УДК 004.75

**К.т.н., доцент Орлова М.М., магістрант Колодницький М.В.**

**Національний технічний університету України
«Київський політехнічний інститут»**

# МІНІМІЗАЦІЯ НАВАНТАЖЕННЯ НА КОМУНІКАЦІЙНУ МЕРЕЖУ ЗА ДОПОМОГОЮ ДЕКОМПОЗИЦІЇ *SQL*-ЗАПИТІВ

## Вступ

На сьогодні існує велика кількість розподілених систем обробки даних на базі архітектури "клієнт-сервер". У цих системах дані розміщуються на сервері, який отримує запити від клієнтської станції. При цьому часто виникає потреба в передачі з сервера на клієнтську станцію великого обсягу даних у відповідь на запити, тому комунікаційна мережа може стати "вузьким місцем" у системі. Проблема мінімізації обсягу даних для передачі відповідей з сервера на клієнтські станції є актуальною. Останнім часом активно розвивається новий метод декомпозиції відповідей на проміжні результати, який дозволяє зменшити обсяг відповідей. Ці проміжні результати передаються клієнтській станції і та, на їх основі, отримує результуючу відповідь.

Основними працями, в яких проведено дослідження даного питання є [3] та [5].

## Постановка задачі

Основною метою статті є дослідження проблеми мінімізації обсягу даних, які передаються як результати виконання запитів з сервера на клієнтську станцію. Для досягнення цієї мети вирішуються наступні задачі:

- аналіз виконання запитів без використання декомпозиції;
- аналіз виконання запитів з використанням декомпозиції;
- дослідження отриманих результатів.

## Виконання запитів без декомпозиції

Розглянемо декомпозицію запитів на прикладі системи виборів. Припустимо, в цій системі існують наступні таблиці:

- *PERSON(person_id(15), first_name(30), middle_name(30), last_name(50), ps_info(100))* – таблиця, яка містить загальну інформацію про депутатів;
- *PARTY(party_id(10), pname(200), date_begin(4), date_end(4), pt_info(100))* – таблиця, яка містить інформацію про партії.
- *DEPUTY(person_id(15), party_id(10), date_begin(4), date_end(4), dp_info(100))* – таблиця, яка містить інформацію про приналежність депутата до партії.

У дужках після імен таблиць містяться назви полів. Підкреслене поле є первинним ключем відповідної таблиці. Числа в дужках після імен полів означають їх розміри в байтах.

Розглянемо запит 1, який надсилається клієнтською станцією на сервер для отримання поточного списку всіх депутатів:

```
SELECT pt.pname, ps.first_name, ps.middle_name, ps.last_name
FROM person ps, party pt, deputy d
WHERE ps.person_id = d.person_id
AND pt.party_id = d.party_id
AND trunc(sysdate) BETWEEN
  nvl(pt.date_begin, to_date('01.01.1900', 'dd.mm. yyyy'))
  AND
  nvl(pt.date_end, to_date('31. 12.2999', 'dd.mm. yyyy'))
AND trunc(sysdate) BETWEEN
  nvl(ps.date_begin, to_date('01.01.1900', 'dd.mm. yyyy'))
  AND
  nvl(ps.date_end, to_date('31. 12.2999', 'dd.mm. yyyy'))
```

Сервер формує відповідь на запит 1 і надсилає її клієнтській станції. У табл. 1 показано уривок відповіді на запит 1:

Таблиця 1. Уривок відповіді на запит 1

| № зап. | pname | first_name | middle_name | last_name |
|---|---|---|---|---|
| 1 | Партія декомпозиції … [200 символів] | Іван | Іванович | Іваненко |
| 2 | Партія декомпозиції … [200 символів] | Петро | Петрович | Петренко |
| 3 | Партія декомпозиції … [200 символів] | Сидір | Сидорович | Сидоренко |
| 4 | Партія декомпозиції … [200 символів] | Микола | Миколайович | Миколаєнко |
| . . . | | | | |

Припустимо, що на сервері міститься 4000 записів, які відповідають запиту 1. Тоді кількість байтів, переданих клієнтській станції, буде дорівнювати:

4000 * (200 + 30 +30 + 50) = 1 240 000 байтів.

