УДК 004.652.2

А.А. Фурманов, С.А. Смусенок, А.В. Трубилко

Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков

ОБЕСПЕЧЕНИЕ ДИВЕРСНОСТИ ИЕРАРХИЧЕСКИХ СТРУКТУР ДАННЫХ В РЕЛЯЦИОННЫХ СУБД

Сформулирована проблема обеспечения диверсности иерархических структур данных для задач военного применения. Анализируются проблемы хранения иерархических структур в реляционных базах данных. Рассматриваются существующие подходы для решения этих задач.

Ключевые слова: *диверсность данных, иерархические структуры, реляционные базы данных.*

Введение

Одним из эффективных способов повышения надёжности хранения данных в системах критического применения является применение принципа диверсности (разнообразия). Большой популярностью на сегодняшний день пользуется RAID-технология (redundant array of independent disks), предполагающая одновременное использование массива независимых накопителей и горячую замену одного из них в случае отказа. Однако, к системам управления, используемым в военных и космических целях, предъявляются специализированные требования к габаритно-массовым показателям, а также к показателям энергопотребления. И применение массивов из большого числа дисков не всегда является оправданным.

Применение для таких задач современных накопителей на основе флеш-памяти (SSD) позволяет снизить вышеперечисленные показатели, но этого недостаточно для повышения характеристик надёжности. Особенностью технологи SSD является то, что отказ при хранении данных может происходить не только на уровне всего накопителя, но скорее на уровне отдельных ячеек памяти. Этот факт обуславливает развитие подходов обеспечения диверсности данных в рамках одного хранилища.

Сегодня большинство хранилищ данных разрабатывается на основе реляционных баз данных. В основном они удовлетворяют требованиям какой-либо конкретной предметной области, но периодически возникают типовые задачи, требующие представления и хранения данных в иерархическом виде. В данной работе рассматривается применение принципа диверсности применительно к хранению иерархических данных.

Существует несколько подходов к построению иерархических структур в реляционных базах данных:

1. Вложенное множество (Nested Set). Представляет собой дерево, которое отражает подчинение элементов одного уровня вложенности другому (упорядоченное по уровню вложенности).

2. Список смежных вершин (Adjacency List). Для реализации данной методики применяется хранение информации о связях «наследник-родитель» в виде специальной структуры данных со списком смежных вершин и уровнями этих вершин.

3. «Замкнутые» таблицы (Closure Tables). При этом подходе хранится полный перечень путей в дереве. Возможно хранение пути нулевой длины для связи узла с самим собой.

4. Материализованный путь (Materialized Path). Идея данного подхода заключается в хранении информации о полном пути от корня дерева к данному узлу в заранее подготовленном виде.

1. Вложенное множество

Nested Set – это множество элементов (узлов), каждый из которых определяется следующей четверкой чисел:

1) id, уникальный номер элемента;

2) level, уровень вложенности элемента;

3) left_key, левый ключ элемента;

4) right_key, правый ключ элемента.

Удобно изображать элемент Nested Set схематически следующим образом (рис. 1):

Множество из элементов, определенных таким образом, может быть представлено в виде дерева, которое отражает подчинение элементов одного уровня вложенности другому (упорядоченное по уровню вложенности).

