УДК 004.652
DOI 10.17513/snt.39699

О ПОДХОДЕ К ВСТРАИВАНИЮ ТЕМПОРАЛЬНЫХ ДАННЫХ В РЕЛЯЦИОННУЮ МОДЕЛЬ

Певнева А.Г., Обухов А.В., Кириенко А.Б.

Военно-космическая академия имени А.Ф. Можайского, Санкт-Петербург, e-mail: Pevnevaa@inbox.ru

Предлагается методика реализации, в которой используется новый подход к моделированию и реализации временной базы данных на основе интервалов поверх обычных реляционных СУБД. Такой подход существенно не изменяет процедуры проектирования и разработки информационных систем и не вносит изменения в концепцию представления темпоральных данных. Рассматривается линейное представление времени без ветвлений, время дискретно как в смысле счетности отметок на временной оси, так и в смысле возможности индексирования интервалов между отметками. Исторические изменения данных находятся в темпоральной временной схеме, а последние текущие действительные данные доступны из базовой схемы. Таблицы во временной схеме обновляются только с помощью операции вставки, когда определенный атрибут в базовой схеме таблицы обновляется, таким образом, рост этой таблицы зависит от частоты обновления атрибутов. Ограничения целостности в базовой схеме, а также временной схеме могут быть легко определены и реализованы средствами управления базами данных без каких-либо серьезных обновлений существующих приложений. Предлагаемая реализация устраняет избыточность данных и обеспечивает высокий уровень экономии памяти по сравнению с другими методами реализации.

Ключевые слова: темпоральная база данных, проектирование временных изменений данных, устранение избыточности, ограничения целостности темпоральных данных, дискретное время в реляционной модели

ABOUT APPROACH TO BUILDING TEMPORAL DATA INTO A RELATIONAL MODEL

Pevneva A.G., Obuhov A.V., Kirienko A.B.

Military Space academy named after A.F. Mozhaisky, Saint-Peterburg, e-mail:Pevnevaa@inbox.ru

The proposed implementation methodology uses a novel approach to modeling and implementing a temporal interval-based database on top of conventional relational DBMS. This approach does not significantly change the procedures for designing and developing information systems and does not change the concept of representing temporal data. A linear representation of time without branches is considered, time is discrete in the sense of the countability of marks on the time axis, and in the sense of the possibility of indexing intervals between marks. Historical data changes are in the temporal time schema, and the latest current valid data is available from the base schema. Tables in the temporary schema are only updated with an insert operation when a certain attribute in the base schema table is updated, thus the growth of this table depends on how often the attributes are updated. Integrity constraints in the base schema as well as the temporary schema can be easily defined and implemented by database management tools without any major upgrades to existing applications. The proposed implementation eliminates data redundancy and provides a high level of memory savings compared to other implementation methods.

Keywords: temporal database, designing temporal data changes, redundancy elimination, temporal data integrity constraints, discreet tame on relative model

Начиная с 1980-х годов проводились исследования, дающие основу разработки приложений обработки данных, имеющих временную составляющую. Очевидно, этот аспект более актуален для данных наблюдения, когда необходимо фиксировать изменение состояния объекта и (или) события, происходящие с объектом. Примером может служить процесс распознавания в ходе дистанционного зондирования Земли. В аналитических приложениях на первый план выходит время отклика системы в ущерб нормализации, обеспечивающей облегчение запросов в транзакционном приложении. Объемы таких данных нарастают вместе с избыточностью [1, с. 165–172], поэтому проектирование темпоральных баз данных остается актуальной проблемой.

В основном, исследования касаются структуры хранилища и обработки запросов, а также прототипа темпоральной системы управления базами данных (СУБД) [2, 3, 4], так [2] демонстрирует подход к представлению временных данных в стандартном SQL, приводятся примеры манипуляции с такими данными. Исследование в работе [3] показывает частичную реализацию темпорального подхода поверх широко используемых коммерческих СУБД, однако не приводит оценок избыточности и методов ее устранения. В работе [5] рассматриваются проблемы ограничения целостности

темпоральных данных для различных коммерческих СУБД. В работе [6] исследуются аспекты динамически изменяемой схемы данных путем добавления атрибутов в отношение и удаления утративших актуальность данных.

