УДК 004.42

РЕДАКТОР ЭЛЕКТРОННОГО УЧЕБНОГО КУРСА ДЛЯ СТРУКТУРНО-ИЕРАРХИЧЕСКОЙ ДИДАКТИЧЕСКОЙ МОДЕЛИ

Н.С. Силкина, М.Н. Глизница

В данной статье рассмотрены подходы для хранения иерархических структур в реляционных базах данных. Авторами была выбрана наиболее подходящая модель для реализации редактора электронного учебного курса для структурно-иерархической дидактической модели. В статье также дается краткое описание реализации выбранного подхода.

Ключевые слова: структурно-иерархическая дидактическая модель, СИД модель, электронный учебный курс, ЭУК, электронное обучение, e-learning, SCORM.

Образование с использованием интернета и информационных технологий в настоящее время является актуальной задачей. Существует много различных форм, которые оно может принимать. Некоторые из них существуют лишь в качестве самообразования, в то время как другие могут предусматривать дистанционное взаимодействие с преподавателями и выдачу документа о получении образования.

Одной из самых известных на сегодняшний день онлайн-энциклопедий является «Википедия». Применение онлайн-энциклопедии не имеет определенной траектории обучения, позволяя по запросу получить информацию по широкому выбору тем. С другой стороны, различные электронные курсы, предлагаемые такими организациями как Coursera, Лекториум или Интуит, предполагают систематическое обучение, тестирование и выполнение заданий. Такие организации могут даже выдавать документы об окончании курсов. Такие курсы получили название *массовый открытый онлайн-курс* или МООК (*Massive Open Online Course*, МООС). Однако отсутствие единой структуры у большинства таких курсов является весомым недостатком, так как приводит к невозможности экспорта и импорта курсов из одной системы в другую.

На сегодняшний день существует большое количество систем управления обучением (*Learning Management System*, LMS). Часть из них является свободно распространяемым программным обеспечением. Примерами таких систем являются Moodle (moodle.org), Sakai (sakailms.org), ILIAS (ilias.de). Такие LMS как правило поддерживают международный стандарт SCORM [0–0]. Стандарт SCORM содержит требования к организации учебного материала и всей системе дистанционного обучения. Стандарт позволяет обеспечить совместимость компонентов и возможность их повторного

использования. Учебный материал согласно стандарту, представлен отдельными небольшими блоками, которые могут включаться в разные учебные курсы и использоваться системой дистанционного обучения независимо от того, кем, где и с помощью каких средств они были созданы. Помимо SCORM, существуют и другие международные стандарты, определяющие структуру и построение элементов содержания электронных учебных курсов.

Однако на сегодняшний день отсутствуют стандарты, определяющие принципы формирования дидактической структуры электронных учебных курсов, отсутствуют. В связи с этим является актуальной задача создания структурно-иерархической модели дидактического содержания электронного учебного курса (СИД модели) [0–0]. Основой для модели являются структурированные учебные модули, которые включают в себя стандартный набор дидактических компонент. Учебные модули некоторой предметной области объединяются в электронные энциклопедии. Курсы, согласно модели, представляют собой иерархическую структуру – дерево, узлам которого сопоставляются учебные модули. СИД модель позволяет генерировать SCORM-пакеты для курсов или их частей, которые могут быть использованы в любой системе дистанционного обучения, поддерживающей данный стандарт. Это свойство является основным преимуществом данной модели, поскольку появляется возможность повторного использования и переноса «удачных» учебных блоков.

В настоящее время авторами ведется разработка системы «Конструктор онлайн-курсов» для проверки адекватности разработанной модели. Одним из компонентов системы, является редактор электронного учебного курса, основное назначение которого заключается в работе с деревьями. При этом запросы на выборку узлов дерева будут выполняться намного интенсивнее, чем запросы на изменение иерархической структуры курса. Для выбора способа хранения деревьев в базе данных необходимо выбрать подходящую модель. Далее представлен обзор таких моделей.

