Краткие сообщения

УДК 517.9

Темпоральная орнганизация данных и метаданных для коллективного управления данными

© **С. Н. Лизин**[1]

Аннотация. Производители СУБД и корпоративных платформ активно соревнуются в разработке новых функциональных возможностей своих продуктов. Однако многие проблемы, такие как совместная обработка данных или управление их жизненным циклом, могут быть решены проще и эффективнее посредством технологий управления данными.
Ключевые слова: Базы данных, данные, темпоральные, транзакционные, СУБД, информационные системы.

1. Введение

Возрастающие объемы информации требуют изменения бизнес-процессов предприятий и организаций. На смену традиционному бумажно-ориентированному взаимодействию, где информационные системы лишь обрабатывают сведения из бумажных документов-первоисточников, приходят электронно-ориентированные системы, в которых первоисточником являются операции, совершаемые пользователями в информационной системе.

Большинство современных корпоративных информационных систем —- это многопользовательские автоматизированные информационные системы на основе реляционных баз данных. Коллективная обработка данных в таких системах предполагает наличие конфликтов доступа к данным. В настоящее время глубоко развито научное направление управления конкурентным доступом к данным (on-line transactions processing, OLTP), изучающее обеспечение согласованности данных при их параллельной обработке.

Пусть I — информационная система, задачей которой является поддержание в ее базе данных Z сведений об об объектах множества X. Состояние множеств X и Z в момент времени t будем обозначать $X(t)$ и $Z(t)$ соответственно. Управление конкурентным доступом предполагает недопущение ситуаций, при которых одна транзакция читает данные, изменяемые другой транзакцией до ее завершения. Положим r –– некоторая транзакция, $W_r(t)$ –– некоторое подмножество базы данных $Z(t)$, на основании содержания которого выполняется транзакция r. Пусть также t_r — время чтения транзакцией r данных множества $W_r(t)$, а t_t –– время записи результата транзакции r. Тогда, если $W_r(t_r) = W_r(t_t)$, то результат транзакции r фиксируется во множестве $Z(t)$. Если же $W_r(t_r) \neq W_r(t_t)$, то результат транзакции r откатывается. Такой подход называется сериализацией транзакций.

Однако в клиент-серверных системах чтение и запись обычно производятся в разных хранилищах: чтение из локального кэша, а запись –– в СУБД. В результате подмножество W_r не всегда может быть достаточно просто выделено. Данная проблема может

[1]Доцент, ФГБОУ ВПО «Мордовский государственный педагогический институт им. М.Е. Евсевьева», г. Саранск; sergey.lizin@gmail.com.

быть решена посредством использования полных транзакций, то есть траззакций, в состав операций которых операций входит проверка неизменности множества W_r. Однако далеко не все выполняемые операции представляют собой полные транзакции. Зачастую программными средствами просто невозможно определить последовательность чтений и записей пользователем данных, составляющих атомарную транзакцию. В результате ее невозможно откатить при изменении данных чтения.

Альтернативой обеспечению гарантированной согласованности данных за счет сериализации транзакций является обеспечение возможности восстановления данных чтения для каждой транзакции (причем в распределенных системах требуется восстановление данных локального кэша/реплики) с сохранением сведений об авторстве изменений. Базы данных, обеспечивающие версионность данных, как в разрезе действительного, так и транзакционного времени, называются битемпоральными. Однако разработанные технологии использования темпоральной организации данных для целей управления коллективным доступом к данным в настоящее время отсутствуют.

Управление коллективным доступом не ограничивается отслеживанием авторства изменений. Электронно-ориентированные информационные системы требуют обеспечения юридической значимости информации с разграничением ответственности между пользователями за внесенные изменения. Для этого необходимо сохранение доказательств авторства изменений, которое обычно реализуется средствами электронной цифровой подписи.

Также следует учитывать тот факт, что не все изменения в базу данных могут вноситься по решению одного пользователя. Зачастую необходимо согласование изменений несколькими пользователями. Данная задача решается, как правило, посредством взаимодействия вне базы данных посредством обмена сообщениями. Недостатком данного подхода является невозможность восстановления данных чтения. Таким образом, более целесообразно информацию о согласовании изменений также сохранять в базе данных.

