электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

УДК04.054

Обзор темпоральных баз данных и платформ их реализации

Карамнова В.М., *студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Системы обработки информации и управления»

Научный руководитель: Тоноян С.А., к.т.н., доцент,
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Системы обработки информации и управления»
tonsa@bmstu.ru

Введение

Необходимость создания эффективных алгоритмов обработки и новых методов хранения данных является одной из важных задач, встающей во многих областях. Темпоральная база данных представляется как востребованное в настоящее время расширение реляционных баз данных. Исследования в области темпоральных баз данных ведутся с конца 70-х гг. прошлого века, и по сегодняшний день остаются весьма актуальными. За это время специалистами было представлено множество методик и способов построения темпоральных баз данных, предложено много различных идей реализации систем управления, которые до сих пор полностью не воплощены ни в одной реализации крупных коммерческих системах управления базами данных.

Основой различных информационных систем (ИС) являются базы данных (БД). Информация, содержащаяся в них, является динамичной, то есть, в той или иной степени связанной с различными событиями, интервалами времени и имеет временную (темпоральную) составляющую. Обеспечение хранения и доступа к информации, а также ее достоверность определяют функциональные возможности ИС. Сегодня вместе с ростом потребностей в качестве и достоверности информации увеличивается и интерес к темпоральным базам данных. К сожалению, на данный момент не существует коммерческой реализации системы управления базами данных, обладающей полноценными темпоральными возможностями, несмотря на то, что многие производители коммерческих продуктов предлагают различные расширения и дополнения, которые позволяют работать с темпоральными данными. Однако, в силу того, что в целом на данном направлении еще не создано удовлетворительного теоретического и инструментального базиса для построения ИС, а также ввиду отсутствия

полноценных темпоральных СУБД, задачи представления темпоральных данных в интегрированной ИС остаются крайне перспективными.

Впервые идеи управления темпоральными данными появились в работе Якова Бен-Зви (Jacob Ben-Zvi) в 1982 г. Тогда внимание большинства исследователей временных баз данных, концентрировалось на расширениях реляционной модели. Когда в начале 80-х гг. начались исследования в области временных баз данных, одна из главных задач состояла в том, чтобы точно определить, что такое «время». Яков Бен-Зви в своей работе предложил два термина: эффективное время и время регистрации. Эффективное время определялось как время, «когда некоторый факт в действительности вступает в силу» (теперь его называют «действительным временем»). Время регистрации - это время, когда новое значение помещается в базу данных (называется теперь «время транзакции»). Работа Бен-Зви не получила тогда широкой известности, хотя и предвосхитила последующие исследования в области ТБД. В середине 80-х гг. начали появляться работы по темпоральной логике и использованию данных, зависимых от времени, их представлению внутри системы и визуализации для пользователя. В последствии, предлагалось много различных моделей, создавались прототипы систем темпоральных баз данных.

Важный период в области исследований ТБД пришелся на 1992–1993 гг. В 1992 г. специалисты, занимающиеся исследованиями в области баз данных, предприняли усилия для разработки согласованной версии расширения языка SQL для поддержки временных свойств - Temporal SQL. Так Ричард Снодграс (Richard Snodgrass) высказал идею о возможном темпоральном расширении стандарта языка запросов к реляционным базам данных SQL-92, а Андреас Стейнер (Andreas Steiner) и Михаель Бехлен (Michael H. Böhlen) создали прототип TimeDB, поддерживающий возможность работать и с действительным, и с транзакционным временами. Система являлась прослойкой между интерфейсом пользователя и одной из коммерческих СУБД, обеспечивая поддержку темпоральных конструкций путем их преобразования обычные реляционные структуры и обратно. Одной из предпосылок создания темпорального расширения языка SQL явилось то, что почти все приложения, использующие реляционные базы данных, являются темпоральными по своей сути. В БД хранятся данные, изменяющиеся с течением времени. С другой стороны, многие понимают, что в языке запросов SQL отсутствует адекватная и эффективная поддержка работы с ТБД. Традиционная реляционная база данных хранит информацию лишь о текущем состоянии, и СУБД не предоставляет возможности работать с данными, привязанными к определенным датам или интервалам времени (то есть

темпоральными данными). Поэтому почти во всех базах данных поддержка работы с такими данными обеспечивается усилиями программистов и администраторов, которые используют для решения задач «неудобные» конструкции языка. При этом многие простые запросы, которые легко формулируются «вне времени», довольно тяжело переписать для «самодельной» темпоральной системы, и они получаются достаточно громоздкими, что чревато появлением ошибок.

