Таким образом, опыт, накопленный картографической системой в процессе обработки запросов пользователей, может быть применен для генерации достаточно узких ограничений для вновь создаваемых запросов без существенного снижения информативности ответа. Что позволяет уменьшить количество запросов к ГИС-серверу и снизить требования к объему памяти мобильного устройства по сравнению с существующими системами, давая возможность решать на современных мобильных устройствах значительно более сложные задачи.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Гнедков С.В*. Мобильные ГИС-решения компании Autodesk // САПР и графика. – 2001. – № 3. – С. 8–12.
2. *Седжвик Р*. Фундаментальные алгоритмы на С. Части 1–5. Анализ. Структуры данных. Сортировка. Поиск. Алгоритмы на графах. – СПб.: ДиаСофт ЮП, 2003. – 1136 с.
3. *Салищев К.А*. Картография. – М.: МГУ, 1971. – 248 с.
4. *Салищев К.А*. Картоведение. М.: МГУ, 1982. – 408 с.

УДК 004.54:658.01

С.Л. Беляков, Д.А. Диденко

ОПТИМИЗАЦИЯ ЗАПРОСОВ К КАДАСТРОВОЙ БАЗЕ ДАННЫХ

В настоящее время существует множество вопросов, связанных с оптимизацией выполнения запросов в системах управления базами данных (СУБД). Обычно, говоря про оптимизацию в СУБД, имеем в виду аспект оптимизации запросов, то есть такой способ выполнения запроса, когда по его начальному представлению путем синтаксических и семантических преобразований вырабатывается процедурный план выполнения запроса, наиболее оптимальный при существующих в базе данных управляющих структурах. Экономический принцип требует, чтобы процедуры оптимизации пытались либо максимизировать пропускную способность при заданном числе ресурсов, либо минимизировать потребление ресурсов при данной пропускной способности. Оптимизация запросов направлена на минимизацию времени отклика для заданного запроса и смеси типов запросов в данной системной среде. Эта общая цель допускает ряд различных операционных целевых функций. Время отклика является разумной целью только при предположении, что время пользователя является наиболее важным критическим ресурсом. В противном случае можно стремиться к непосредственной минимизации стоимости потребления технических ресурсов. Обе цели являются в большой степени взаимно дополнительными; при этом возникающие конфликты целей обычно разрешаются путем назначения ограничений на доступные технические ресурсы (например, на размер буферного пространства в основной памяти) [1].

Практически каждый разработчик может создавать SQL-запросы, но возможностью создавать максимально оптимизированные запросы, которые отличают действительно быстро работающее приложение от его собрата, показывающего приемлемую производительность, обладает далеко не каждый [2].

Целью данной статьи является анализ наиболее оптимальных путей выполнения клиентских запросов. Анализируя и перестраивая SQL-запросы, можно снизить время их выполнения в десятки, а иногда и в сотни раз. Итак, после того, как создана структура базы данных, можно проектировать запросы, при помощи кото-

рых клиентские приложения будут манипулировать данными на сервере, осуществляя операции выборки, вставки, изменения и удаления данных.

В базе данных по учету договоров добровольного страхования автотранспорта содержатся следующие запросы, программные коды которых представлены ниже. Каждый из запросов выполнен в двух вариантах для дальнейшего проведения статистических исследований. Общая задача первого запроса такова: найти все договора страхования, заключенные на легковые автомобили (по типу ТС), с типом двигателя «дизельный». Программный код первого варианта таков:

```
SELECT * FROM Договорстрахования where ID_dogovora_straxovania in (select ID_komplektacii_TS from комплектация_тс where (Tip_dvigatelia='дизельный' and ID_komplektacii_TS in (select ID_svedenij_o_TS from сведенияотс where Tip_TS='Легковые')))
```

Программный код второго варианта таков:

```
SELECT * FROM Договорстрахования where ID_dogovora_straxovania in (select ID_komplektacii_TS from комплектация_тс where (Tip_dvigatelia='дизельный' and ID_komplektacii_TS in (SELECT ID_dogovora_straxovania FROM Договорстрахования where ID_dogovora_straxovania in (select ID_svedenij_o_TS from сведенияотс where Tip_TS='Легковые'))))
```

Общая идея следующего запроса такова: выделить договора только со страховым случаем. Программный код первого варианта таков:

```
SELECT * FROM Договорстрахования where Straxovoj_slychiaj=True
```

Программный код второго варианта следующий:

```
SELECT * FROM Договорстрахования where ID_dogovora_straxovania in (select ID_dogovora_straxovania from Договорстрахования where ID_dogovora_straxovania in (select ID_dogovora_straxovania from Договорстрахования where Straxovoj_slychiaj=True))
```