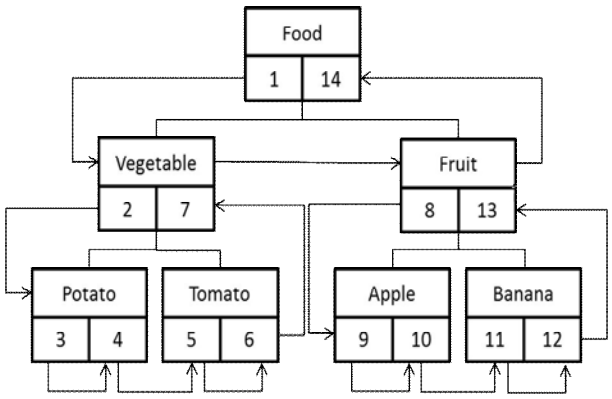

Рис. 1. Вложенное множество

2. Список смежных вершин

Поскольку узлы содержат данные, в БД можно добавить столбцы с описанием ребер дерева. Обычно это делают одним из двух способов – в виде одной таблицы или двух таблиц. В первом случае соединения ребер находятся в той же таблице, что и сам узел. Во втором – одна таблица содержит узлы графа, а другая – ребра, представленные в виде пар конечных точек. Такое двухтабличное представление позволяет обрабатывать не только деревья, но и обобщенные направленные графы. Однако такое представление не слишком отличается от однотабличного – обычно оно используется, если узлы представляют собой очень сложные объекты или дублируются и должны быть сохранены отдельно для нормализации.

В однотабличном представлении иерархий в SQL один столбец предназначается для идентификатора узла, еще один столбец того же типа данных – для родителя этого узла (рис. 2).

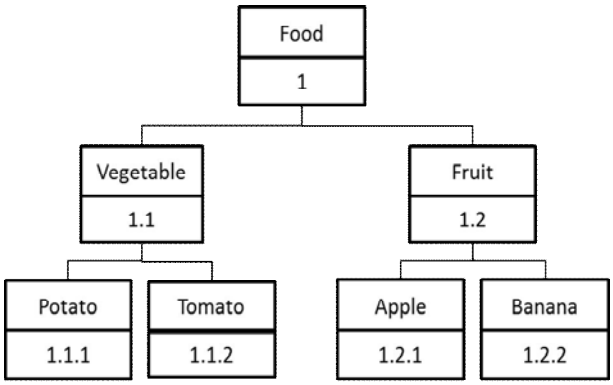

Рис. 2. Вложенное множество

В структуре классификации живых организмов столбец связи содержит идентификационный номер непосредственного таксона. Для структуры организации однотабличный подход работает очень хорошо, так как каждый организм может появиться в ней лишь один раз. Это отличает ее, к примеру, от спецификации изделия, в которую одна и та же деталь может входить несколько раз для разных сборочных единиц.

3. «Замкнутые» таблицы

Расширяя идею списков смежных вершин, можно построить таблицу, содержащую начала и концы всех путей в дереве, а также их длины. Такая таблица называется перечислением пути (path enumeration) или транзитивным замыканием (transitive closure) дерева. Последнее название обусловлено тем, что при построении данной таблицы выполняется транзитивная операция, перемещение, причем это повторяется несколько раз, пока множество путей не будет завершено (рис. 3).

Существуют убедительные основания для того, чтобы отделить друг от друга таблицу узлов и иерархию ребер. Основная причина – это нормализация. Узлы представляют собой сущности, а ребра – связи между ними. Допускается моделировать в таблице только что-либо одно.

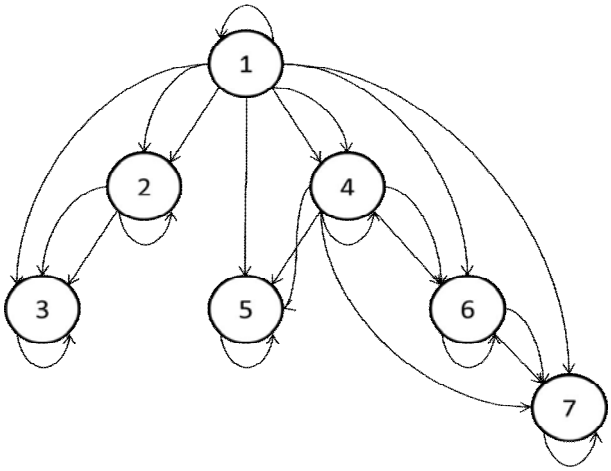

Рис. 3. «Замкнутые» таблицы

Недостаток транзитивного замыкания заключается в увеличении объёма данных таблицы с ростом дерева. Дерево, состоящее всего из одного уровня, содержит n строк. С другой стороны, если дерево представляет собой одиночную цепь узлов, она содержит (n*(n-1))/2 строк. Между ними располагается множество способов упорядочить n узлов в бинарное дерево.