В 1999 году был принят стандарт SQL:1999. В этот стандарт не удалось включить ряд важных компонентов, среди которых средства для управления распределенными транзакциями (SQL/Transaction), поддержки временных свойств данных (SQL/Temporal), управления внешними данными (SQL/MED), связывания для объектных языков программирования (SQL/OLB), а также возможности поддержки интерактивной аналитической обработки базы данных (SQL/OLAP). Три последних компонента были доработаны и приняты ISO/IEC как дополнения к стандарту SQL:1999. Однако создание SQL/Transaction было исключено из плана развития стандарта SQL. К моменту принятия новой версии стандарта не была завершена и работа над SQL/Temporal. Новая версия стандарта SQL-2003 была принята ISO/IEC осенью 2003 г. Она также не включает SQL/Temporal. Работа над ним до сих пор продолжается.

В широком смысле под темпоральными данными следует понимать произвольные данные, явно или неявно связанные с определенными датами (моментами) или промежутками времени (интервалами). Под такое определение попадают почти любые данные и информация. Темпоральные базы данных – это базы данных, хранящие темпоральные данные. Таким образом, в темпоральной системе управления базами данных(СУБД) учитываются специфическая природа времени и изменчивость данных с течением времени.

Ранее считалось, что темпоральная СУБД будет реализована «с нуля», но вскоре исследователи обратили внимание на возможность реализации темпоральной поддержки поверх одной из существующих коммерческих СУБД в виде некоторого proxy-уровня, встраиваемого между приложением пользователя и реляционной СУБД и преобразующего темпоральные запросы в запросы на SQL-92 или SQL:1999, а результаты этих запросов, обратно, в темпоральные. При этом оставалась возможность использовать уже существующие алгоритмы и возможности языка запросов нижележащей СУБД, поэтому этот метод в дальнейшем приобрел популярность, учитывая постоянно развивающийся и дополняющийся стандарт языка SQL.

Чтобы определить, является ли рассматриваемая СУБД темпоральной в полном смысле этого слова, необходимо понять, можно ли отдельно выделить и специальным образом интерпретировать данные атрибута «время». В категорию темпоральных СУБД не будут попадать обычные реляционные СУБД, в которых поддерживаются связанные со временем типы данных, но интерпретацией и связью данных (или событий) между собой с учетом времени приходится заниматься разработчику. В «настоящей» темпоральной СУБД учитываются специфическая природа времени и изменчивость данных с течением времени.

Для эффективной реализации ТБД и анализа хранящихся в них данных предлагается использовать технологию OLAP (Online Analytical Processing). OLAP – набор технологий для оперативной аналитической обработки данных, использующих методы и средства для сбора, хранения и анализа многомерных данных в целях поддержки процессов принятия решений. В основе OLAP-технологий лежит представление информации в виде OLAP-кубов (рис. 1).

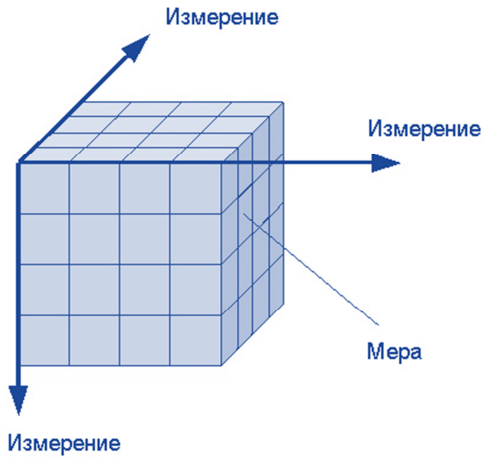

Рис. 1. Данные в виде гиперкуба

Измерение – наборы данных, необходимые для описания событий. Например, для параметра «товары», это список товаров.

Мера – данные, характеризующие количественно анализируемые факты. Например, прибыль от продажи товаров.

OLAP-кубы содержат бизнес-показатели, используемые для анализа и принятия управленческих решений. Бизнес-показатели хранятся в кубах не в виде простых таблиц, как в обычных системах учета или бухгалтерских программах, а в разрезах.

Из OLAP-куба (рис. 2) может быть составлен обычный плоский отчёт (табл. 1). По столбикам и строчкам отчёта будут бизнес-категории (грани куба), а в ячейках меры.