Во внутреннем пространстве информационной системы существуют три категории времени: действительное время – время совершения (фиксации) события в реальном мире, транзакционное, фиксирующее период сохранения данных, и битемпоральное время, обеспечивающее связь между моментами действительного и транзакционного времени. Предложенный в данной работе подход существенно не изменяет процедуры проектирования и разработки информационных систем и не вносит изменения в концепцию представления темпоральных данных. Рассматривается линейное представление времени без ветвлений, время дискретно как в смысле счетности отметок на временной оси, так и в смысле возможности индексирования интервалов между отметками. Измеримость времени обеспечивается понятием гранулярности – введением внутреннего временного интервала во внутреннем пространстве информационной системы.

Целью работы является формулировка рекомендаций по внедрению темпорального слоя в реляционную модель с наименьшими потерями как используемой памяти, так и времени исполнения запросов.

Материал и методы исследования

При проектировании временной базы данных можно рассматривать стандартные этапы: концептуальное, логическое и физическое проектирование, дополняя их действиями, обеспечивающими реализацию временного аспекта данных. Следующие шаги объясняют предложенную методологию проектирования реляционной временной базы данных в реляционной модели данных.

На первом шаге при составлении концептуальной модели бизнес-логики системы и отображении ее в реляционную модель данных игнорируются все временные аспекты.

Далее все сущности, участвующие в модели, ранжируются по уровням темпоральности. На первый уровень вносятся сущности, изменение атрибутов которых не влияет (или влияет незначительно) на логику системы, второй уровень содержит сущности с атрибутами, изменения которых в соответствии с логикой приложения отслеживаются «пассивно», т.е. частота запросов с их участием предполагается сравнительно невысокой. Значения атрибутов сущностей, обработка которых занимает центральное место в бизнес-логике системы, попадают на третий уровень.

После этого на втором и третьем уровне вводится временная шкала, в которой величина интервала деления зависима от частоты фиксации изменений атрибута.

При построении логической модели данных изменения атрибутов сущностей третьего уровня темпоральности заносятся в таблицы фактов, которая становится центральным элементом логической модели. Время в ней учитывается одним атрибутом как момент фиксации события. Для сущностей второго уровня, у которых частота изменения не так велика, допустимо представление времени периодами в виде двух атрибутов – начала и окончания периода актуальности. Для фиксирования изменений создается дополнительное отношение.

На последнем шаге определяются необходимые функции времени, в общем случае реализующие как точечные, так и интервальные операции. Они обеспечивают целостность и непротиворечивость темпоральных данных и реализуют типовые для конкретного приложения аналитические сценарии.

В математической модели данных на языке логики предикатов требуется ввести дополнительные соотношения для учета временных изменений в ходе эксплуатации [6]. Каждому отношению R в модели данных, зависящих от времени t, соответствует n-арный предикат $R(x_1, x_2 \ldots x_n, t)$, где n – число атрибутов atr_i , i=1…n в отношении R. Каждый атрибут определен на своем домене $Datr_i$. Пусть время дискретно, т.е. любой промежуток T=[$t_{нач}$, $t_{ок}$] представляется в виде последовательности T=$\{t_i\}$, где для любого t_{i1}, t_{i2} выполнено t_{i1}<t_{i2} либо t_{i1}>t_{i2}, равенство возможно только для одинаковых индексов и для всех индексов выполнено $t_i < t_{ок} \mid t_i > t_{нач}$, т.е. определено отношение порядка. Тогда событие, состоящее в изменении атрибута atr_i в интервале T, выражается предикатом:

$$C(x_i, t):\ \exists x1, x2 \in Datr \exists t_c \in T$$

$$(\forall t < t_c \rightarrow R(x_1 \ldots x_{i-1}, x1, x_{i+1}, \ldots x_n, t)) \&$$

$$\&(\forall t > t_c \rightarrow R(x_1 \ldots x_{i-1}, x2, x_{i+1}, \ldots x_n, t)),$$

