Самолюк І.І. , ст. викладач,
Житомирський державний технологічний університет

АВТОМАТИЗАЦІЯ АНАЛІЗУ ПОБУДОВАНИХ SQL-ЗАПИТІВ З МЕТОЮ ОПТИМІЗАЦІЇ ШВИДКОДІЇ

Бази даних є сердцем практично кожної інформаційної системи і використання даної системи напряму залежить від швидкодії обмінних операцій. Існує багато випадків, коли результат вибірки отримується за допомогою різних запитів до БД, які будуть потребувати різного часу та ресурсів сервера на обробку.

Вибір того чи іншого типу SQL-запиту, особливо якщо це критично для роботи інформаційної системи потребує певних досліджень при розробці системи і, можливо, залучення досвідченого експерта. В цілому розробка та оптимізація SQL-запитів потребує досвіду, накопичення і постійного доповнення значної бази знань та методів оптимізації. Спростити та безпосередньо автоматизувати таку роботу можна розробивши певну експертну систему, яка з певним часом та за умов ефективного накопичення надаватиме великих переваг порівняно з існуючими системами оптимізації запитів СКБД.

Для демонстрації ефекту, який можна досягти вибором ефективного SQL-запиту - порівняємо два SQL-запити які дають однаковий набір даних. Це порівняння також дає можливість окреслити ряд параметрів від яких може залежати продуктивність та швидкодія SQL-запитів і які необхідно враховувати при виборі того чи іншого варіанту. Скористаємось двома SQL-запитами, що виконують групування рядків таблиці і здіснюють обчислення даних таблиці з використанням агрегатних функцій COUNT, SUM, AVG, MIN, MAX. Один з запитів використовує віконні функції (ВФ). Другий – звичайне групування (ЗГ) з використанням тих самих агрегатних функцій.

В якості даних для порівняння продуктивності SQL-запитів використовується 4 таблиці однакової структури з різною кількість записів: приблизно 1 тисяча, 40 тисяч, 400 тисяч, 4 мільйони.

Структура таблиць досить проста, задається наступним SQL-запитом:

```
CREATE TABLE SampleData
(   ID NCHAR(10)
   ,N INT);
```

Результати порівняння двох SQL-запитів наведені в табл. 1.

SQL-запит з використанням віконних функцій:

```
SELECT ID, N
,[Rows]=COUNT(*) OVER (PARTITION BY ID)
  ,[Sum]=SUM(N) OVER (PARTITION BY ID)
  ,[Average]=AVG(N) OVER (PARTITION BY ID)
  ,[Minimum]=MIN(N) OVER (PARTITION BY ID)
  ,[Maximum]=MAX(N) OVER (PARTITION BY ID)
  FROM SampleData
```

SQL-запит з використанням звичайного групування:

```
SELECT a.ID, N
    ,[Rows],  [Sum],  [Average],
[Minimum], [Maximum]
FROM SampleData a
CROSS APPLY
(   SELECT ID
      ,[Rows]=COUNT(*)
      ,[Sum]=SUM(N)
      ,[Average]=AVG(N)
      ,[Minimum]=MIN(N)
      ,[Maximum]=MAX(N)
    FROM SampleData b
    WHERE a.ID = b.ID
    GROUP BY ID) b;
```

Таблиця 1. Порівняння продуктивності SQL-запитів

Кількість записів	Вид запиту	1k	40k	400k	4M
Час ЦП (ms)	ВФ	0	66	905	14084
	ЗГ	0	22	291	3321
Загальний час (ms)	ВФ	113	427	3136	31975
	ЗГ	108	376	2892	27582
Кількість логічних читань	ВФ	1706	87534	835008	8233349
	ЗГ	1706	346	3448	32930

Порівнявши дані з наведеної вище таблиці ми можемо переконатись у сутєвій різниці в часі обробки запитів. При чому із збільшенням кількості записів в оброблюваній таблиці час, що витрачається ЦП відрізняється в кілька разів. Також, значно відрізняється кількість сторінок, прочитаних з пам’яті. Аналіз даного прикладу показує, що при виборі того чи іншого запиту доводиться керуватися певним рядом параметрів, таких як об’єм таблиці, наявність достатньої кількості пам’яті, швидкодії дискової підсистеми, якщо пам’яті недостатньо.

Наявність проблеми вибору кращого SQL-запиту в розглянутому та інших випадках наводить на ідею розглянути можливість створення єдиної бази знань (експертної системи), за допомогою якої можна буде автоматизовано оптимізувати найбільш часто використовувані SQL-запити в залежності від певного набору параметрів на основі наявних результатів експериментів і напрацьованого досвіду.