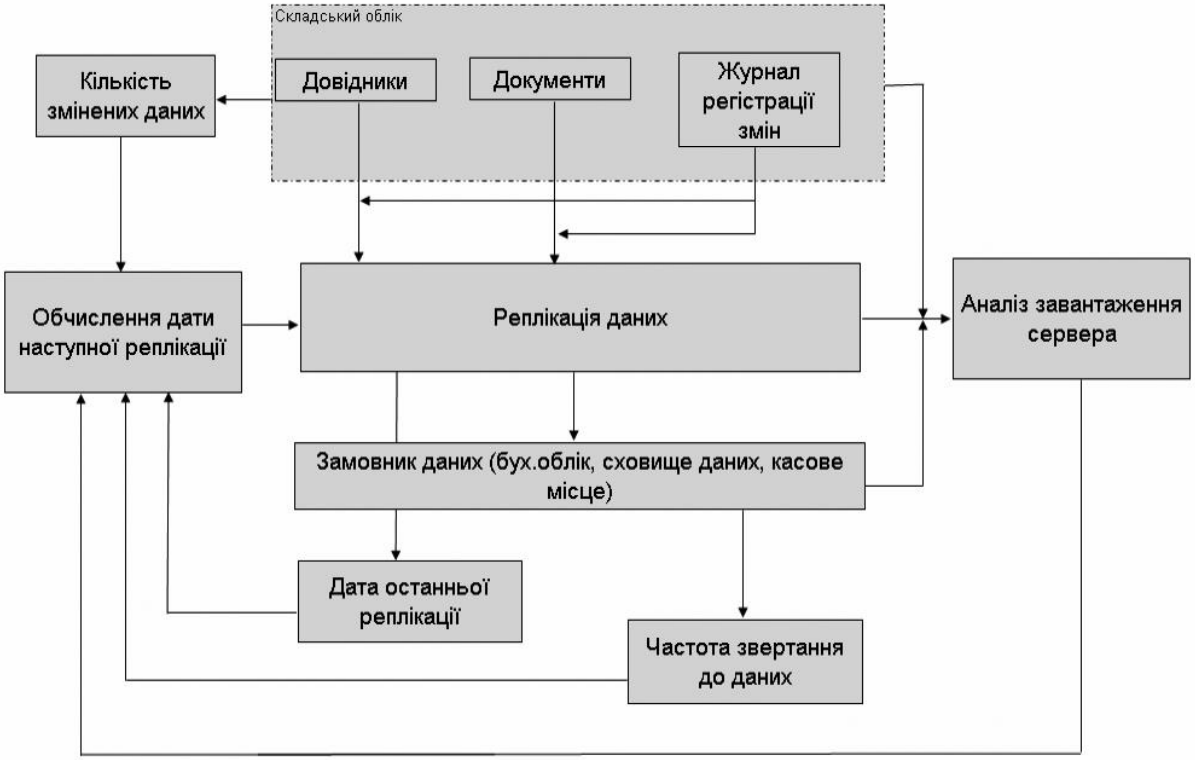

Рис. 2. Схема алгоритму реплікації даних у супермаркеті.

Далі детальніше розглянемо алгоритм обчислення моменту наступної реплікації між складським обліком та касовим місцем. Для визначення того, потрібна реплікація чи ні, обчислюється коефіцієнт антуальності даних, далі $К_{акт}$. Якщо $К_{акт}$ перевищує гранично припустиме значення, далі $К_{гран}$, то відбувається реплікація.

$$К_{акт} – К_{гран}>0 \quad (1)$$

Коефіцієнт актуальності даних обчислюємо за формулою

$$К_{акт}= К_{c1} * (a_{c1,1} * К_{c1,1}+ a_{c1,2} * К_{c1,2}+...+ a_{c1,n1} * К_{c1,n1}) + К_{c2} * (a_{c2,1} * К_{c2,1}+ a_{c2,2} * К_{c2,2}+...+ a_{c2,n2} * К_{c2,n2}) + К_{cm} * (a_{cm,1} * К_{cm,1}+ a_{cm,2} * К_{cm,2}+...+ a_{cm,nm} * К_{cm,nm}) \quad (2)$$