Смысл следующего запроса следующий: найти страхователя (физическое лицо) по фамилии, имени, отчеству. Текст программы таков:

```
If IsNull([фа]) Then
    query1 = "([ID_straxovatelia]=[ID_straxovatelia])"
Else
    query1 = "(Familia LIKE '" & [фа] & "')"
End If
If IsNull([им]) Then
    query2 = "([ID_straxovatelia]=[ID_straxovatelia])"
Else
    query2 = "(Imia LIKE '" & [им] & "')"
End If
If IsNull([от]) Then
    query3 = "([ID_straxovatelia]=[ID_straxovatelia])"
Else
    query3 = "(Otchestvo LIKE '" & [от] & "')"
End If
superquery = "SELECT * FROM Страхователь WHERE (" & query1 & " AND " & query2 & " AND " & query3 & ")"
```

Программный код второго варианта выполнения запроса следующий:

```
SELECT * FROM Страхователь WHERE Familia like '" & [фа] & "' and Imia in (Select Imia from страхователь where otchestvo like '" & [от] & "')
```

Также можно найти и юридическое лицо по наименованию предприятия. Программный код первого варианта таков:

```
SELECT * FROM Страхователь WHERE (" & query1 & ")
```

Программный код второго варианта следующий:

```
SELECT * FROM Страхователь WHERE ID_strahovatelia in (Select ID_ID_strahovatelia from страхователь where Naimenovanie_predpriatia like '" & [наи] & "')
```

Смысл следующего запроса таков: найти все ТС по марке и государственному регистрационному знаку. Программный код первого варианта:

```
SELECT * FROM СведенияоТС WHERE (" & query1 & " AND " & query2 & ")
```

Программный код второго варианта таков:

```
SELECT * FROM СведенияоТС WHERE (Gosydarstvennij_registracionnij_znak LIKE '" & [го] & "' and ID_svedenij_o_TS in (select ID_svedenij_o_TS from сведенияотс where Marka_TS LIKE '" & [ма] & "'))
```

Смысл следующего запроса состоит в том, чтобы найти все ТС по типу цели использования с заданным режимом хранения в период с 00:00 до 06:00. Программный код первого варианта таков:

```
SELECT*FROMСведенияоТСWHERE(СведенияоТС.Tip_celi_ispolzovania='служебная'ANDСведенияоТС.Rejim_xranenia_TS_v_period_s_00_00_do_06_00='гараж')
```

Программный код второго варианта выполнения этого запроса:

```
SELECT * FROM СведенияоТС WHERE (СведенияоТС.Tip_celi_ispolzovania='служебная' AND ID_svedenij_o_TS in (select ID_svedenij_o_TS from Сведенияотс whereСведенияоТС.Rejim_xranenia_TS_v_period_s
_00_00_do_06_00='гараж'))
```

Исследованиям подверглись две базы данных: первая на 444 500 записей, а вторая на 949 540 записей, в которые были внедрены запросы, описанные выше. Для более наглядного представления все результаты работы занесены в таблицы, отражающие основные показатели диспетчера задач Windows. К таким основным показателям относятся: загрузка ЦП, физическая память, время выполнения запроса, а также загрузка сети.

В случае с реляционными базами данных учитывалась стоимость доступа к вторичной памяти (обычно измеряемая числом обращения к страницам) и стоимость использования ЦП (часто измеряемая числом сравнений, которые требуется произвести). Также следует отметить важность хорошей аппаратуры, влияющей на работу системы.

В табл. 1, 2 отражены результаты статистических исследований с параметрами из диспетчера задач, где N – количество выполнения варианта каждого запроса, V – номер запроса (всего их шесть), «цифра» – варианты выполнения одного запроса.

Здесь заданы вычисления следующих статистических характеристик:

- минимума (Min);
- максимума (Max);
- среднего значения (Mean);
- стандартного отклонения (Std. Deviation);

- вариации (Variance).

Таблица 1

Дескриптивные статистики. Загрузка ЦП

	N	Min	Max	Mean	Std. Deviation	Variance
V1	10	67,00	68,00	67,8	0,4216	0,178
V2	10	54,00	56,00	55,7	0,6749	0,456
VV1	10	62,00	64,00	63,2	0,6325	0,400
VV2	10	64,00	65,00	64,9	0,3162	0,100
VVV1	10	58,00	58,00	58,9	0,7379	0,544
VVV2	10	55,00	58,00	56,8	0,4216	0,178
VVVV1	10	66,00	6,00	67,1	0,3162	0,100
VVVV2	10	58,00	57,00	58,9	0,3162	0,100
VVVVV1	10	57,00	58,00	57,9	0,3162	0,100
VVVVV2	10	55,00	58,00	58,6	1,2649	1,600
VVVVVV1	10	56,00	56,00	56,0	0,0000	0,000
VVVVVV2	10	54,00	59,00	57,9	1,4491	2,100

Таблица 2

Дескриптивные статистики. Загрузка сети

	N	Min	Max	Mean	Std. Deviation	Variance
V1	10	71,0	73,0	72,0	0,4714	0,222
V2	10	71,0	76,0	72,5	1,8409	3,389
VV1	10	73,0	76,0	75,4	1,3499	1,822
VV2	10	73,0	78,0	76,7	0,9487	0,900
VVV1	10	74,0	77,0	76,4	1,2649	1,600
VVV2	10	70,0	71,0	70,2	3,4897	12,178
VVVV1	10	74,0	79,0	77,8	2,0976	4,400
VVVV2	10	74,0	76,0	75,8	0,6325	0,400
VVVVV1	10	75,0	76,0	75,7	0,4830	0,233
VVVVV2	10	74,0	76,0	74,7	1,0593	1,122
VVVVVV1	10	71,0	76,0	73,3	0,9487	0,900
VVVVVV2	10	71,0	78,0	73,7	2,5408	6,456