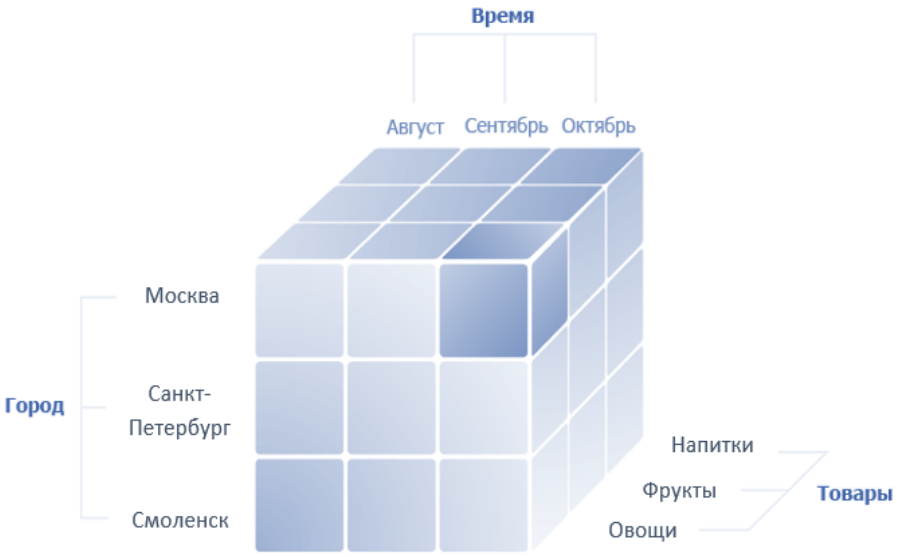

Рис. 2. Пример OLAP-куба

Таблица 1

Отчет по примеру OLAP-куба

		Август	Сентябрь	Октябрь	
Москва	Напитки	30 000	34 000	29 000	93 000
	Фрукты	28 000	29 000	28 000	85 000
	Овощи	78 000	76 500	77 500	232 000
	Итого:	136 000	139 500	134 500	410 000
Санкт-Петербург	Напитки	20 000	23 000	25 000	68 000
	Фрукты	25 000	26 500	26 500	73 000
	Овощи	79 000	77 000	77 500	233 500
	Итого:	124 000	126 500	129 000	379 500
Смоленск	Напитки	18 000	20 000	22 000	60 000
	Фрукты	23 000	22 000	20 000	65 000
	Овощи	58 000	55 000	57 500	170 500
	Итого:	99 000	97 000	99 000	295 000
Итого		359 000	363 000	362 500	1084 500

Основное назначение OLAP-систем – поддержка аналитической деятельности, произвольных запросов пользователей-аналитиков. OLAP-система состоит из двух компонентов:

- OLAP-сервер – хранит данные и выполняет над ними необходимые операции;

- OLAP-клиент – представляет пользователю интерфейс для необходимых манипуляций при анализе данных.

Назначение OLAP-анализа – проверка возникающих гипотез.

Существует три способа реализации OLAP:

1. Многомерная OLAP (Multidimensional OLAP – MOLAP) – классическая форма OLAP. Она позволяет отражать, рассматривать любые взаимосвязи в самых сложных многокомпонентных системах и создаёт требуемую пространственную схему данных с сохранением как базовых данных, так и агрегатов. Данные хранятся в многомерной базе данных. Физические данные, представленные в многомерном виде, хранятся в «плоских» файлах. При этом куб представляется в виде одной плоской таблицы, в которую построчно вписываются все комбинации членов всех измерений с соответствующими им значениями мер (табл.2).

Таблица 2

Отчет по примеру многомерных OLAP (MOLAP)

Измерения				**Мера**	
Дисциплина	**Дата**	**Преподаватель**	**Студент**	**Сумма баллов**	**Оценка**
Физика	24.01.2015	Королев И.А.	Иванова Ю.С.	90	5
Базы данных	14.01.2015	Бондарь А.А.	Сидоров А.Р.	83	4

2. Реляционная OLAP (Relational OLAP – ROLAP) – работает напрямую с реляционным хранилищем, все данные хранятся и обрабатываются в реляционных системах управления базами данных, и для хранения агрегатов создаются дополнительные реляционные таблицы. В настоящее время распространены две основные схемы реализации многомерного представления данных с помощью реляционных таблиц:

- Схема «звезда» (рис. 3) - если существует одна таблица, в которой содержится каждое измерение;

- Схема «снежинка» - если существует несколько связанных таблиц, в которых содержится хотя бы одно измерение.