а дополнительное отношение $P(Atr, ValAtr, T)$, отражающее промежутки постоянства атрибутов, представляется предикатом:

$$\text{P(y, x,T)}:\ \forall y = x_i \exists x1 \exists T\ (\forall t \in T \neg C(x_i,\ t)).$$

Результаты исследования и их обсуждение

Далее на примере подробно рассмотрено введение дополнительного отношения и функций для реализации темпоральной логики. На рисунке 1 приводится фрагмент концептуального представления модели данных кадрового делопроизводства без учета изменений, происходящих на предприятии в течение времени. В него включается одно поле, имеющее формат **Дата/время – Дата1**, этот атрибут показывает дату появления рассматриваемого объекта в модельной реальности, в примере – сотрудника на производстве.

Далее в базовое отношение **Сотрудники** (рис. 2) включаются два временных атрибута: Время начала (Дата2) и Время окончания срока службы (Дата3), в течение которого все остальные атрибуты сохраняли свои значения.

Далее для каждого отношения, для которого отслеживается история изменений, создается дополнительное отношение с тем же именем, что и в базовой схеме, с суффиксом «ИЗМ» ТАБЛИЦА_ИЗМ. Атрибуты этого отношения следующие:

– **ID** – идентификатор объекта (экземпляра сущности), ключевой атрибут в базовой схеме;

– **Индекс** – идентификатор обновленных атрибутов;

– **Значение** – используется для хранения старого значения обновленных атрибутов в базовой таблице;

– Дата_Н и **Дата_ОК** – начало и конец интервала, в течение которого значения конкретного атрибута были актуальными. Пример отношения для сущности СОТРУДНИКИ приводится на рисунке 3.

Эта таблица будет иметь составной ключ, состоящий из первичного ключа базовой таблицы, индекса и столбца **Дата_Н**. Для таблицы СОТРУДНИКИ первичный ключ (**Таб_№, индекс, Дата_Н**).

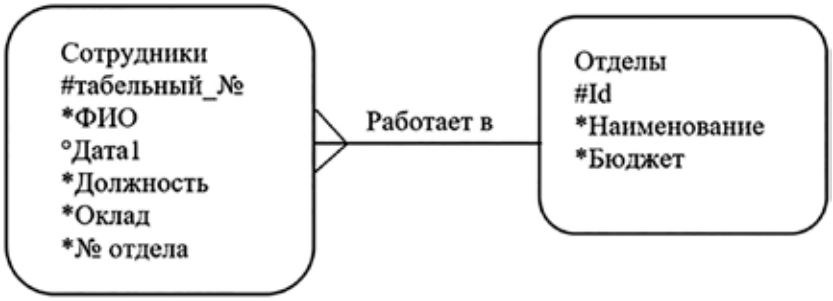

Рис. 1. Фрагмент концептуальной схемы без темпоральных данных

Сотрудники	1	2	3	4	5		
Таб_№	**ФИО**	**Дата1**	**Должность**	**Оклад**	**№_о**	**Дата2**	**Дата3**
101	Иванов	2012	Инженер	20000	10	01.02.16	01.01.3000
102	Петров	2014	Менеджер	15000	20	01.04.17	01.01.3000

Рис. 2. Отношение, в котором актуальные данные

Сотрудники_ИЗМ				
Таб_№	Индекс	Значение	Дата_Н	Дата_ОК
101	3	Менеджер	01.03.12	30.01.16
101	4	12000	01.03.12	30.01.14
101	4	15000	01.02.14	01.01.3000
102	4	10000	01.01.2014	30.03.2017
101	5	20	01. 03.12	30.01.2016

Рис. 3. Промежутки постоянства значений атрибутов

Данные в базовой таблице содержат данные после последнего обновления, т.е. актуальные в данный момент, а история изменения хранится в отношении ТАБЛИЦА_ИЗМ. Данные этой таблице обновляются автоматически с помощью триггеров базы данных или функции приложения. Далее поясняются некоторые детали операций модифицирования данных в базовой таблице.