Отдельного изучения заслуживают случаи, когда обработка данных осуществляется не в одной, а в нескольких информационных системах. При этом далеко не всегда коллективное управление совместными данными несколькими информационными системами предполагает обработку единой распределенной базы данных. Количественный состав элементов, подлежащих обработке в каждой информационной системе, может значительно различаться, а обработка требоваться только в отношении совместных элементов. Кроме того, одни и те же сущности в разных информационных системах могут иметь различные значения атрибутов, причем задача синхронизации не ставится. В отношении таких элементов, как правило, требуется лишь поддержание информации о связях между сущностями. При этом требуется учитывать возможность не только таких событий как появление или исчезновение (прекращение существования) сущностей, но и их объединение, присоединение, разделение и т. п., что требует расширения множества допустимых операций над сущностями.

Существенным ограничением для решения перечисленных задач является изменчивость не только самих данных, но и используемых структур данных (метаданных). Изменение структуры таблиц — добавление и удаление, объединение и разделение столбцов — приводят к нарушению целостности электронной цифровой подписи, так как по сути исходные данные теряются. Следовательно, в отношении метаданных также должна обеспечиваться темпоральность, причем данные каждого периода должны быть доступны в соответствующей структуре. При взаимодействии информационных систем сложность представляет асинхронность используемых метаданных, а также их изменение с течением времени. Таким образом, в дополнение к поддержанию связей между сущностями, требуется поддержание связей между элементами метаданных — классами (таблицами) и атрибутами.

2. Нарушение истинности соединений предикатов

Пусть C — множество классов базы данных Z. Каждому классу $c \in C$ взаимнооднозначно соответствует некоторое реляционное отношение R_c. Множество всех реляционных отношений будем обозначать R. Схемой отношения R_c будем называть конечное множество атрибутов A_c:

$$A_c = \{a_{i_1}, a_{i_2}, \ldots, a_{i_{n_c}}\}.$$

Множество всех атрибутов базы данных Z будем обозначать A:

$$A = \bigcup_{c \in C} A_c.$$

Пусть D — множество доменов, используемых в базе данных Z. Каждому атрибуту $a \in A$ ставится в соответствие некоторый домен $D_i \in D$, причем один и тот же домен может соответствовать нескольким различным атрибутам. Домен атрибута a будем обозначать $\mathrm{dom}(a)$. Объединение доменов атрибутов класса c обозначим $\mathrm{dom}(A_c)$:

$$\mathrm{dom}(a_{i_1}) \cup \mathrm{dom}(a_{i_2}) \cup \ldots \cup \mathrm{dom}(a_{i_{n_c}}) = \bigcup_{a_i \in A_c} \mathrm{dom}(a_i) = \mathrm{dom}(A_c).$$

Декартово произведение доменов атрибутов класса c обозначим $\prod \mathrm{dom}(A_c)$:

$$\mathrm{dom}(a_{i_1}) \times \mathrm{dom}(a_{i_2}) \times \ldots \times \mathrm{dom}(a_{i_{n_c}}) = \prod_{a_i \in A_c} \mathrm{dom}(a_i) = \prod \mathrm{dom}(A_c).$$

Реляционная модель данных предполагает, что каждая база данных представляет собой структурированное множество истинных высказываний. Высказывания сгруппированы в отношения, каждое из которых задается некоторым предикатом. Каждому классу $c \in C$ сопоставим n_c–местный предикат P_c:

$$P_c(v_1, v_2, \ldots, v_{n_c}),$$

где $v_i \in \mathrm{dom}(a_i)$, $a_i \in A_c$. Кортеж $r = \langle v_1, v_2, \ldots, v_{n_c} \rangle$ принадлежит отношению R_c тогда и только тогда, когда предикат $P_c(v_1, v_2, \ldots, v_{n_c})$ является истинным. Другими словами, кортежи отношений представляют собой значения переменных, которые при подстановке в предикат превращают его в истинное высказывание. Соответственно, отношение R_c представляет собой:

$$R_c = \{\langle v_1, v_2, \ldots, v_{n_c} \rangle \in \prod \mathrm{dom}(A_c) \mid P_c(v_1, v_2, \ldots, v_{n_c}) = 1\}.$$