**Використання декомпозиції запитів**

Розглянемо можливість зменшення обсягу переданої на клієнтську станцію інформації при збереженні всіх необхідних даних для отримання відповіді на вихідний запит. Очевидно, що відповідь на запит містить надлишкові дані. Наприклад, записи 1, 2, 3 і 4 містять однакову назву партії. Зважаючи на це, можна провести декомпозицію відповіді на запит 1 на проміжні результати – подання *V1*(депутати) і *V2*(партії):

*V1:*

```
SELECT distinct pt.party_id, ps.ps.first_name, ps.middle_name,
ps.last_name
FROM person ps, party pt, deputy d
WHERE ps.person_id = d.person_id
AND pt.party_id = d.party_id
AND trunc(sysdate) BETWEEN
  nvl(pt.date_begin, to_date('01.01.1900', 'dd.mm. yyyy'))
  AND
  nvl(pt.date_end, to_date('31. 12.2999', 'dd.mm. yyyy'))
AND trunc(sysdate) BETWEEN
  nvl(ps.date_begin, to_date('01.01.1900', 'dd.mm. yyyy'))
  AND
  nvl(ps.date_end, to_date('31. 12.2999', 'dd.mm. yyyy'))
```

V2:

```
SELECT distinct, pt.party_id, pt.pname
FROM person ps, party pt, deputy d
WHERE ps.person_id = d.person_id
AND pt.party_id = d.party_id
AND trunc(sysdate) BETWEEN
  nvl(pt.date_begin, to_date('01.01.1900', 'dd.mm. yyyy'))
  AND
  nvl(pt.date_end, to_date('31. 12.2999', 'dd.mm. yyyy'))
AND trunc(sysdate) BETWEEN
  nvl(ps.date_begin, to_date('01.01.1900', 'dd.mm. yyyy'))
  AND
  nvl(ps.date_end, to_date('31. 12.2999', 'dd.mm. yyyy'))
```

У таблицях 2 та 3 показано уривки відповідей на запит *V1* та *V2* відповідно.

Таблиця 2. Уривок відповіді на запит *V1*

| № зап. | party_id | first_name | middle_name | last_name |
|---|---|---|---|---|
| 1 | 1 | Іван | Іванович | Іваненко |
| 2 | 1 | Петро | Петрович | Петренко |
| 3 | 1 | Сидір | Сидорович | Сидоренко |
| 4 | 1 | Микола | Миколайович | Миколаєнко |
| . . . | | | | |

Таблиця 3. Уривок відповіді на запит *V2*

| № зап. | party_id | Pname |
|---|---|---|
| 1 | 1 | Партія декомпозиції… [200 символів] |
| 2 | 1 | Партія реляційних … [200 символів] |
| . . . | | |

Використовуючи відповіді на подання *V1* і *V2* клієнтська станція може сформувати результуючу відповідь. Припустимо, що відповіддю на запит *V1* будуть 4000 записів, а на запит *V2* – 40 записів. Використовуючи розміри полів таблиць можна підрахувати, що розмір відповіді на основі цих подань буде становити:

4000 * (10 + 30 + 30 + 50) + 40 * (10 + 200) = 488 400 байтів,

у той час як розмір відповіді на вихідний запит становить 1 240 000 байтів.

З прикладу видно, що за допомогою декомпозиції запитів можна значно зменшити обсяг відповіді на запит, передаючи клієнтській станції результати подань.

## Висновок

У статті розглянуто задачу передачі відповідей на запити з сервера на клієнтську станцію. Сутність питання полягає в надлишковості передавання даних. Дослідження показали, що для мінімізації обсягу даних, які передаються як результати запитів, необхідно застосовувати декомпозицію запитів на проміжні подання, що значно зменшує обсяг даних для передачі і час відповіді на запит. Даний метод є гнучким і високоефективним, що підтверджується експериментально.

**Література**


1. *R. Chirkova, A. Y. Levy, D. Suciu.* A Formal Perspective on the View Selection Problem // Proceedings of the 27th International Conference on Very Large Data Bases / Rada Chirkova, Alon Y. Levy, Dan Suciu. – San Francisco, 2001. – P. 59-68.
2. *R. Chirkova, C. Li, J. Li.* Answering queries using materializing views with minimal size / Rada Chirkova, Chen Li, Jia Li // The VLDB Journal – The International Journal on Very Large Data Bases, 2006. – №3. – P. 191-210.
3. *R. Chirkova, C. Li, J. Li.* Minimizing Data-Communication Costs by Decomposing Query Results in Client-Server Environments (Extended Version) / Rada Chirkova, Chen Li, Jia Li. – Irvin : UCI ICS, 2003. – P. 18.
4. *R. Pottinger, A. Y. Levy.* A Scalable Algorithm for Answering Queries Using Views // Proceedings of the 26th International Conference on Very Large Data Bases / Rachel Pottinger, Alon Y. Levy. – San Francisco, 2000. – P. 484-495.
5. *Э. Таненбаум, М. ван Стеен*. Распределенные системы. Принципы и парадигмы / Эндрю Таненбаум, Маартен ван Стеен. — СПб.: Питер. – 2003. — 877 с.