УДК 004.82, 004.65

РЕАЛИЗАЦИЯ БАЗЫ ЗНАНИЙ СТРУКТУРНЫХ ОБЪЕКТОВ

Инж. ПАВЛОВСКИЙ М. С.

Белорусский национальный технический университет

В настоящей работе рассматривается проблема хранения представлений фрагментов знаний сложной структуры (далее – фрагменты знаний) в реляционной системе управления базой данных (СУБД). Проблема состоит в том, что осуществление поиска в больших массивах информации, представленных в виде XML-документов, методом прямого перебора является низкоэффективным. В то же время XML-формат является стандартом де-факто для передачи данных. Именно XML используется для передачи данных в [1], где был представлен сервер баз знаний, основанный на использовании информационно-логических таблиц (ИЛТ) и представляющий знания для внешнего потребителя в виде XML-документов. Одним из недостатков такого подхода является то, что знания хранятся в базе данных во внутреннем двоичном формате, который не позволяет провести ни индексацию данных, ни их нормализацию, осуществить доступ к базе из других программных комплексов. В настоящей работе предлагаются механизмы, которые бы позволили преодолеть эти недостатки.

Механизм хранения фрагментов знаний, представленных в виде XML-документов. В [1] предлагается использовать технологию XML для формализации представления знаний. Этот подход позволяет вести «одновременную (параллельную) работу с одним и тем же элементом данных как человеку, так и программе» [1]. Но, кроме представления знаний, их требуется и хранить. Основные базы данных обладают реляционной структурой, а XML – иерархической, и до недавнего времени не было простого и элегантного способа интегрировать их [2].

Традиционными формами хранения XML является либо хранение каждого документа в виде отдельного файла дисковой системы компьютера, либо хранение XML-документов в таблицах БД с использованием средств, которые предоставляют СУБД для обработки XML. В [2] отмечается, что «важно помнить, что “перемещение” XML в базы данных нельзя сделать единым для всех случаев образом. У каждой модели хранения данных есть свои плюсы и минусы. Понимание того, какая модель хранения данных XML наилучшим образом соответствует потребностям вашего приложения, критично для вашего успеха». А это значит, что мы должны доказать преимущества одного способа хранения фрагментов знаний перед другим.

Особенностью фрагментов знаний является то, что имеется очень много классов объектов при небольшом количестве самих объектов. При использовании файлов в качестве хранилищ данных мы получим на выходе большое число файлов, даже если будем объединять в один файл документы, содержащие один и тот же класс документов.

Для осуществления поиска среди множества XML-файлов может быть использован язык запросов XQuery, который является наиболее мощным и современным языком, «разработанным для формулировки запросов к реальным и виртуальным XML-документам и коллекциям этих документов» [3]. Но XQuery не подходит для осуществления поиска в XML-файлах нашего формата (они не содержат достаточно информации для составления XQuery-запросов напрямую), а в [4] отмечается, что «XQuery используется преимущественно для управления контентом и интеграционными сервисами. Такие решения лучше всего подходят для создания и поддержки Web-сайтов и порталов для ограниченной аудитории».

При использовании встроенных инструментов СУБД для работы с XML-данными мы получим большое число таблиц, количество которых будет увеличиваться при добавлении новых XML-документов. Кроме неудобств, связанных с разрозненностью хранимых данных, такие подходы не позволят осуществлять быстрый поиск требуемой информации. Перебор всех документов неприемлем по времени выполнения операции, а реализация механизма индексирования потребует очень больших затрат ресурсов (человеческих и временных) и результатом будет низкая релевантность получаемых результатов.

При выборе хранилища для XML-документов следует все-таки остановиться на реляционных СУБД. Основными достоинствами РСУБД являются высокое быстродействие, надежность, большой опыт использования, отличная поддержка со стороны других производителей средств разработки [5] и пр. Но в отличие от описанного выше подхода, при котором используются встроенные средства работы с XML, в данной работе предлагается иной механизм хранения XML-данных.

Методом преодоления поставленной проблемы может служить слитное хранение фрагментов знаний, представленных в виде XML-документов, в реляционной СУБД. Слитность хранения заключается в том, что фрагмент знаний разбирается на составные части, и они последовательно сохраняются в БД. Разработанная модель БД, состоящая из фиксированного числа таблиц, позволяет хранить неограниченное число фрагментов знаний (ограничениями являются только максимальный размер БД и доступное дисковое пространство). Для увеличения скорости поиска следует создать индексы для тех полей, которые используются наиболее часто. Схема модели БД для слитного хранения фрагментов знаний представлена на рис. 1.

