УДК 681.3.06

Кулябко О. П., Кулябко П. П.

ПРО ПЕРЕТВОРЕННЯ ЗАПИТІВ У МОВІ DLI (IMS) НА SQL-ЗАПИТИ

Мова DLI (СУБД IMS) має суттєво нижчий рівень порівняно з мовою SQL, але для багатьох технологій при міграції кожному оператору цієї мови ставиться у відповідність точно один оператор мови SQL. Разом з тим досить часто один SQL-оператор може замінити (тобто є функціонально еквівалентним) цілу низку DLI-операторів. Така заміна може суттєво оптимізувати код порівняно із заміною один на один.

СУБД IMS все ще досить широко використовується в наш час, хоча була запропонована IBM близько 40 років тому [1]. Враховуючи ієрархічну нереляційну структуру бази даних, для IMS стає дедалі важче інтегруватися з сучасними технологіями та новими мовами. Наприклад, технологія реляційної СУБД Oracle (або DB2) є цікавою альтернативою для користувачів IMS, навіть IBM рекомендує перехід до технологій реляційних баз даних як спосіб досягнення більш гнучких систем, хоча з можливою втратою ефективності.

Складність міграції програм з використанням IMS на сучасні платформи змусила деякі організації відкласти ці перетворення на пізніший час, але аргументи на їх користь невідпорні:

- складність інтеграції даних, що надходять з бази даних IMS;
- особливо важко здійснюється доступ до IMS даних із сучасних web-орієнтованих мов;
- наявні ресурси IMS є недостатніми і можуть бути дуже дорогими;
- IMS не підтримує багато з тих можливостей, які пропонують нові технології;
- використання нових технологій може суттєво поліпшити продуктивність розробників програм.

Однією з найбільш актуальних задач сьогодення є задача міграції програмного забезпечення із застарілих програмних платформ у більш сучасне програмне середовище.

Оскільки обсяги великі, то процес міграції бажано автоматизувати.

Мова DLI має суттєво нижчий рівень порівняно з мовою SQL, але для багатьох технологій [2-9] при міграції кожному оператору EXEC DLI (або ж виклик процедури взаємодії з СУБД IMS) ставиться у відповідність точно один оператор EXEC SQL. Разом з тим досить часто один SQL-оператор може замінити (тобто є функціонально еквівалентним) цілу низку DLI-операторів. Така заміна може суттєво оптимізувати код порівняно із заміною один на один, оскільки значна частина функціональності програми (написаної, наприклад, на мові JAVA) перекладається на плечі реляційної СУБД.

Розглянемо кілька прикладів виконання такої заміни.

Нехай у СУБД IMS задано таку структуру даних (DBD) для бази даних ЛІКУВАННЯ.

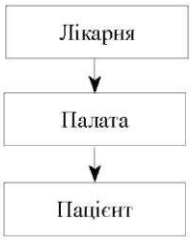

Рис. 1. Структура бази даних

Назва сегмента	Список полів
Лікарня	(Номер, Назва, Адреса)
Палата	(Номер, Тип, Кільк-місць)
Пацієнт	(Номер, Прізвище, Температура, Діагноз)

Відзначимо, що в наведеному описі пропущено низку параметрів, які не потрібні для нашої задачі. Крім того, назви сегментів та полів наведені кирилицею без обмеження довжини назви. Назви ключових полів підкреслені.

Будемо також вважати, що для прикладних програм задано підсхему (PSB), яка складається з одного PCB. До складу PCB1 входять сегменти Лікарня, Палата, Пацієнт. Окремо структура PCB не наводиться, оскільки вона збігається зі структурою бази даних.

Наведемо структуру таблиць реляційної бази даних, яка відповідає вищенаведеній ієрархічній структурі.

Назва таблиці	Список полів
Лікарня	(Номер, Назва, Адреса)
Палата	(Номер, *Номер-Лікарні*, Тип, Кільк-місць)
Пацієнт	(Номер, *Номер-Палати*, *Номер-Лікаря*, Прізвище, Температура, Діагноз)
Лікар	(Номер, Прізвище, Спеціальність)

© *Кулябко О. П., Кулябко П. П.,* 2008

Як і в попередньому випадку, назви ключових полів підкреслені, а «чужі ключі» подано курсивом.