де $К_{ci}$ – ваговий коефіцієнт і-го довідника, що має бути зреплікованой, $К_{ci,j}$ – ваговий коефіцієнти j-ї позицій і-го довідника, $a_{ci,j}$ – флаг, що вказує на факт змін j-ї позиції в і-му довіднику {0,1}

$К_{ci}$ – встановлюється примусово, виходячи із значущості довідника для касового місця (так, наприклад, найзначущим є довідник товарів, а зміни у довіднику складів не вплинуть на поточну роботу касси до завершення робочої доби)

$К_{ci,j}$ – розраховується програмно виходячи із характеристик позиції довіднику (наприклад, товар може бути «ходовий» – той, по якому постійно відбуваються відвантаження, або «мертвий» - такий, що давно не продається)

$a_{ci,j}$ – встановлюється програмно при зміні реквізитів у базі даних.

Гранично припустиме значення залежить від дати останнього оновлення даних та від поточного завантаження сервера, та обчислюється за формулою

$$К_{гран} = F(\Delta T) * (К_{період} / \Delta T + К_{завант}) \quad (3)$$

де $К_{період}$ – коефіцієнт періодичності оновлення даних, ΔT – інтервал, протягом якого дані не оновлювалися, $К_{завант}$ – коефіцієнт завантаження сервера, $F(\Delta T)$ – функція допустимості

$К_{період}$ – встановлюється примусово із можливістю ручного редагування (є більш актуальним для реплікації сховища даних. Можливе зменшення періоду при необхідності у оперативних данних напередодні свят).

$К_{завант}$ – обчислюється за допомогою спеціальних моніторингових програм (наприклад MS SQL Profiler, Performance monitor).

$F(\Delta T)$ – приймає значення 0, якщо значення ΔT перевищує гранично допустимий інтервал, інакше 1.

На рис. 3 наведено схему обміну даними між підсистемою імпорту-експорту даних та різнорідними СКБД із огляду на представлення у локальній мережі та використання даних кінцевими користувачами. Підсистема імпорту-експорту даних реалізована на базі MS SQL Server 2000 із використанням Data Transformation Services. За обчислення точки реплікації відповідає окремий спеціально розроблемий модуль, що має бути завантажений на одному із комп'ютерів локальної мережі у якості сервісу.

Висновки та перспективи подальших досліджень. Запропонований алгоритм синтезу систем управління у супермаркеті із використанням відстеження змін даних та визначенням суттєвості цих змін, та із динамічно змінюваним періодом оновлення даних дозволяє значно підвищити актуальність та точність звітних даних, при цьому не допускаючи нераціонального використання серверних ресурсів та запобігаючи перевантаженню сервера. Введення в експлуатацію нової підсистеми імпорту-експорту даних дозволило на 30% підвищити актуальність даних приймача у порівнянні із використовуваним раніше алгоритмом примусової реплікації за період.

Перспективи подальшого розвитку та вдосконалення системи вбачаються у розробці модуля навчання шляхом додавання зворотнього зв'язку, що динамічно корегуватиме вагові коефіцієнти довідників, конкрентих записів та граничних значень часу оновлення виходячи із даних щодо своєчасності оновлення інформації.