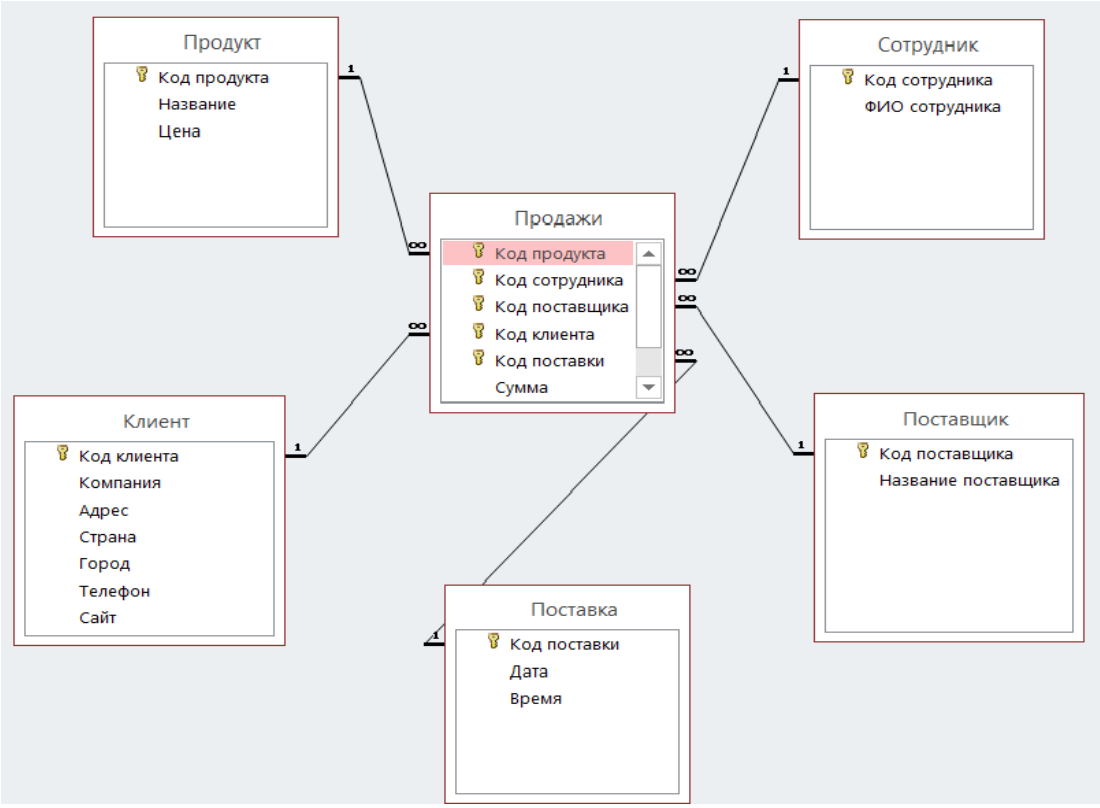

Рис. 3. Пример схемы данных «звезда»

3. Гибридная OLAP (Hybrid OLAP — HOLAP) – данные хранятся в реляционной базе данных, а агрегаты – в многомерной. Серверы HOLAP разделяют запрос на несколько подзапросов, направляют их к соответствующим фрагментам данных, комбинируют результаты, а затем предоставляют результат пользователю.

Каждый тип хранения имеет определённые преимущества, хотя есть разногласия в их оценке у разных производителей. MOLAP лучше всего подходит для небольших наборов данных, он быстро рассчитывает агрегаты и возвращает ответы, но при этом генерируются огромные объёмы данных. ROLAP оценивается как более масштабируемое решение, использующее к тому же наименьшее возможное пространство. При этом скорость обработки значительно снижается. HOLAP находится посреди этих двух подходов, он достаточно хорошо масштабируется и быстро обрабатывается. Сложность в применении OLAP состоит в создании запросов, выборе базовых данных и разработке схемы, в результате чего большинство современных продуктов OLAP поставляются вместе с огромным количеством предварительно настроенных запросов. Другая проблема — в базовых данных. Они должны быть полными и непротиворечивыми.

Сейчас сложно найти приложение, использующее базы данных для хранения и совместного использования информации и которое бы не оперировало данными,

связанными с определенными моментами или интервалами времени. Подобные приложения называются темпоральными. До сих пор не существует коммерческой реализации полноценной темпоральной СУБД, но многие производители предлагают различные расширения и дополнения, которые позволяют работать с темпоральными данными. Рассмотрим некоторые разработки:

1. TimeDB

TimeDB была разработана Андреасом Стейнером. В своей диссертации он предложил общий подход к определению и реализации временных данных. Первая версия TimeDB была написана с помощью SICStus Prolog. Версия TimeDB 2.0 реализована на языке Java, использует интерфейс JDBC и предлагает больше возможностей. Версию TimeDB 2.0 Beta 4 можно свободно скачать с сайта компании TimeConsult в исследовательских целях.