Слитное хранение XML-документов в реляционной СУБД позволит решить задачу поиска факта, который может встречаться в различных документах в разных контекстах. При традиционном подходе пришлось бы последовательно открывать XML-документы и осуществлять поиск средствами XML-парсеров или же последовательно осуществлять поиск по всем таблицам и полям БД, в которой хранятся XML-документы. При применении нового подхода поиск будет осуществляться в БД с учетом использования индексов, что значительно увеличивает скорость поиска среди преобразованных документов.

Примеры SQL-запросов для поиска информации. На основании приведенной схемы базы данных и ИЛТ, представленной в табл. 1, можно составить несколько примеров поисковых запросов.

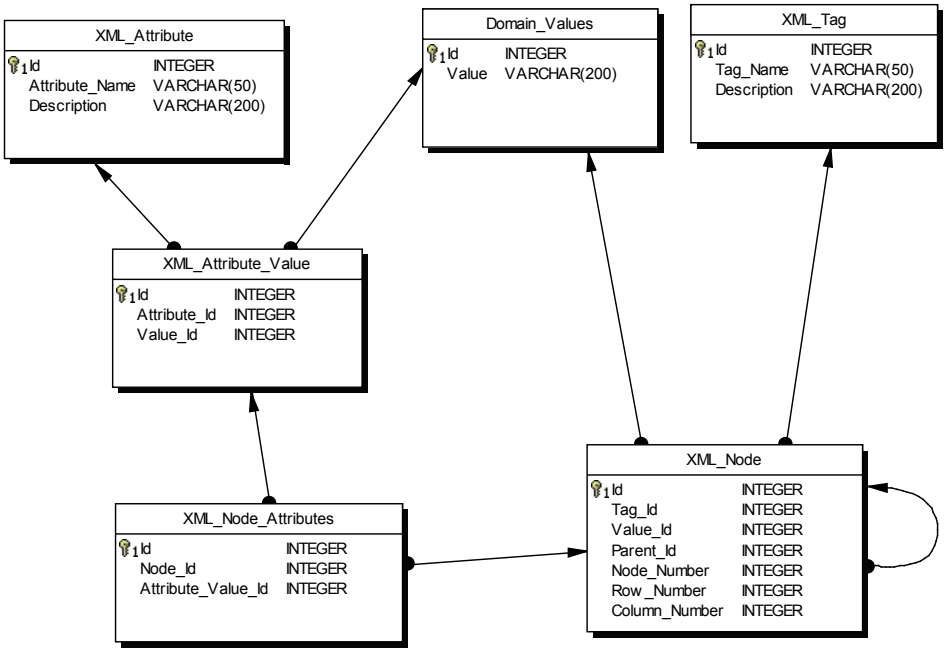

Рис. 1. Схема БД для хранения фрагментов знаний

Таблица 1

Пример ИЛТ

Точение проката обычной точности при закреплении в центрах «_per_pr32»							
Номинальный диаметр *D*	Способ обработки поверхности «СОБП»	Длина вала *L*					
		0–120	120–260	260–500	500–800	800–1250	1250–2000
		Припуск на диаметр «ПРИПУСК»					
0–30	Точение однократное	1,3	1,7	–		–	–
	Точение черновое	1,3	1,7	–	–	–	–
	Точение получистовое	0,45	0,5	–	–	–	–
	Точение чистовое	0,25	0,25	–	–	–	–
	Точение тонкое	0,13	0,15	–	–	–	–
30–50	Точение однократное	1,3	1,6	2,2	–	–	–
	Точение черновое	1,3	1,6	2,2	–	–	–
	Точение получистовое	0,45	0,45	0,50	–	–	–
	Точение чистовое	0,25	0,25	0,30	–	–	–
	Точение тонкое	0,13	0,14	0,16	–	–	–

Пример 1. Составить SQL-запрос для решения задачи «Найти все фрагменты знаний, у которых есть значения “точение однократное”».

Select distinct N.Parent_Id from XML_Node N, Domain_Values D where N.Value_Id = D.Id and D.Value = ‘точение однократное’.

Пример 2. Составить SQL-запрос для решения задачи «Найти все фрагменты знаний, у которых есть совпадение поля “Способ обработки поверхности” и значения “точение однократное”».