При вставке в базовую таблицу нового экземпляра сущности значение поля Дата1 и Дата2 устанавливаются на текущую дату, а значение поля Дата3 устанавливается на отдаленное будущее время, например 1/1/3000. Эта дата всегда больше текущей даты в течение срока действия приложения.

При операции обновления прежнее значение индексированного атрибута, его индекс и соответствующее значение первичного ключа вставляются в ТАБЛИЦУ_ИЗМ, при этом, если этот атрибут обновляется впервые, то значение **Дата_Н** будет иметь то же значение, что и **Дата2** в таблице базовой схемы, а **Дата_Ок** будет иметь значение текущего времени. Если же этот атрибут уже обновлялся, то **Дата2** будет иметь значение **Дата_Ок+ед**, где **ед** – гранула внутреннего времени системы.

Удаление записи в базовой схеме выполняется путем установки значения **Дата3** на текущее время.

Запросы к временным базам данных, представленные нашим подходом с использованием стандартного SQL2, традиционно разделяются на текущий запрос, упорядоченные запросы и запросы без последовательности. Текущий запрос предоставляет актуальные данные из базовой таблицы, является «моментальным снимком» модельной реальности. Последовательный запрос предоставляет данные, которые были актуальны в течение закончившегося периода времени. Запросы без последовательности предоставляют исторические изменения данных объектов.

Некоторые текущие запросы реализуют темпоральные предикаты для исключения (включения) допустимых (недопустимых) значений атрибутов в смысле продолжительности жизни экземпляра сущности в системе, например запрос, который выбирает последний оклад уволенных сотрудников.

Упорядоченный запрос предоставляет данные, которые актуальны в течение определенного интервала времени, и результатом запроса является таблица интервалов актуальности. Например, запрос, который возвращает оклад сотрудника за определенный промежуток времени:

```
Q2 для интервала [t1 t2]
SELECT ES.Таб_№, Сотрудники_ИЗМ.значение
FROM СОТРУДНИКИ_Изм.
WHERE СОТРУДНИКИ_ИЗМ.index = 4 and
СОТРУДНИКИ_ИЗМ. Дата_Н < t2 and
СОТРУДНИКИ_ИЗМ.Дата_Ок >= t1 and
СОТРУДНИКИ_ИЗМ.Таб_№ = 89;
```

может вернуть несколько записей, поскольку интервалы изменения оклада могут перекрываться с входным временным интервалом [t1, t2]. Запрос не вернет дублированных записей, поскольку данные в модели объединены.

Запрос без последовательности возвращает исторические изменения данных объектов в течение их жизненного цикла, результатом запроса является временной срез. Сложность запросов без последовательности зависит от количества задействованных таблиц, поскольку интервалы, в течение которых выбранные записи были действительными, должны перекрываться для разных таблиц.

Для временных запросов необходимо определить интервальные функции, в широком понимании – это операции интервальной математики. Ниже приводится пример реализаций в SQL2 некоторых из них.

Функция перекрытия ([X, Y], [Z, W]) принимает два временных интервала в качестве параметров и возвращает 1, если временные интервалы перекрываются, в противном случае 0.

```
CREATE FUNCTION
OVERLAP (X IN NUMBER, Y IN NUMBER,
Z IN NUMBER, W IN NUMBER)
RETURN NUMBER
IS
BEGIN
RETURN
CASE
WHEN X > W AND Y >= Z THEN 1
ELSE 0
END;
END OVERLAP;
```