Архитектура современных систем управления базами данных (СУБД) и общий подход к управлению данными в значительной степени определяется историческими причинами. Существовавшие во времена появления первых СУБД компьютеры имели небольшую память, обеспечивали относительно низкую производительность и использовались в основном для проведения различного рода расчетов. В этих условия очевидным способом повышения эффективности СУБД воспринималась возможность следующих допущений:

– соединение в одной таблице множества различных данных;

– хранение в базе данных только актуальных в данный момент сведений.

Соединение в одной таблице множества различных данных заключается в том, что в каждой отдельно взятой таблице хранятся значения соединенного предиката. Например, пусть $P_1(v_1, v_2)$ и $P_2(v_1, v_3)$ — два предиката. Для хранения значений параметров, на которых данные предикаты превращаются истинные высказывания, может использоваться две таблицы — для каждого предиката, либо одна таблица для соединенного по параметру v_1 предиката: $P_1 \bowtie P_2(v_1, v_2, v_3)$.

Второй способ позволяет сократить место, необходимое для хранения значений, а также повысить скорость обработки данных за счет отсутствия необходимости выполнения операций соединения при выборке данных. В то же время, если для некоторого значения v'_1 известно значение v'_2, при котором является истинным предикат $P_1(v_1, v_2)$, но при этом не известно значение v'_3 при котором $P_1(v'_1, v'_3) = 1$, то данная информация не может быть корректно отражена в базе данных, поскольку соединение предикатов $P_1 \bowtie P_2(v'_1, v'_2, ?)$, где ? — неизвестное значение, очевидно, не будет являться истинным высказыванием.

Замечание 2.1. *На практике наиболее частый подход к решению этой проблемы основан на применении неопределенных значений (NULL) на основе трехзначной логики. Данный подход подвергается широкой критике так как нарушает реляционную модель* [1].

Второе допущение — хранение в базе данных только актуальных значений — не представляет сложности в условиях, когда базы данных используются исключительно для проведения различного рода расчетов. Однако в современных автоматизированных информационных системах СУБД используются преимущественно для целей длительного хранения данных. Особенно это важно в электронно-ориентированных системах. В частности, современное законодательство требует хранения финансовой информации не менее 3-5 лет.

Пусть $P_1(v_1, v_2)$ и $P_2(v_2, v_3)$ — два предиката. Также пусть в момент времени t' данные предикаты истинны на значениях v'_1, v'_2, v'_3:

$$P_1\big(v'_1(t'), v'_2(t')\big) = 1,\ P_2\big(v'_2(t'), v'_3(t')\big) = 1.$$

Тогда в момент времени t' на данных значениях истинным также будет высказывание на основе соединения предикатов по v_1:

$$P_1 \bowtie P_2\big(v'_1(t'), v'_2(t'), v'_3(t')\big) = 1.$$

Положим в момент времени t'' значения параметра v'_3 изменены: $v'_3(t'') \neq v'_3(t')$. Остальные же параметры остались прежними: $v'_1(t'') = v'_1(t')$, $v'_2(t'') = v'_2(t')$. При этом сохраняется истинность предикатов:

$$P_1\big(v'_1(t''), v'_2(t'')\big) = 1,\ P_2\big(v'_2(t''), v'_3(t'')\big) = 1.$$

В то же время истинность соединения предикатов может быть нарушена:

$$P_1 \bowtie P_2\big(v'_1(t''), v'_2(t''), v'_3(t'')\big) = 0.$$

3. Решение задач управления коллективным доступом к данным

В классической реляционной теории важное место занимает понятие ключа отношения. Ключом отношения R_c будем называть подмножество $K_c \subset A_c$, для которого следствием предиката P_c является:

$$\forall r_1, r_2 \in R_c : \big(r_1(A_c) \neq r_2(A_c)\big) \to \big(r_1(K_c) \neq r_2(K_c)\big).$$

Ключ представляет собой некоторый набор атрибутов, значения которых уникально идентифицируют кортеж. При этом в одном отношении может быть несколько потенциальных ключей.