Select distinct N.Parent_Id from XML_Node N, XML_Node N1 where

(N.Tag_Id = GetTagIdByName(‘NL’) and N.Value_Id = GetDomainIdByValue(‘Способ обработки поверхности’) and N1.Tag_Id = GetTagIdByName(‘VL’) and N1.Value_Id = GetDomainIdByValue(‘точение однократное’) and N1.Column_Number = N.Column_Number and N1.Parent_Id = N. Parent_Id)

or

(N.Tag_Id = GetTagIdByName(‘NV’) and N.Value_Id = GetDomainIdByValue(‘Способ обработки поверхности’) and N1.Tag_Id = GetTagIdByName(‘VV’) and N1.Value_Id = GetDomainIdByValue(‘точение однократное’) and N1.Row_Number = N.Row_Number + 1 and N1.Parent_Id = N. Parent_Id).

Пример 3. Составить SQL-запрос для решения задачи «Найти решение “Припуск на диаметр” для номинального диаметра 0–30 и способ обработки поверхности “точение получистовое” при длине вала 0–120».

Select N.Value_Id from Nodes N, LeftValuesRows(:LeftConditions) LVR, VerticalValueCol-umns(:VerticalConditions) VVC where

LVR.Parent_Id = VVC.Parent_Id and

LVR.Row_Number = N.Row_Number and

VVC.Column_Number = N.Column_Number and

N.Parent_Id = LVR.Parent_Id,

где

- LeftValuesRows() – хранимая процедура, которая вычисляет список номеров строк фрагментов знаний, которые соответствуют левым наборам условий;
- VerticalValueColumns() – то же, список номеров столбцов фрагментов знаний, которые соответствуют вертикальным наборам условий.

При использовании такого подхода можно осуществлять также поиск документов, обладающих общим контекстом.

Такой подход позволяет реализовать механизм пользовательских запросов. Пользовательские запросы – это SQL подобные конструкции, которые позволяют записать поисковую конструкцию в терминах фрагмента знаний. Таким образом, если условие задачи из примера 3 переписать в виде пользовательского запроса, то получится:

Select ПРИПУСК from _per_pr32 where

D = “0-30” and СОБП = “точение получистовое” and L = “0-120”.

Этот запрос может быть транслирован во внутренние SQL-команды (как запрос из примера 3) и выполнен на сервере БД.

Сравнение быстродействия. Как было сказано, одним из главных достоинств РСУБД является ее высокое быстродействие при работе с данными, представленными в табличном виде. Для сравнения быстродействия систем, построенных на хранении данных в XML-файлах и СУБД, был проведен следующий тест. Поисковым запросом служило следующее условие: «Найти решения, для которых длина L = 260–500 и номинальный диаметр D_1 = 50–80 и способ обработки поверхности СОБП = точение черновое». Поиск решения осуществлялся в хранилищах трех различных объемов – 950, 6650 и 19000 файлов. Это количество файлов соответствует малому, среднему и большому объемам хранимой информации.

Для расчета времени поиска решения использовался следующий подход. В каждом случае поисковый запрос запускался пять раз; две попытки – самая лучшая и самая худшая отбрасывались, а на основании оставшихся попыток вычислялось среднее арифметическое, округлялось до сотых долей секунды и заносилось в таблицу (табл. 2).

Таблица 2

Сравнение скорости поиска

Количество файлов в хранилище	Тип хранилища	Время поиска решения, с	Разница, %
950	XML	2,80	2800
	БД	0,10	
6650	XML	19	2814,81
	БД	0,68	
19000	XML	56,55	2469,43
	БД	2,29	

Тестирование осуществлялось на компьютере Celeron 2.8 GHz, 512 Mb RAM, 160 Gb HDD, Win XP SP2. Для тестирования XML хранилища была разработана программа на C# (.Net Framework 2.0), которая получала на вход имя каталога, где хранятся XML-файлы, зачитывала, разбирала и выполняла поиск в каждом файле. После окончания разбора и поиска во всех файлах выводилось время, потраченное на работу. Это приложение использовало для разбора XML-документов «родной» MSXML-парсер.

Для тестирования БД выбрана СУБД Firebird 1.5.0, в которой были созданы таблицы и хранимые процедуры. Для измерения производительности СУБД использовалась система IB Expert v2007.01.20.

Приведенные данные свидетельствуют о том, что скорость поиска информации в реляционной СУБД в 24–28 раз больше, чем скорость поиска в XML-файлах. Поэтому можно считать, что выбор в пользу РСУБД полностью оправдан.

