УДК: 004.652

А.В. Маликов

Ориентированные графы в реляционных базах данных

Рассматриваются и исследуются методы управления данными, представленными в виде орграфов. Исследуются вопросы эффективного моделирования орграфов в реляционных базах данных по критериям компактности хранения и высокой производительности манипулирования данными. Приводятся примеры запросов манипулирования данными орграфов.

Ключевые слова: реляционные базы данных, иерархические структуры, ориентированные графы, модификация иерархических данных, запросы манипулирования данными.

Моделирование ориентированных графов в реляционных базах данных

Очень часто возникает необходимость в обработке больших массивов данных, представленных в виде иерархии. Если в качестве структурной абстракции при моделировании данных используется реляционная модель данных, то для моделирования иерархических структур должны использоваться специальные методы и подходы. В настоящее время не существует единого подхода к управлению иерархическими структурами данных. Наиболее исследованными, обладающими высокой эффективностью хранения и обработки данных в реляционных системах, являются методы управления деревьями. Основные направления в области моделирования деревьев в реляционных базах данных [1, 2]:

1. Использование отношения типа «родитель-потомок».
2. Использование типа данных XML.
3. Использование концепции «материализованного пути» (Materialized Path).

Последнее направление является самым молодым и предполагает использование специального ключа, сопоставляемого каждой вершине дерева и содержащего в себе информацию обо всех вершинах по пути из корня дерева в саму вершину. Существует несколько подходов к вычислению материализованного пути. Множественная модель дерева [3] предполагает сопоставление каждой вершине дерева $v \in V$, где V – множество вершин дерева, пары целочисленных значений $(\mathrm{left}_v, \mathrm{right}_v)$, такой что, если $u \in V$, $u \to v$, то $\mathrm{left}_v < \mathrm{left}_u$, $\mathrm{right}_u < \mathrm{right}_v$. В [4] для реализации идеи материализованного пути каждой вершине дерева ставится в соответствие матрица размера 2×2, содержащая целочисленные значения. Операциям модификации дерева ставятся в соответствие операции на матрицах. В [5] вводится новый тип данных на основе строкового типа данных для хранения значений материализованного пути как конкатенации символьных строк идентификаторов вершин на всех уровнях по пути из корня дерева в искомую вершину и расширяется набор функций для выполнения основных операций модификации дерева и извлечения данных. Похожий подход использован корпорацией Microsoft при создании в версии СУБД SQL Server 2008 нового типа данных HierarchyId [1, 2], являющегося расширением двоичных данных переменной длины.

Очень часто в реляционных системах возникает необходимость в моделировании иерархий данных, описываемых в виде ориентированных графов. Возможны две модификации описанных ранее методов. Так как в общем случае число вершин и дуг орграфа различно, то, в отличие от деревьев, необходимо использовать не одно реляционное отношение, а два – отдельно для хранения данных о вершинах и о характере связей между вершинами. Пусть мощность множества вершин орграфа равна n, а множества дуг – m, тогда использование концепции типа «родитель – потомок» предполагает создание реляционного отношения с вершинами мощностью n и отношения с дугами мощностью m, содержащего пару атрибутов $\{\mathrm{IDparent}, \mathrm{IDchild}\}$, являющихся внешними ключами к первичному ключу отношения с вершинами орграфа. По всей видимости, данный подход предполагает наиболее компактное хранение данных об орграфе и обладает низкими вычислительными затратами на операции модификации орграфа. Проблема такого подхода заключается в крайне низкой эффективности запросов выборки данных и, соответственно, низкой производительности решения вычислительных задач на графах.

Непосредственное использование концепции материализованного пути невозможно. Для этого необходимо без потери информации заменить орграф деревом. Возможны реализации нескольких алгоритмов такой замены, например алгоритм обхода ориентированной сети [6]. Реализация такого подхода возможна только для небольших орграфов, в противном случае это может привести к значительному росту мощности отношения с данными о дугах дерева замены, а длина ключа материализованного пути может превысить ограничения СУБД для индексных структур. Действительно, если в ключе материализованного пути зашифрована информация обо всех предках вершины, включая саму вершину, то длина ключа L некоторой вершины на уровне l ориентированной сети равна

$$L = \frac{a^l - 1}{a - 1} \cdot ceil\left(\log_2 b\right),$$

где $a > 1$ – среднее число родителей вершин орграфа; $b \in (1, \infty)$ – среднее число детей вершин орграфа; ceil() – функция, возвращающая минимальное целочисленное значение, большее аргумента; $\text{ceil}\left(\log_2 b\right)$ – длина фрагмента ключа для кодирования одной вершины на пути из корня дерева. Вообще значение длины фрагмента ключа кодирования определяется способом формирования ключа материализованного пути. Здесь использован самый оптимистичный способ кодирования, при котором для всякого родителя порядковый номер вершины-потомка кодируется битовой строкой переменной длины.