1. Обзор моделей хранения деревьев

Существуют различные модели хранения деревьев в реляционных базах данных. К наиболее популярным относятся:

- модель «Родитель–потомок» (*Adjacency List*);
- модель материализованного пути (*Materialized Path*);
- модель «Вложенные множества» (*Nested Sets*).

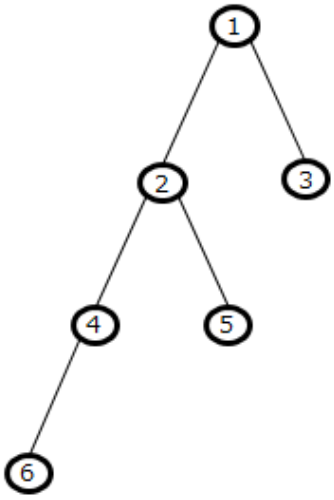
Модель «Родитель–потомок»

Модель «Родитель–потомок» [0] является наиболее простой в реализации. В соответствии с этой моделью каждая запись в таблице будет соответствовать узлу дерева и хранить его уникальный идентификатор и ссылку на родительский узел. Если узел является корнем, то ссылка на родите-

ля в его записи будет пустой. Пример дерева и его представление согласно данной модели изображены на рис. 1.

К преимуществам этой модели можно отнести низкую трудоемкость многих операций изменения структуры дерева. Для добавления нового узла достаточно лишь вставить его в таблицу со ссылкой на родителя. Перемещение узла и всего его поддерева требует изменения только одной записи в таблице – изменение ссылки на родителя для указанного узла.

Основной недостаток модели – это сложность при обращении к поколениям потомков после первого. К примеру, удаление поддерева требует рекурсивного обращения к потомкам каждого узла. Существует некоторая сложность извлечения узлов поддерева. Каждое поколение потомков некоторого узла после первого не имеет прямой связи с этим узлом, поэтому для извлечения всех потомков такого узла требуются рекурсивные процедуры, существенно увеличивающие нагрузку на сервер. Кроме того, многие сервера баз данных имеют ограничение на максимальное количество рекурсивных вызовов процедуры, что означает в свою очередь, что даже рекурсивная процедура не сможет гарантированно извлечь все поддерево вне зависимости от его высоты на любом сервере.



ID узла	ID родителя
1	NULL
2	1
3	1
4	2
5	2
6	4

Рис. 1. Пример дерева и его представление в модели «Родитель–потомок»

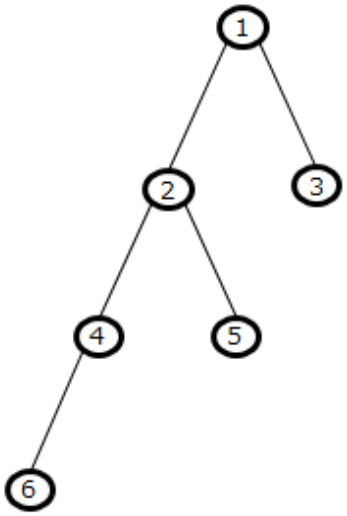
Таким образом, модель «Родитель–потомок» хорошо подходит для задач, в которых требуется активная модификация структуры дерева и не требуется выборка узлов с их потомками. Применение этой модели для задач, в которых требуется выборка поддеревьев, неэффективно.

Модель материализованного пути

Модель материализованного пути [0] похожа на модель «Родитель–потомок», но дополняет каждую запись об узле списком узлов на пути от корня дерева к узлу. Этот список должен быть упорядочен от корня и объ-

единен в строку с использованием некоторого символа в качестве разделителя. Каждая запись, как и в предыдущей модели, содержит уникальный идентификатор узла и ссылку на родителя. Пример дерева и его представление согласно данной модели приведены на рис. 2.