Практика применения. Для приведенной выше модели разработана база данных, в которую была импортирована информация из произвольных ИЛТ. Для БД были написаны хранимые процедуры, которые обрабатывают код, получаемый от сервера приложений (как показано в примерах). Такая реализация используется для того, чтобы сделать приложение гибким и более безопасным. Сервер приложений реализует web-сервис, который позволяет осуществлять запросы как с локального компьютера, так и с удаленного (по локальной сети или через Интернет).

Предоставляемый сервис дает возможность осуществлять различные виды поиска в ИЛТ. Это может быть поиск конкретных решений или поиск некой справочной информации. Сам сервис может быть использован как одна из составляющих систем поддержки принятия решений на этапе технологической подготовки производства.

Восстановление исходного документа из БД осуществляется очень просто – таблица «XML_NODE» содержит для каждого узла XML-документа такую информацию, как имя тэга, значение, содержащееся в узле, порядковый номер тэга в XML-документе, номер колонки и строки, в которой находится этот узел. Атрибуты, принадлежащие восстанавливаемому узлу, восстанавливаются из соответствующих таблиц и вместе с информацией об узле возвращаются в вызвавшую программу, где и преобразуются в исходный XML-документ.

ВЫВОД

Таким образом, различные фрагменты знаний, представленные в виде XML-файлов определенного формата, могут быть сохранены

в реляционную БД в разобранном виде. Такой подход позволяет значительно ускорить поиск информации в документах за счет использования всех возможностей современных реляционных СУБД и одновременно решает актуальную задачу поиска общего факта, который может содержаться в документах с разным контекстом.

Основным предназначением этого подхода является реализация доступа к базам знаний в гетерогенном информационном пространстве современного предприятия и обеспечение возможности создания интуитивно понятных запросов людьми, которые не владеют тонкостями языка SQL.

ЛИТЕРАТУРА

1. **Кочуров, В. А.** О проблеме принятия проектных решений в САПР / В. А. Кочуров // Моделирование интеллектуальных процессов проектирования, производства и управления: сб. науч. тр. – 2003. – Вып. 1. – С. 199–206.

2. **Scardina, M.** XML Storage Models: One Size Does Not Fit All / M. Scardina // Oracle Magazine [Electronic resource]. – Web column. – 2003. – Mode of access: http://otn.oracle.com/oramag/webcolumns/2003/techarticles/scardina_xmldb. html

3. **Eisenberg, A.** XQuery 1.0 is Nearing Completion / A. Eisenberg, J. Melton // SIGMOD Record [Electronic resource]. – Vol. 34, № 4. – 2005. – Mode of access: http://www.sigmod.org/sigmod/record/issues/0512/p78-column-eisenberg-melton.pdf

4. **Cohen, F.** Debunking XQuery myths and misunderstandings / F. Cohen // [Electronic resource]. – Mode of access: http://www.ibm.com/developerworks/xml/library/x-xqmyth.html

5. **Павловский, М. С.** Исследование и разработка методов создания интеллектуальных САПР на основе распределенных систем: дис. … маг. техн. наук. / М. С. Павловский. – Минск, 2003.

Поступила 25.05.2007

УДК 621.398

СИНТЕЗ СТРУКТУР СИСТЕМ СБОРА ДАННЫХ, УСТОЙЧИВЫХ К ОТКАЗАМ ОДНОТИПНЫХ ЭЛЕМЕНТОВ

ПАЦЕЙ Н. Е.

Белорусский национальный технический университет

Свойство отказоустойчивости для технических систем не является таким же однозначным, как, например, свойство надежности. Об этом можно судить исходя из количества определений, приводимых в различных источниках. В данной работе под отказоустойчивостью будем понимать способность системы выполнять свои функции в полном или частичном объеме с сохранением или частичным сохранением параметров качества выполняемых услуг. Под устойчивостью к отказам отдельных элементов будем понимать способность системы сохранять показатели качества выполняемых функций не зависимо от отказа конкретного отдельного элемента.

Рассмотрим способность системы сохранять равные показатели качества при отказе различных элементов, являющихся идентичными по техническим характеристикам. Синтезируемая система сбора данных является основой для создания автоматизированной системы коммерческого учета электроэнергии, которые, как правило, имеют три уровня сбора данных. Первый уровень представлен совокупностью счетчиков учета электроэнергии – точки учета (ТУ). Помимо параметров точности, надежности и прочего эти приборы характеризуются видом измеряемой энергии (активная, реактивная). Второй уровень представлен устройствами сбора первичных данных (УСПД), которые выполняют функции автоматического сбора данных измерений со счетчиков, энергонезависимого хранения данных энергоучета по каналам и группам учета, передачи данных на серверы