2. Informix TimeSeries Datablade

Модуль TimeSeries компании Informix предназначен для обработки, анализа динамики процессов на основе модели временных рядов и принципах физической модели. Для хранения временных рядов большого размера на дисковом пространстве сервера выделяется раздел, называемый контейнером. Для повышения эффективности при создании временного ряда можно указать, является ли он регулярным или нерегулярным. Если ряд является регулярным, то доступ константен для любого момента времени. Повышается скорость работы с регулярными временными рядами, поскольку для позиционирования в нерегулярном временном ряде используется индекс по отметкам времени. Таким образом, можно говорить об эффективном хранении и обработке временных рядов, но нельзя назвать TimeSeries Datablade полноценным темпоральным решением для действительного времени.

3. Oracle Flashback

Технология Oracle Flashback – простой способ восстановления данных на любом уровне. Можно просматривать ранние версии данных и осуществлять анализ изменений, восстановление при нарушении логической целостности. Операция Flashback Query позволяет восстанавливать единовременные копии данных, которые могли быть случайно удалены или изменены. Flashback Versions Query обеспечивает механизм просмотра изменений, произошедших в БД с течением времени, Flashback Transaction Query дает возможность просмотра изменений, внесенных заданной транзакцией. Операции восстановления Flashback Database, Flashback Query и Flashback Drop обеспечивают возможность быстрого и простого восстановления до указанного момента времени, не

зависящее от объема базы данных. Flashback Data Archive – новейшее дополнение к линейке продуктов Flashback. Данные архивов всегда готовы и доступны.

Сегодня исследования, проводимые специалистами в области темпоральных баз данных, являются весьма востребованными. Возможности информационных систем, разработанных на базе ТБД, переходят на качественно новый уровень. Ведь практически все данные, которыми оперируют ИС, в той или иной степени связаны с динамикой изменения во времени, то есть являются темпеpальными. Однако использование темпоральной модели может привести к ряду проблем, многие из которых достаточно существенны. Так, например, скорость работы запросов в темпоральной модели в некоторых ситуациях может значительно замедлиться. Остаются нераскрытыми проблемы быстродействия темпоральных моделей, возможности редактирования ошибочно введенных данных, а также вопросы сравнения различных моделей в условиях практического использования, что может послужить направлением для дальнейших исследований в данном направлении.

Заключение

В статье проведен анализ существующих технологий реализации темпоральных баз данных. На данный момент, на коммерческом рынке баз данных практически отсутствуют СУБД, которые обладали бы полноценными темпоральными возможностями, несмотря на успехи ряда отдельных разработок и решений для конкретных приложений. Возможным решением в сложившейся ситуации может быть построение темпоральной модели данных в рамках расширения реляционной модели. В будущем, спектр проблем, стоящих перед разработчиками, будет только расширяться, но в целом, ТБД представляется как востребованное и перспективное расширение реляционных БД.

Список литературы

1. Тоноян С.А., Сараев Д.В. Темпоральные модели базы данных и их свойства // Инженерный журнал: наука и инновации. 2014. № 12 (36). Режим доступа: http://engjournal.ru/articles/1333/1333.pdf (дата обращения 19.02.2015).
2. Балдин А.В., Тоноян С.А., Елисеев Д.В. Анализ избыточности хранения темпоральных данных средствами реляционных СУБД // Инженерный журнал: наука и инновации. 2014. № 4 (28). Режим доступа: http://engjournal.ru/articles/1273/1273.pdf (дата обращения 19.02.2015).

3. Порай Д. С., Соловьев А. В., Корольков Г. В. Реализация концепции темпоральной базы данных средствами реляционной СУБД: УРСС, 2004. Режим доступа: http://cognitive.ru/assets/docs/scienwork/sbornic5/Doc8.doc (дата обращения: 19.02.2015).
4. Костенко Б. Б. История и актуальные проблемы темпоральных баз данных: МГУ, 2007. Режим доступа: http://www.citforum.ru/database/articles/temporal (дата обращения: 19.02.2015).
5. Дейт К. Введение в системы баз данных. 8-е изд. М.: Вильямс, 2006. 1328 с.
6. Andreas Steiner. A Generalisation Approach to Temporal Data Models and their Implementations. Available at: http://www.timeconsult.com/Publications/diss.pdf, accessed 19.02.2015.