Эта модель в общих чертах сохраняет главное преимущество модели «Родитель–потомок», то есть простоту изменения структуры. При добавлении нового узла нужно сформировать путь к корню на основе такового в родительском узле. Перемещение поддерева требует изменения путей во всех его узлах, что является более трудоемкой задачей. Однако удаление поддерева в этой модели больше не требует рекурсии, так как осуществляется на основе записи о пути к корню. Извлечение поддерева также становится проще, так как всех потомков можно получить с помощью запроса на основе пути к корню.



ID узла	ID родителя	Путь к корню
1	NULL	/
2	1	/1/
3	1	/1/
4	2	/1/2/
5	2	/1/2/
6	4	/1/2/4/

Рис. 2. Пример дерева и его представление
в модели материализованного пути

К недостаткам модели относится тот факт, что поле пути к корню является строкой, а его длина может вырасти до значительных размеров. Это может привести к значительному увеличению размера базы данных в памяти, а также замедлить исполнение запросов в достаточно высоких деревьях.

Таким образом, данная модель материализованного пути подходит для применения во многих задачах, и приспособлена как к модификации структуры дерева, так и к извлечению поддеревьев. Однако, при этом высота дерева будет ограничена размером текстового поля, а таблица может занимать неоправданно большой объем памяти.

Модель «Вложенные множества»

Обе ранее упомянутые модели основаны на работе с древовидной структурой как с графом. Реляционная модель данных и язык SQL, однако, предназначены для работы с множествами, а не с графами, что и вызывает проблемы с представлением деревьев в реляционных базах данных.

Модель «Вложенные множества» [0] представляет древовидную структуру как набор вложенных множеств, вводя для каждого узла два целочисленных параметра: левый и правый ключ. Ключи удовлетворяют следующим условиям:

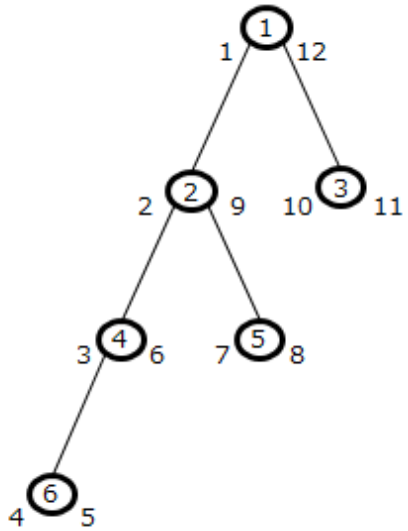
1) правый ключ узла больше левого;

2) левый ключ потомка больше левого ключа родительского узла;

3) правый ключ потомка меньше правого ключа родительского узла;

4) интервалы, определенные левыми и правыми ключами узлов, имеющих одного родителя, не должны пересекаться.

Пример дерева и его представление согласно данной модели приведены на рис. 3.



ID узла	Левый ключ	Правый ключ
1	1	12
2	2	9
3	10	11
4	3	6
5	7	8
6	4	5

Рис. 3. Пример дерева и его представление
в модели «Вложенные множества»

Главным преимуществом этой модели является скорость и простота выборки. Границы, определенные ключами, позволяют выбрать всех потомков одним запросом, который, в отличие от модели материализованного пути, не требует сравнения строк, и поэтому выполняется заметно быстрее.

К недостаткам данной модели относятся высокая сложность и низкая скорость изменения структуры, так как добавление или удаление даже одного узла может потребовать изменения многих записей вплоть до всей таблицы. Перемещение узлов и поддеревьев также будет являться слож-

ным процессом, требующим изменений как в перемещаемых узлах, так и в узлах, не затронутых перемещением.

Как итог, данная модель подходит для случаев, когда изменение структуры дерева требуется редко, а при запросах на выборку узлов требуется максимальная производительность. Таким образом, для решения проблемы хранения деревьев в редакторе электронного учебного курса модель «Вложенные множества» хорошо подходит. Далее представлено краткое описание реализации редактора электронного учебного курса для СИД модели.