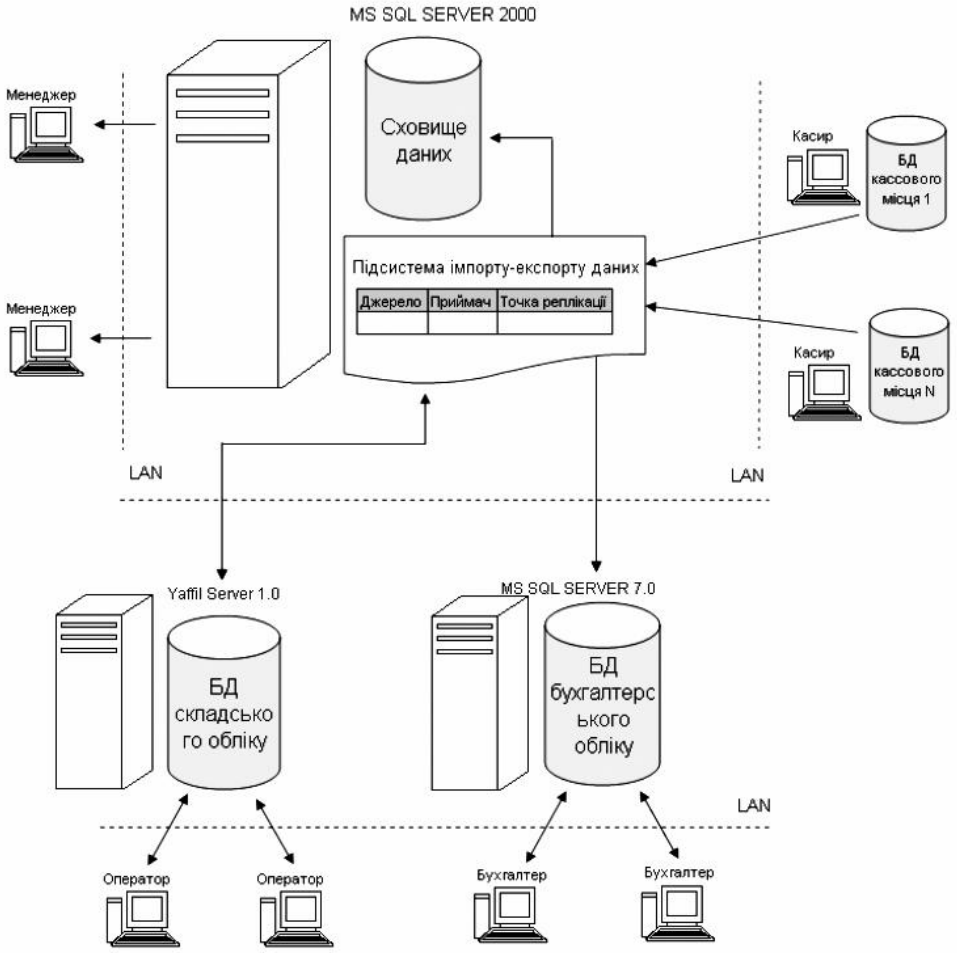

Рис. 3. Схема обміну даними.

Література

1. Мамаев Е.В. SQL Server 2000. – СПб.: БХВ-Питербург, 2001. – 1280 с.
2. Гэри Хансен, Джэймс Хансен. Базы данных: разработка и управление: Пер. С англ. – М.: ЗАО “Издательство БИНОМ”, 1999. – 704 с.
3. Оутей М., Конте П. Эффективная работа: SQL Server 2000. – СПб.: Питер; К.: Издательская группа BHV, 2002. – 992 с.
4. Дейт К. Дж. Введение в системы баз данных. – Киев*Москва: Диалектика, 1998. – 787 с.
5. Гарсиа-Молина Г., Ульман Дж.Д., Уидом Дж. Системы баз данных. Полный курс. – М.: Изд. дом “Вильямс”, 2003. – 1088 с.
6. Конноли Т., Бегг К., Страчан А. Базы данных: проектирование, реализация и сопровождение. Теория и практика, 2-е издание. – “Вильямс”: Москва*Санкт-Петербург*Киев, 2000. – 1112 с.
7. Гайдышев И. Анализ и обработка данных: специальный справочник – СПб: Питер, 2001. – 752 с.