Далее для каждого атрибута, изменения которого отслеживаются во времени, создается представление. Например, представление ОКЛАД_ИЗМ включает данные, в том числе и текущие, об окладе всех сотрудников. Представление ОКЛАД_ИЗМ определяется следующим образом:

```
CREATE VIEW ОКЛАД_ИЗМ AS
SELECT СОТРУДНИКИ.ТАБ_№ СОТРУДНИКИ.ОКЛАД,
MAX (CASE
WHEN СОТРУДНИКИ.ДАТА2 IS NULL THEN СОТРУДНИКИ.Дата3
WHEN СОТРУДНИКИ_ИЗМ. Дата_Н IS NOT NULL AND
СОТРУДНИКИ.Дата2>СОТРУДНИКИ_ИЗМ.Дата_Н
THEN СОТРУДНИКИ.Дата3
WHEN СОТРУДНИКИ_ИЗМ.VET IS NOT NULL AND
СОТРУДНИКИ.LSST > СОТРУДНИКИ_ИЗМ.VET THEN (ES.VET+1)
END) AS VST, СОТРУДНИКИ.LSET AS VET
FROM СОТРУДНИКИ_VT ES WHERE
СОТРУДНИКИ_ИЗМ.АТТ_ Индекс = 4)
ON СОТРУДНИКИ.ТАБ_№ = СОТРУДНИКИ_ИЗМ.ТАБ_№
GROUP BY СОТРУДНИКИ.ТАБ_№ СОТРУДНИКИ.ОКЛАД
UNION
SELECT ТАБ_№ TO_NUMBER (Значение) VST VET
FROM СОТРУДНИКИ_ИЗМ WHERE Индекс = 4;
```

Примером запроса, который возвращает журнал отслеживания заработной платы сотрудника за время его жизни, является

```
SELECT * FROM ОКЛАД_ИЗМ
WHEN ТАБ_№ = 89;
```

Вышеуказанные запросы могут быть применены для любой другой временной информации в таблицах сотрудников или отделов.

Таблицы во временной схеме обновляются только с помощью операции вставки, когда определенный атрибут в базовой схеме таблиц обновляется, таким образом, рост этой таблицы зависит от частоты обновления атрибутов. Кластеризация временных интервалов за счет этого, по некоторым оценкам, позволяет существенно сократить время исполнения запроса даже в случае соединения отношений, содержащих до 1 миллиона записей.

Заключение

Предлагаемая методика реализации использует новый подход к моделированию и реализации временной базы данных на основе интервалов поверх обычных СУБД. Исторические изменения данных находятся в темпоральной временной схеме, а последние текущие действительные данные доступны из базовой схемы, и предложенный подход полезен по следующим причинам: ограничения целостности в базовой схеме, а также временной схеме могут быть легко определены и реализованы в СУБД без каких-либо серьезных обновлений существующих приложений.

Предлагаемая реализация устраняет избыточность данных и обеспечивает высокий уровень экономии памяти по сравнению с другими методами реализации, предложенный подход к представлению временных данных обеспечивает экономию в диапазоне использования памяти от 70 до 90% по сравнению с другими временными представлениями.

Список литературы

1. Тарасов С.Н. СУБД для программиста. Взгляд изнутри. М.: Солон Пресс, 2013. 342 с.

2. Тоноян С.А., Сараев Д.В. Темпоральные модели базы данных и их свойства // Инженерный журнал: Наука и инновации. 2014. № 12. URL: http://engjournal.ru/articles/1333/1333.pdf (дата обращения: 13.06.2023). DOI: 10.18698/2308-6033.

3. Павленко А.Г. Time series-данные в реляционной СУБД// HighLoad++Siberia2019 [электронный ресурс]. URL: https://otus.ru/nest/post/1041/ (дата обращения: 12.06.2023).

4. Novikov B.A., Gorshkova E.A. Temporal databases: From theory to applications. // Programming and Computer Software. 2008. Vol. 34. Is17. P. 1–6. DOI: 10.1134/S0361768808010015.

5. Atay C., Tansel A. BtSQL: Nested Bitemporal Relational Database Query Language. // Turkish Journal Electrical Engineering & Computer Science, 2014. Vol. 22. No. 2. P. 479-498. URL: https://journals.tubitak.gov.tr/elektrik/vol22/iss2/19/ (дата обращения: 12.06.2023). DOI: 10.3906/elk-1207-100.

6. Елисеев Д.В. Темпоральные объектно-реляционные модели в многомерном представлении // Объектные системы-2015: материалы международной научно-практической конференции (г. Ростов-на-Дону, 10-12 мая 2015 г.) Ростов н/Д.: Издательство ЮРГПУ (НПИ) им. М.И. Платова, 2015. С. 45-52.