2. Реализация редактора электронного учебного курса

Для реализации редактора электронного учебного курса были использованы веб-сервер Apache, интерпретаторы PHP и СУБД MySQL. Формальной моделью для хранения иерархической структуры электронного курса или *граф-плана* – выбрана модель «Вложенные множества». Таким образом, для хранения информации об узлах дерева и их связях необходима единственная таблица, структура которой представлена в табл. Пример представления граф-плана в базе данных представлен на рис. 4.

Таблица

Структура таблицы

№	Название	Тип	Семантика
1.	idNod*	integer	уникальный идентификатор узла
2.	idCour	integer	идентификатор курса
3.	idMod	integer	идентификатор учебного модуля
4.	level	integer	уровень узла (расстояние до корня дерева)
5.	leftKey	integer	левый ключ узла
6.	rightKey	integer	правый ключ узла

idNod	idCour	idMod	leftKey	rightKey	level
1	1	1	1	12	0
2	1	5	5	6	2
3	1	2	4	7	1
4	1	6	2	3	1
5	1	3	9	10	2
6	1	4	8	11	1

Рис. 4. Пример представления граф-плана в базе данных

Основные действия по модификации граф-плана – добавление, удаление и перемещение узлов – реализованы в виде веб-интерфейса. На сегодняшний день существует немало библиотек, позволяющих визуализиро-

вать древовидные структуры с помощью JavaScript. Для визуализации граф-плана была использована сторонняя библиотека Vue.js [0]. Эта библиотека позволяет отображать структуру дерева в реальном времени. На рис. 5 представлен пример визуализации граф-плана с помощью Vue.js соответствующий структуре дерева, приведенного на рис. 4.

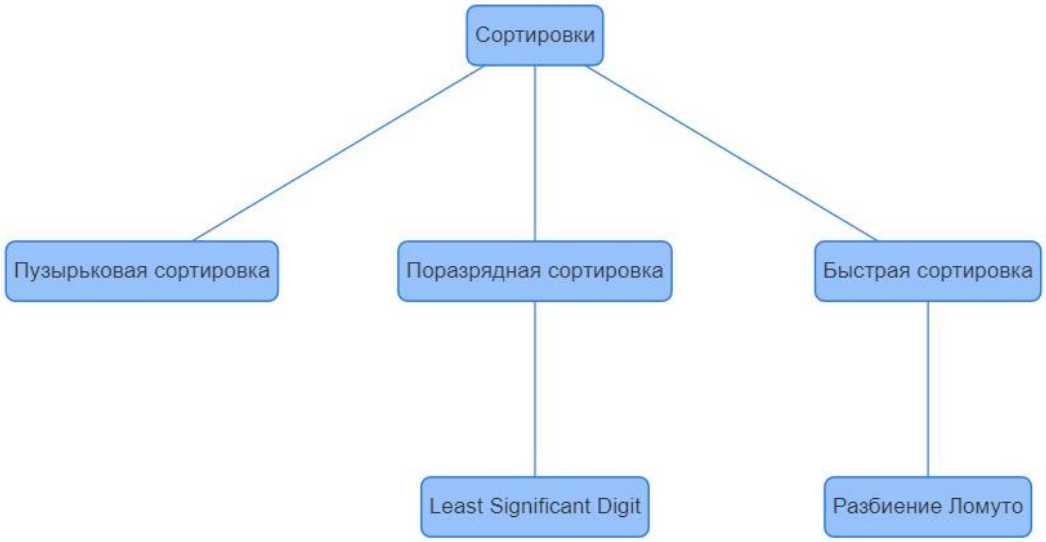


Рис. 5. Визуализация граф-плана с помощью Vue.js

Действия, направленные на изменение структуры дерева, были реализованы в виде функций языка PHP. К этим действиям относятся:

- добавление нового узла;
- удаление узла и (при наличии) его дочерних узлов;
- перемещение узла от одного родителя к другому.