Однако такая трактовка ключа не лишена недостатков. Во-первых, задача идентификации сущностей (как между X и Z, так и между Z^1 и Z^2) зачастую гораздо сложнее и не всегда может быть сведена к простому сравнению ключевых атрибутов: часть информации может отсутствовать, либо же несовпадение ключевых атрибутов не обязательно может означать различность сущностей. Характерным примером здесь может служить задача идентификации людей. Фамилия, имя, отчество и дата рождения являются недостаточной информацией для однозначной идентификации, а их различие — не основанием для того, что это разные люди. Часть информации о человеке может быть неизвестна. Во-вторых, ключевые атрибуты могут меняться с течением времени. Для людей примером может служить смена фамилии (реже имени и отчества), паспорта и т. п. В-третьих, использование ключей из нескольких атрибутов усложняет использование ссылок из других сущностей.

Качество реляционных моделей определяется соответствием входящих в них отношений (предикатов) критериям нормальных формам [2]. В качестве минимального уровня, обеспечивающего достаточную независимость данных, как правило, рассматривается нормальная форма Бойса-Кодда. Данная форма предполагает, что детерминанты всех ее функциональных зависимостей являются потенциальными ключами. Что в свою очередь означает, что каждый кортеж отношения описывает одну и только одну сущность, выделенную во множестве X.

С учетом этого, произведем некоторое расширение реляционной модели. Множество всех сущностей в базе данных Z обозначим E. Множество сущностей класса $c \in C$ будем обозначать E_c, причем:

$$\bigcup_{c \in C} E_c = E.$$

Тот факт, что некоторый элемент $x \in X$ в базе данных Z идентифицируется сущностью $e \in E$, будем обозначать: $\mathrm{id}(x) = e$.

Используя введенные обозначения введем новое определение отношения R_c:

$$R_c = \left\{ \langle e, v_{a_1}, \ldots, v_{a_{n_c}} \rangle \in E_c \times \prod \mathrm{dom}(A_c) \mid P_c(e, v_{a_1}, \ldots, v_{a_{n_c}} \right\}.$$

Предикат P_c в данном случае может быть сформулирован относительно определенно:

$$P_c(e, v_{a_1}, \ldots, v_{a_{n_c}}) \stackrel{def}{=} \exists x \in X : (\mathrm{id}(x) = e) \wedge (x(a_1) = v_{a_1}) \wedge \ldots \wedge (x(a_{n_c}) = v_{a_{n_c}}).$$

Обрабатываемые сущности и данные можно разделить на две большие категории: транзакционные и нетранзакционные. Транзакционные сущности — сущности во множестве X, каждая из которых характеризуется относительно некоторого фиксированного (точечного) момента времени, все ее характеристики также относятся к данному моменту времени и не изменяются в последующем. Транзакционные данные — это данные во множестве Z, непосредственно содержащие сведения о транзакционных сущностях множества X. Сущности множества X, не являющиеся транзакционными, называются нетранзакционными. Нетранзакционные сущности относятся к некоторому временн?му интервалу (возможно неограниченному), в течение которого его характеристики могут изменяться. Аналогично, данные непосредственно хранящие сведения о нетранзакционных сущностях называются нетранзакционными данными. Как правило, в транзакционных данных часто используются ссылки на нетранзакционные данные. Изменение (актуализация) записи

нетранзакционных данных, на которую уже имеются ссылки из записей транзакционных данных, приводит к появлению в базе данных ложных высказываний на основе соединений предикатов. Решение этой проблемы лежит в необходимости использования только транзакционных данных. Для этого необходимо решить проблему представления нетранзакционных сущностей транзакционными данными.