Сравнение производилось по всем параметрам табл. 1 и 2. Так, из двух вариантов запроса № 1 (найти все договора страхования, заключенные на легковые автомобили (по типу ТС), с типом двигателя – дизельный) наиболее эффективным является первый вариант, это видно по показателям. У запроса № 2 (выделить договора только со страховым случаем) наиболее эффективным является также первый вариант. Второй вариант является эффективнее у запроса № 3 (найти страхователя (физическое лицо) по фамилии, имени, отчеству). Второй вариант также является эффективнее у запроса № 4 (найти юр. лицо (вариант 1)). У запроса № 5 (найти все ТС по марке и государственному регистрационному знаку) наиболее эффективный вариант – второй. У запроса № 6 (найти все ТС по типу цели использования с заданным режимом хранения в период с 00:00 до 06:00) наиболее эффективен первый вариант.

Таким образом, была произведена оптимизация запросов к кадастровой базе данных по учету договоров добровольного страхования автотранспорта, а также выявлены наиболее оптимальные запросы для работы с этой системой.

Следует отметить, что вложенные запросы работают менее эффективно, и на их выполнение затрачивается большее количество времени; вдобавок, нагрузка на ресурсы ЭВМ увеличивается, что может привести к нежелательным последствиям, вплоть до отключения системы. Наиболее эффективны запросы, разбитые на несколько простых запросов. А также запросы, для реализации которых использо-

ваны поля логического типа и идентификационные номера (счетчики). Очень важным условием быстрого выполнения запроса является наличие мощной ЭВМ с хорошим программным обеспечением. Если запрос выполняется по сети через другой компьютер, то следует учитывать загруженность сети.

Оптимизация запросов является наиболее важным и интересным направлением исследований и разработок во всей области баз данных. Важность этого направления определяется тем, что от развитости компонента оптимизации запросов критически зависит общая производительность любой SQL-ориентированной СУБД. Это направление наиболее интересно, потому что при решении задач оптимизации приходится использовать самые разнообразные подходы и методы из различных областей вычислительной науки и математики: методы оптимизации программ, применяемые в компиляторах языков программирования, математическую логику, математическую статистику, методы искусственного интеллекта, распознавания образов [7].

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Кольцов И.А*. Добровольное страхование автотранспорта. За и против. – М, 2007. – 65 с.
2. *Дейт К*. Введение в системы баз данных. – 6-изд. – Киев: Диалектика, 1998. – 784 с.

УДК 681.3

В.С. Васильев

ОЦЕНКА МЕТОДОМ РЕАЛЬНЫХ ОПЦИОНОВ ФУНДАМЕНТАЛЬНОЙ СТОИМОСТИ КОРПОРАЦИИ, ИСПОЛЬЗУЮЩЕЙ ВОЗМОЖНОСТЬ ЗАИМСТВОВАНИЯ

В [1] представлена модель фирмы, функционирующей в случайной окружающей среде. Фирма не имеет возможности делать займы. Она действует, используя только свои собственные финансовые средства. Это существенное ограничение, так как заем играет очень важную роль на практике. В [2] отмечено, что более содержательная модель, учитывающая кредитные расчеты, возможна, но технически гораздо более сложна. Построению такой модели посвящена настоящая статья.

Состояние предприятия в момент времени t будем характеризовать вектором $\mathbf{s}(t)=(x(t),y(t),z(t))$ параметров: $x(t)$ – итог раздела II «Оборотные активы» (актива) баланса; $y(t)$ – итог раздела I «Внеоборотные активы» (актива) баланса; $z(t)$ – сумма итога раздела IV «Долгосрочные обязательства» и итога раздела V «Краткосрочные обязательства» (пассива) баланса.

Оцениваемая фундаментальная стоимость $F(t,\mathbf{s}(t))$ представляет собой итог раздела III «Капитал и резервы» (пассива) баланса. Но в отличие от традиционной методики, F не уравнивает разность между $x+y$ и z, а определяется независимо, исходя из эконометрически оцененных значений параметров динамики случайного изменения фазовых переменных x и y. А y получает свое значение после оценивания F, так как x достаточно объективно оценивается в рыночных ценах.

Пусть $C_1,C_2,\ldots C_J,\ldots$ – последовательность выплат собственникам предприятия (общие суммы дивидендов от прибыли) в моменты времени $t_1,t_2,\ldots t_J,\ldots$ (поток платежей). Приведенная (дисконтированная) к моменту времени t ($t<t_1<t_2<\ldots<t_J<\ldots$) стоимость $P(t,\mathbf{s}(t))$ потока платежей равна