Приблизительная оценка длины ключа может быть вычислена по формуле

$$L \le l a^{l-1} \cdot \text{ceil}\left(\log_2 b\right).$$

Методы повышения эффективности моделирования ориентированных графов в реляционных базах данных

Обе описанные концепции можно обобщить, что позволит использовать достоинства каждой из них. Концепция типа «родитель – потомок» моделирует в реляционных таблицах бинарное отношение непосредственной достижимости на множестве вершин орграфа. Рефлексивно-транзитивным замыканием отношения непосредственной достижимости является отношение достижимости на орграфе. Данное отношение рефлексивно, транзитивно, антисимметрично и может быть представлено квадратной матрицей достижимости. Для реализации матрицы достижимости в реляционной системе создается две таблицы: в первой хранятся данные о вершинах, во второй – данные о достижимости между вершинами в виде пар *{предок, потомок}*. Назовем данную концепцию типа «предки – потомки».

Таблица с данными о вершинах:

```
CREATE  TABLE HierarchyObject
(ID int NOT NULL primary key,     – идентификатор вершины,
name varchar(50),                 – наименование вершины,
level int)                        – уровень вершины в ориентированной сети.
```

Таблица с данными о достижимости между вершинами:

```
CREATE  TABLE HierarchyLink
(IDancestor int NOT NULL,         – идентификатор вершины предка,
IDdescendant int NOT NULL)        – идентификатор вершины потомка.
```

Атрибуты HierarchyLink.IDancestor и HierarchyLink.IDdescendant являются внешними ключами по отношению к атрибуту HierarchyObject.ID. В большинстве запросов к таблице HierarchyLink будет накладываться фильтр по атрибуту IDancestor и/или IDdescendant, поэтому для повышения производительности таких запросов по каждому из атрибутов строится отдельный некластерный индекс типа B-дерево, в котором листовые узлы будут ссылаться на конкретные значения данных. По атрибуту с более низкой селективностью рекомендуется построить кластерный индекс, что позволит в запросах на выборку данных считывать меньшее число страниц памяти, вследствие упорядочивания записей таблицы на диске.

Предложенный подход позволяет моделировать материализованный путь в более привычном для реляционных систем виде: в виде конечного множества кортежей. На рис. 1 показано сравнение методов кодирования и хранения дерева с использованием концепции типа «предки – потомки» и материализованного пути.

Таблица HierarchyLink построена на основе матрицы достижимости вершин дерева.

Оценим мощности отношений HierarchyObject и HierarchyLink. Мощность HierarchyObject равна числу вершин орграфа n. Мощность H отношения HierarchyLink сопоставима с числом ячеек матрицы достижимости вершин орграфа:

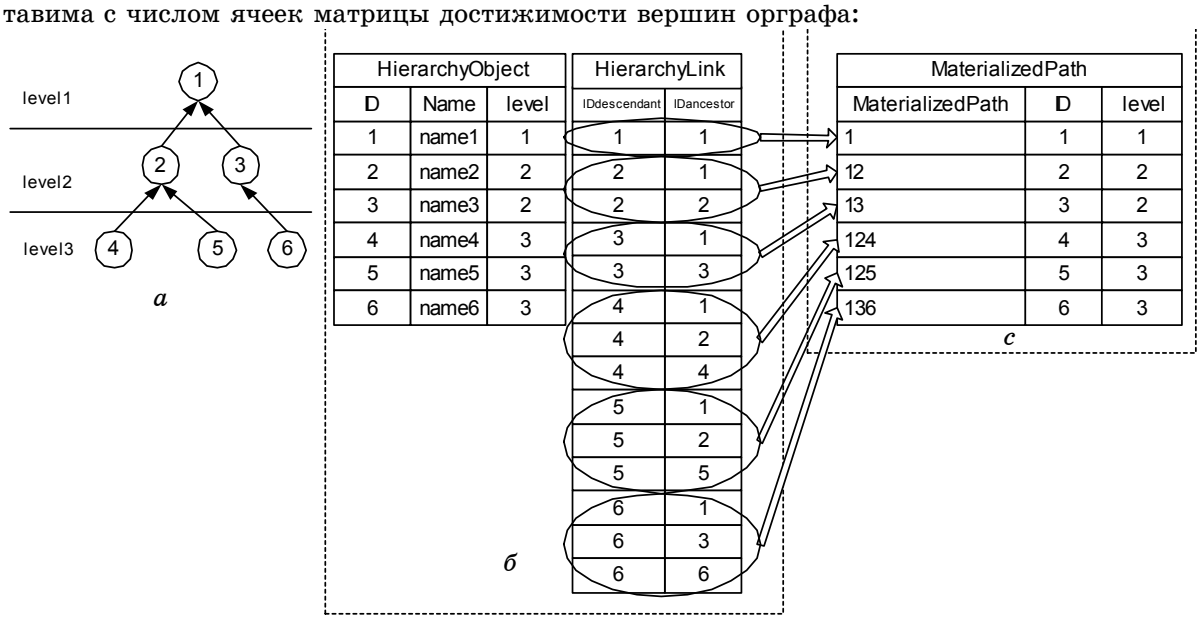

Рис. 1. Различные способы представления иерархии: *а* – в графическом виде;
б – с использованием концепции типа «предки – потомки»;
с – с использованием концепции материализованного пути