Для решения проблемы нарушения истинности соединенных предикатов обычно применятся следующая полнотемпоральная реляционная модель:

$$R_c^{fv} = \Big\{ r = \langle e, t_{v_b}, t_{v_e}, t_{t_b}, t_{t_e}, v_{a_1}, \ldots, v_{a_{n_c}} \rangle \in E_c \times T^4 \times \prod \mathrm{dom}(A_c) \;\Big|$$

$$P_c^f(e, t_{v_b}, t_{v_e}, v_{a_1}, \ldots, v_{a_{n_c}}) \wedge \big(\forall t_t \in [t_{t_b}, t_{t_e})\ r \in Z(t_t)\big) \Big\}.$$

Предикат $P_c^f(e, t_{v_b}, t_{v_e}, v_{a_1}, \ldots, v_{a_{n_c}})$ имеет вид:

$$P_c^f(e, t_{v_b}, t_{v_e}, v_{a_1}, \ldots, v_{a_{n_c}}) \stackrel{def}{=} \exists x \in X\ \forall t_v \in [t_{v_b}, t_{v_e}) :$$

$$Q_c(x, e, t_v) \wedge (x(a_1, t_v) = v_{a_1}) \wedge \ldots \wedge (x(a_{n_c}, t_v) = v_{a_{n_c}}),$$

где $Q_c(x, e, t_v) = \big(x \in X(t_v)\big) \wedge \big(e \in E_c(t_v)\big) \wedge \big(\mathrm{id}(x) = e\big)$.

Полнотемпоральная модель не относится к транзакционным данным, поскольку t_{v_e} и t_{t_e} являются значениями, обновляемыми после создания записи. Кроме того, для данной модели характерно наличие проблемы отсутствующих значений, так как t_{v_e} и t_{t_e} до некоторого момента являются неизвестными параметрами. Взамен полнотемпоральной модели может быть предложена простая полутемпоральная модель, обладающая свойством транзакционности:

$$R_c^{s_1} = \Big\{ r = \langle e, t_{v_b}, t_{t_b}, f, v_{a_1}, \ldots, v_{a_{n_c}} \rangle \in E_c \times T^2 \times \prod \mathrm{dom}(A_c) \;\Big|$$

$$\Big(\big(f \wedge P_c^s(e, t_{v_b}, v_{a_1}, \ldots, v_{a_{n_c}})\big) \vee \big(\neg f \wedge \bar{P}_c(e, t_{v_b})\big)\Big) \wedge$$

$$\wedge \big(\exists \varepsilon > 0 : \forall t_t \in [t_{t_b}, t_{t_b} + \varepsilon)\ r \in Z(t_t)\big) \Big\}.$$

Предикат P_c^s здесь определяется следующим образом:

$$P_c^s(e, t_{v_b}, v_{a_1}, \ldots, v_{a_{n_c}}) \stackrel{def}{=} \exists x \in X, \exists \varepsilon > 0\ \forall t_v \in [t_{v_b}, t_{v_b} + \varepsilon) :$$

$$Q_c(x, e, t_v) \wedge (x(a_1, t_v) = v_{a_1}) \wedge \ldots \wedge (x(a_{n_c}, t_v) = v_{a_{n_c}}).$$

Предикат $\bar{P}_c$ имеет вид:

$$\bar{P}_c(e, t_{v_b}) \stackrel{def}{=} \exists \varepsilon > 0\ \forall t_v \in [t_{v_b}, t_{v_b} + \varepsilon) : \neg Q_c(x, e, t_v).$$

Однако с сущностями могут происходить не только события появления, исчезновения и изменения атрибутов. Для отражения изменений в жизненном цикле атрибутов необходимо ввести дополнительное отношение жизненного цикла сущности в целом:

$$R_c^E = \big\{ r^E = \langle e_1, e_2, t_{v_b}, t_{t_b} \rangle \in E_c^2 \times T^2 \;\big|\; P_c^E(e_1, e_2, t_{v_b}) \wedge (\exists \varepsilon > 0 : \forall t_t \in [t_{t_b}, t_{t_b} + \varepsilon)\ r^E \in Z(t_t) \big\}.$$

Предикат P_c^E имеет следующий вид:

$$P_c^E(e_1, e_2, t_{v_b}) \stackrel{def}{=} \exists x_1, x_2 \in X, \exists \varepsilon > 0 : \forall t_v \in [t_{v_b}, t_{v_b} + \varepsilon)\ \neg Q_c(x_1, e_1, t_v) \wedge Q_c(x_2, e_2, t_v) \wedge (x_1 \subset x_2).$$

Поскольку данные отношения являются битемпоральным и транзакционным, на их основе могут быть решены задачи управления коллективным доступом к данным.