Операции, меняющие структуру дерева, в рамках модели «Вложенные множества» имеют сложную реализацию. Например, алгоритм операции перемещения узла включает следующие шаги:

1) освободить место под перемещаемое поддерево;
2) переместить поддерево в освобожденное место;
3) удалить пустое место, оставшееся после перемещения поддерева.

На этапе освобождения места вычисляется ширина перемещаемого поддерева и вычисленное значение прибавляется ко всем ключам, значения которых больше или равны левому ключу, который будет иметь перемещаемый узел после перемещения.

На этапе перемещения узла ко всем ключам перемещаемого поддерева прибавляется значение расстояния, равное разнице между левым ключом, который перемещаемый узел будет иметь после перемещения и его текущим левым ключом.

На этапе удаления пустого места от всех ключей, превосходящих значение, которое имел правый ключ перемещаемого узла до перемещения, отнимается значение ширины дерева.

Редактор электронного учебного курса в полной мере выполняет обозначенные выше функции, предоставляя визуальный интерфейс для модификации структуры деревьев.

Заключение

В статье был проведен анализ распространенных моделей для хранения иерархических структур в реляционных базах данных. Была выбрана наиболее подходящая модель для реализации редактора электронного учебного курса, построенного на основе СИД модели. Выбор обоснован тем, что структура граф-плана довольно редко будет подвергаться изменениям, а выборка узлов будет производиться довольно часто. Система была реализована в виде веб-приложения с использованием библиотеки Vue.js. Редактор электронного учебного курса предоставляет пользователю визуальный графический интерфейс для добавления, изменения и удаления узлов граф-плана. Система внедрена на кафедре системного программирования ЮУрГУ и доступна по адресу https://unicst.susu.ru/ecod.

Библиографический список

1. SCORM 2004 4th Edition. Content Aggregation Model (CAM). Advanced Distributed Learning (ADL) Initiative. 2009.

2. SCORM 2004 4th Edition. Run-Time Environment (RTE). Advanced Distributed Learning (ADL) Initiative. 2009.

3. SCORM 2004 4th Edition. SCORM Users Guide for Programmers. Advanced Distributed Learning (ADL) Initiative. 2011.

4. SCORM 2004 4th Edition. Sequencing and Navigation (SN).Advanced Distributed Learning (ADL) Initiative. 2009.

5. Жигальская, Н.С. Моделирование дидактической структуры электронных учебных комплексов / Н.С. Жигальская // Вестник Южно-Уральского государственного университета. Серия «Математическое моделирование и программирование». – 2008. – № 27(127). – Вып. 2. – С. 4–9.

6. Жигальская, Н.С. Модель вариантов использования универсальной среды электронного обучения UniCST / Н.С. Жигальская // Инновационные технологии обучения: проблемы и перспективы: Материалы всерос. науч.-метод. конф. (29–30 марта 2008 г., Липецк). – Липецк: Изд-во ЛГПУ, 2008. – С. 204–207.

7. Силкина, Н.С. Система UniCST – универсальная среда электронного обучения / Н.С. Силкина, Л.Б. Соколинский // Системы управления и информационные технологии. – 2010. – № 2. – С. 81–86.

8. Маликов, А.В. Ориентированные графы в реляционных базах данных / А.В. Маликов // Доклады ТУСУР. – 2008. – № 2(18). – Ч. 2. – С. 100–104.

9. Богданов, Д.В. Оптимальный способ хранения и обработки древовидных структур в базах данных / Д.В. Богданов // Программные продукты и системы. – 2009. – № 1. – С. 140–142.

10. Joe Celko. Trees in SQL. Some answers to some common questions about SQL trees and hierarchies, 2004. – URL: http://www.ibase.ru/files/articles/programming/dbmstrees/sqltrees.html (дата обращения: 16.12.2018 г.).

11. Vue.js. – URL: https://vuejs.org (дата обращения: 28.02.2019 г.).

К содержанию