Матрица достижимости вершин дерева

вершины	1	2	3	4	5	6
1	1					
2	1	1				
3	1		1			
4	1	1		1		
5	1	1			1	
6	1		1			1

Матрица достижимости антисимметрична, следовательно, мощность отношения HierarchyLink может быть оценена как площадь прямоугольного треугольника с ячейками под главной диагональю матрицы достижимости: $H \le \frac{n^2+n}{2}$. Полученная оценка достаточно грубая, так как заполненность описываемого треугольника признаками наличия пути между парами вершин может быть произвольной и зависит от характера связей между вершинами. Всякая вершина уровня l ориентированной сети имеет $A = \left(a^0 + a^1 + a^2 + \ldots + a^{l-1}\right) \le l a^{l-1}$ достижимых вершин. Если $a = 1$, то ориентированная сеть является деревом и число достижимых вершин равно $A = l$. Если $a \in (1,\infty)$, то сумма ряда равна $A = \frac{a^l - 1}{a-1} < n$. Если ориентированная сеть состоит из l уровней, то мощность отношения HierarchyLink можно оценить по формуле $H \le \frac{a^l - 1}{a-1} \cdot n \le l a^{l-1} n$. Для дерева справедливо $a = 1$, $H \le l \cdot n$.

Манипулирование данными ориентированных графов

Рассмотрим несколько примеров, демонстрирующих особенности формирования запросов к таблицам HierarchyObject и HierarchyLink.

Задача 1: найти общих предков вершин @ID1, @ID2, @ID3. Запрос на Transact SQL имеет вид:

```
SELECT hl3.*
FROM HierarchyLink hl1, HierarchyLink hl2, HierarchyLink hl3
WHERE hl1.IDdescendant=@ID1 and
hl2.IDdescendant=@ID2 and
```

```
hl3.IDdescendant=@ID3 and
hl1.IDancestor=hl2.IDancestor and
hl2.IDancestor=hl3.IDancestor
```

Задача 2: найти общих потомков вершин @ID1, @ID2, не являющихся потомками вершины @ID3.

```
SELECT hl3.*
FROM HierarchyLink hl1, HierarchyLink hl2, HierarchyLink hl3
WHERE hl1.IDancestor=@ID1 and
hl2.IDancestor=@ID2 and
hl3.IDancestor=@ID3 and
hl1.IDdescendant=hl2.IDdescendant and
hl2.IDdescendant<>hl3.IDdescendant
```

Запросы на выборку данных из таблиц HierarchyObject и HierarchyLink обладают высокой производительностью, наглядностью и не предполагают изменения предикатов при изменении структуры орграфа.

Операции добавления, удаления и модификации дуг орграфа предполагают обработку подмножества записей таблицы HierarchyLink. Наиболее трудоемкой предполагается операция удаления дуги, при выполнении которой необходимо удалить записи о предках множества вершин-потомков. Если таблица HierarchyLink содержит только пару атрибутов {IDancestor, IDdescendant}, то на основе содержащихся в них данных невозможно гарантированно определить, какие записи подлежат удалению. Данное ограничение обусловлено возможностью существования в орграфе альтернативных путей между парами вершин. Существуют, как минимум, два способа решения данной проблемы. Во-первых, возможно дополнение таблицы HierarchyLink новыми атрибутами, отражающими характер связей между вершинами, например введение признака наличия дуги между вершинами. Тогда для выполнения операций удаления и модификации дуги необходимо выполнять рекурсивный поиск всех связанных вершин для определения их подмножества, которое будет удалено в запросе. Такой подход не может считаться удачным, так как снижает производительность запросов удаления, модификации дуг и значительно загружает аппаратные ресурсы сервера базы данных. Во-вторых, возможна модификация без потери информации исходного орграфа к орграфу, не содержащему альтернативных путей между парами вершин. Проведение такой модификации возможно с использованием специальных алгоритмов, реализованных в нормализованных на основе операций выборки и соединения базах данных [7]. Рассмотрим особенности запросов добавления, удаления и модификации дуг орграфа, в котором отсутствуют альтернативные пути между парами вершин. Исходной информацией для запросов являются значения двух переменных {@IDparent, @IDchild}, идентифицирующих конкретную дугу орграфа.