Для отслеживания авторства вносимых изменений с каждой записью в базу данных достаточно сохранять информацию о пользователе, внесшем ее. Поскольку все записи носят транзакционный характер, то данная информация не будет потеряна при изменении значений атрибутов сущностей. Множество пользователей информационной системы обозначим символом U, а элементы данного множества соответственно $u \in U$. Множество значений электронной цифровой подписи обозначим символом S, а элементы данного множества соответственно $s \in S$. Функцию расчета значения электронной цифровой подписи на основе закрытого ключа пользователя u по данным некоторого подмножества $\zeta \in Z$ обозначим: $\mathrm{sign}(u, \zeta)$.

Для решения задачи отслеживания исходных данных, использовавшихся пользователями при принятии решений о внесении изменений, может быть использовано транзакционное время. Пусть $\Theta \subset Z$. Введем следующие обозначения:

$$\rho^E(\Theta, \tau) = \big\{r = \langle e_1, e_2, t_{v_b}, t_{t_b}\rangle \in R_c^E \subset Z \mid (t_{t_b} = \tau) \wedge (r \in \Theta)\big\};$$

$$\rho^V(\Theta, \tau) = \big\{r = \langle e, t_{v_b}, t_{t_b}, v_{a_1}, \ldots, v_{a_{n_c}}\rangle \in R_c^V \subset Z \mid (t_{t_b} = \tau) \wedge (r \in \Theta)\big\}.$$

В случае если обработка данных распределенная, то для локальной базы данных Z^j множество исходных данных имеет вид:

$$Z\big(\max(\{t \in T \mid (t < t_t) \wedge (\rho^E(Z^j, t) \cup \rho^V(Z^j, t) \neq \emptyset)\})\big).$$

Для обеспечения отслеживания авторства всех производимых изменений с сохранением соответствующих доказательств, а также обеспечения согласования изменений, необходимо включение в модель дополнительного отношения транзакций. В качестве идентификатора транзакции целесообразно использовать транзакционное время:

$$R_c^O = \big\{r = \langle t_{t_b}, t_r, u, s\rangle \in T^2 \times U \times S \mid$$

$$t_r = \max\big(\{t \in T \mid (t < t_t) \wedge (\rho^E(Z^j, t) \cup \rho^V(Z^j, t) \neq \emptyset)\}\big) \wedge$$

$$\wedge s = \mathrm{sign}(u, \rho^E(Z, t) \cup \rho^V(Z, t))\big\}.$$

Для решения проблемы отсутствующих значений необходимо разделение каждого отношения (предиката) на простые, каждое из которых содержит значение только одного атрибута сущности:

$$R = \big\{\langle c, e, a, v\rangle \in C \times E \times A \times \mathrm{dom}(A) \mid P(c, e, a, v) = 1\big\}.$$

Предикат $P(c, e, a, v)$ имеет вид:

$$P(c, e, a, v) \stackrel{def}{=} \exists x \in X : (\mathrm{id}(x) = e) \wedge (e \in E_c) \wedge (a \in A_c) \wedge (x(a) = v).$$

Аналогичные темпоральные отношения будут иметь следующий вид:

$$R^E = \big\{r^E = \langle e_1, e_2, c, t_{v_b}, t_{t_b}\rangle \in E^2 \times C \times T^2 \mid e_1, e_2 \in E_c \wedge$$

$$\wedge P_c^E(e_1, e_2, t_{v_b}) \wedge (\exists \varepsilon > 0\ \forall t_t \in [t_{t_b}, t_{t_b} + \varepsilon)\ r^E \in Z(t_t))\big\};$$

$$R^V = \big\{r^V = \langle e, a, v, t_{v_b}, t_{t_b}\rangle \in E \times A \times \mathrm{dom}(A) \times T^2 \mid$$

$$P^V(e, a, v, t_{v_b}) \wedge (\exists \varepsilon > 0\ \forall t_t \in [t_{t_b}, t_{t_b} + \varepsilon)\ r^V \in Z(t_t))\big\}.$$