Розглянемо кілька прикладів запитів як у термінах DLI, так і SQL.

Запит 1. Знайти пацієнта № 5 з палати № 6 з лікарні № 1.

```
EXEC DLI GU USING PCB(1)
        SEGMENT (Лікарня)
        WHERE (Номер=1)
        SEGMENT(Палата)
        WHERE(Номер=6)
        SEGMENT(Пацієнт)
        WHERE(Номер=5)
        INTO(ioarea);
```

```
EXEC SQL SELECT * FROM Лікарня INNER JOIN (Палата INNER JOIN
Пацієнт ON Палата.Номер = Пацієнт.Номер-Палати)
 ON Лікарня.Номер = Палата.Номер-Лікарні
WHERE Лікарня.Номер=1 AND Палата.Номер=6 AND Пацієнт.Номер=5
```

У наведеному запиті, оскільки точно вказані значення ключових полів по кожному сегменту бази даних, то один SQL-оператор замінює один DLI-оператор.

Запит 2. Знайти прізвища пацієнтів палати № 6 з лікарні № 1 з температурою вище 37°.

```
EXEC DLI GU USING PCB(1)
   SEGMENT(Лікарня)
   WHERE(Номер=1)
   SEGMENT(Палата)
        WHERE(Номер=6)
   INTO(ioarea);
    {інші оператори}
WHILE {умова}DO
    {інші оператори}
  EXEC DLI GNP USING PCB(1)
     SEGMENT(Пацієнт)
     WHERE(Температура>37)
     INTO(ioarea);
   {інші оператори}
END;
```

```
EXEC SQL SELECT Пацієнт.Прізвище
 FROM Лікарня INNER JOIN
(Палата INNER JOIN Пацієнт ON Палата.Номер=Пацієнт.Номер-Палати)
 ON Лікарня.Номер = Палата.Номер-Лікарні
WHERE Лікарня.Номер=1 AND Палата.Номер=6 AND
Пацієнт.Температура > 37
```

Прокоментуємо реалізацію запиту у термінах DLI (IMS). Він складається з першого EXEC DLI оператора з функцією GU, який знаходить потрібний примірник сегмента Палата і фіксує на ньому поточний стан пошукового процесу. Наступний EXEC DLI-оператор з функцією GNP розташований всередині циклу базової мови (яка саме базова мова використовується, для нашого розгляду суттєвого значення не має). Функція GNP діє так, що кожного разу запам'ятовується поточний стан пошуку на знайденому примірнику сегмента Пацієнт і на наступному кроці циклу починає діяти від знайденого. Разом з тим перебір Пацієнтів обмежений зафіксованим станом пошукового процесу для примірника сегмента Палата на вищому рівні. Наведений у правій колонці таблиці SQL-оператор матиме такий само результат пошуку.

Відзначимо також, що створення потрібного SQL- оператора в автоматичному режимі вимагає застосування процедури синтаксичного аналізу не тільки для розбору EXEC DLI-оператора, але і для розбору конструкцій базової мови, зокрема циклів.

1. http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp?
2. http://www.migrationpros.com/dbre-IMStoSQL.html
3. http://www.ateras.com/Converting_IMS_Databases.aspx
4. http://findarticles.com/p/articles/mi_hb5570/is_200610/ai_n23570939
5. http://www.migrationware.com/pm/dbims.html
6. http://forums.mysql.com/read.php?63,52523,52523
7. http://www.clerity.com/resources/transition/datasheets/IMS_Datasheet.pdf
8. http://www.move2open.com/ims-to-rdbms.html
9. http://www.zjournal.com/index.cfm?section=article&aid=366

O. Kuliabko, P. Kuliabko

ABOUT TRANSFORMARON DLI (IMS)-QUERIES TO SQL-QUERIES

The programming language DLI (DBMS IMS) has significantly lower level than SQL-language, but many technologies consider that in migration process each statement of this language is changed with exactly one SQL-statement. However sometimes one SQL-statement can be put instead of sequence of DLI-statements. Last case may mean the sufficient optimization of the source code in comparison with the previous one.