Задача 3: добавить в орграф дугу {@IDparent, @IDchild}.

```
INSERT HierarchyLink
SELECT b.IDancestor, a.IDdescendant
FROM HierarchyLink a, HierarchyLink b
WHERE a.IDancestor=@IDchild and b.IDdescendant=@IDparent
```

Предложение SELECT формирует все возможные пары из потомков вершины @IDchild, включая ее саму, и предков вершины @IDparent, включая ее саму.

Задача 4: удалить из орграфа дугу {@IDparent, @IDchild}.

```
DELETE HierarchyLink
FROM HierarchyLink a, HierarchyLink b
WHERE a.IDancestor=@IDchild and b.IDdescendant=@IDparent and
HierarchyLink.IDancestor=b.IDancestor and
HierarchyLink.IDdescendant=a.IDdescendant
```

Задача 5: заменить в орграфе дугу {@IDparentOld, @IDchild} дугой {@IDparentNew, @IDchild}.

```
DELETE HierarchyLink
FROM HierarchyLink a, HierarchyLink b
WHERE a.IDancestor=@IDchild and b.IDdescendant=@IDparentOld and
HierarchyLink.IDancestor=b.IDancestor and
HierarchyLink.IDdescendant=a.IDdescendant
INSERT HierarchyLink
SELECT b.IDancestor, a.IDdescendant
```

```
FROM HierarchyLink a, HierarchyLink b
WHERE a.IDancestor=@IDchild and b.IDdescendant=@IDparentNew
```

Выводы

Наиболее популярные методы моделирования деревьев в реляционных базах данных: отношение типа «родитель – потомок», тип данных XML, концепция материализованного пути. Обладая высокой эффективностью управления данными, представленными в виде деревьев, они не позволяют эффективно управлять данными, представленными в виде орграфов. Для моделирования орграфов в реляционных базах данных предложенные методы должны быть модифицированы. Перспективным направлением представляется отображение в реляционной системе матрицы достижимости, т.е. моделирование бинарного отношения достижимости между вершинами орграфа. Для управления множеством вершин создается отдельное реляционное отношение. Второе реляционное отношение используется для хранения отношения достижимости между парами вершин. Предложенный подход позволяет реализовать концепцию материализованного пути в более привычном для реляционных систем виде: как конечное множество кортежей реляционного отношения. По атрибутам последнего отношения строятся индексы типа сбалансированного дерева, что позволяет добиться высокой производительности запросов выборки данных. Мощности отношений оцениваются полиномиальными функциями от мощности множества вершин исходного орграфа. Представлены запросы поддержания орграфа в актуальном состоянии.

Литература

1. SQL Server 2008 Books Online (May 2008), Using hierarchyid Data Types (Database Engine) [Электронный ресурс]. – Режим доступа: http://technet.microsoft.com/enus/library / bb677173(SQL.100).aspx, свободный.

2. SQL Server 2008 Books Online (May 2008), hierarchyid (Transact-SQL). [Электронный ресурс]. – Режим доступа: http://technet.microsoft.com/en-us/library/bb677290(SQL.100).aspx, свободный.

3. Celko J., Trees in SQL [Электронный ресурс]. – Режим доступа свободный. http://www.intelligententerprise.com/ 001020/celko.shtml

4. Tropashko V. Nested Intervals Tree Encoding with Continued Fractions [Электронный ресурс]. – Режим доступа: http://arxiv.org/abs/cs.DB/0402051, свободный.

5. Roy J., Using the Node Type to Solve Problems with Hierarchies in DB2® Universal Database [Электронный ресурс]. – Режим доступа свободный. http://www-106.ibm.com/developerworks/db2/library/techarticle/0302roy/0302roy.html

6. Malikov A. Mathematical model for storing and effective processing of directed graphs in semistructured data management systems / A. Malikov, Y. Gulevskiy, D. Parkho-menko // ARTIFICIAL INTELLIGENCE, KNOWLEDGE INGINEERING and DATA BASES (AIKED '08) (UK, Cambridge, University of Cambridge). – 2008. – P. 541–548.

7. Маликов А. Проектирование реляционных баз данных на основе операций выборки и соединения. Исследование их свойств. – Ставрополь: СевКавГТУ, 2002. – 245 с.

Маликов Андрей Валерьевич
Д-р техн. наук, доцент, проф. кафедры информационных систем и технологий
Северо-Кавказского государственного технического университета
Тел.: (8652) 56-39-10
Эл. почта: malikov@ncstu.ru

A.V. Malikov

Directed graphs in relational databases

Methods for managing data, which are represented as directed graphs, are considered and studied. Problems of compact storage and high data management performance of directed graphs in relational data bases are analyzed. Power evaluation of relations keeping directed graphs data is given. Examples of data management queries are shown.

Key words: relational databases, hierarchical structures, directed graphs, hierarchical data modification, database queries