4. Материализованный путь

Суть метода заключается в хранении пути от вершины до данного узла в явном виде и в качестве ключа. Например, ранее приведенная на рисунке иерархия организмов могла бы выглядеть так (рис. 4):

Данный метод напоминает нумерацию частей, разделов и глав в книге и является наиболее наглядным с точки зрения кодификации элементов: каждый узел получает интуитивно понятное значение, сам код и его части несут смысловую нагрузку. Подобные свойства важны в классификациях, предназначенных для широкого использования, например в стандартизованных справочниках территорий, отраслей экономики, медицинских диагнозов (МКБ –

международный классификатор болезней) и во многих других областях.

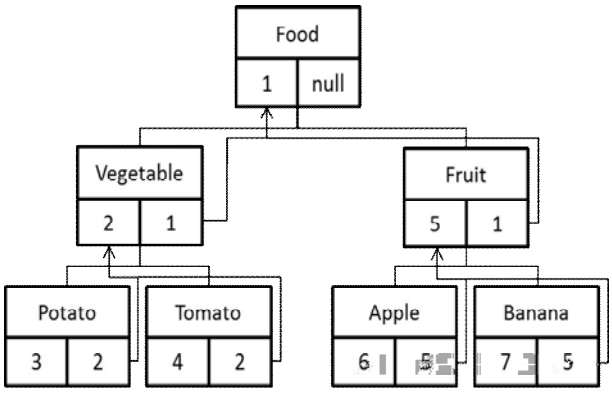

Рис. 4. Материализованный путь

Несмотря на очевидную простоту, реализация SQL-запросов может оказаться не всегда эффективной, поскольку требует поиска по подстроке.

Заключение

Проведенный анализ позволяет выявить достоинства и недостатки рассмотренных диверсных подходов к хранению данных с учётом их критического применения в системах автоматизированного управления войсками и космической технике.

Дальнейшие исследования рассмотренных подходов предполагают разработку API для работы с деревьями, что позволит измерить их эффективность и производительность на различных наборах хранимых данных и на различных СУБД.

Список литературы

1. Maintaining the transitive closure of graphs in SQL / Guozhu Dong, Leonid Libkin, Jianwen Su and Limsoon Wong // Int. Journal of Information Technology – 1999. – No.5. – P. 46-78.

2. Joe Celko's Trees and Hierarchies in SQL for Smarties / Joe Celko // Morgan Kaufmann – 2004.

3. SQL Design Patterns: Expert Guide to SQL Programming / Vadim Tropashko // Rampant Techpress – 2007.

4. Mathematical model for storing and effective processing of directed graphs in Semistructured data management systems // A. Malikov, Y. Gulevskiy, D. Parkhomenko / Artificial Intelligence, Knowledge Ingineering and Data Bases (AIKED '08) (UK,Cambridge, University of Cambridge). – 2008. – P. 541–548.

5. 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.

6. 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.

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

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

Поступила в редколлегию 5.09.2012

Рецензент: д-р техн. наук, проф. В.С. Харченко, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков.

ЗАБЕЗПЕЧЕННЯ ДІВЕРСНОСТІ ІЄРАРХІЧНИХ СТРУКТУР ДАНИХ У РЕЛЯЦІЙНИХ СКБД

О.А. Фурманов, С.А. Смусенок, О.В. Трубілко

Сформульовано проблему забезпечення діверсності ієрархічних структур даних для задач військового застосування. Аналізуються проблеми зберігання ієрархічних структур у реляційних базах даних. Розглядаються популярні підходи для вирішення цих проблем.

Ключові слова: *діверсність даних, ієрархічні структури, реляційні бази даних.*

ENSURING OF THE DIVERSITY OF THE HIERARCHICAL STRUCTURES IN A RELATIONAL DBMS

O.A. Furmanov, S.A. Smusenok, O.V. Trubilko

Problem of the diversity for hierarchical data structures for military purposes was defined. This article analyzes the problem of storing hierarchical data structures in relational databases. The main goal is to review the popular approaches for solving this problem.

Keywords: data diversity, hierarchical data structures, relational databases.