Предикат $P^V(e, a, v, t_{v_b})$ определяется следующим образом:

$$P^V(e, a, v, t_{v_b}) \stackrel{def}{=} \exists x \in X, \exists \varepsilon > 0\ \forall t_v \in [t_{v_b}, t_{v_b} + \varepsilon) : Q(x, e, t_v) \wedge (x(a, t_v) = v).$$

4. Темпоральность метаданных и управление совместными данными нескольких информационных систем

Как отмечалось во введении, одной из серьезнейших проблем информационных систем является необходимость поддержки развития структуры данных во времени, то есть темпоральность метаданных. При этом, очевидно, не должно происходить появления некорректностей, и в первую очередь — сведений об авторстве и электронной подписи.

Для описания жизненного цикла классов воспользуемся отношением, схожим с отношением, описывающим жизненный цикла сущностей:

$$R^C = \{r^C = \langle c_1, c_2, t_{v_b}, t_{t_b}\rangle \in C^2 \times T^2 \mid P^C(c_1, c_2, t_{v_b}) \wedge$$

$$\wedge(\exists \varepsilon > 0 : \forall t_t \in [t_{t_b}, t_{t_b} + \varepsilon)\ r^C \in Z(t_t))\}.$$

Предикат $P^C(c_1, c_2, t_{v_b})$ имеет вид:

$$P^C(c_1, c_2, t_{v_b}) \stackrel{def}{=} \exists \varepsilon > 0, \delta > 0 : \forall t_v^- \in (t_{v_b} - \varepsilon, t_{v_b})\ \forall t_v^+ \in [t_{v_b}, t_{v_b} + \delta)$$

$$(c_1(t_v^+) = c_1(t_v^-) \cup c_2(t_v^-)).$$

Описание жизненного цикла атрибутов, по сути, аналогично описанию жизненного цикла сущностей:

$$R^A = \{r^A = \langle a_1, a_2, c, t_{v_b}, t_{t_b}\rangle \in A^2 \times C \times T^2 \mid P^A(a_1, a_2, c, t_{v_b}) \wedge$$

$$\wedge(\exists \varepsilon > 0 : \forall t_t \in [t_{t_b}, t_{t_b} + \varepsilon)\ r^A \in Z(t_t))\}.$$

Предикат $P^A(a_1, a_2, c, t_{v_b})$ имеет вид:

$$P^A(a_1, a_2, c, t_{v_b}) \stackrel{def}{=} \exists \varepsilon > 0, \delta > 0 : \forall t_v^- \in (t_{v_b} - \varepsilon, t_{v_b})\ \forall t_v^+ \in [t_{v_b}, t_{v_b} + \delta)$$

$$(a_1(t_v^+) = \emptyset) \wedge (a_2(t_v^+) = a_1(t_v^-) \cup a_2(t_v^-)) \wedge (a_2(t_v^+) \in A_c).$$

Управление совместными данными в нескольких информационных системах предполагает решение одной из трех задач: дедупликации, актуализации или синхронизации. Решение задачи дедупликации связано с коллективным (в смысле несколькими системами) управлением связями между сущностями разных информационных систем. Задача актуализации дополнительно предполагает коллективную обработку значений атрибутов. Решение задачи синхронизации вместо управления связями между сущностями предполагает управление непосредственно самими сущностями и значениями атрибутов. Множество данных, коллективно обрабатываемых во всех информационных системах, обозначим символом L.

Для обеспечения синтаксической интероперабельности необходимо связывание метаданных (классов и атрибутов) информационных систем. Для этого во множестве L должны быть сведения определены глобальные классы и атрибуты. Предикат, задающий отношение классов, аналогичен соответствующему предикату во множествах Z^k:

$$R^{C^L} = \{r^{C^L} = \langle c_1^L, c_2^L, t_{v_b}, t_{t_b}\rangle \in C^L \times C^L \times T^2 \mid P^{C^L}(c_1^L, c_2^L, t_{v_b}) \wedge$$

$$\wedge(\exists \varepsilon > 0 : \forall t_t \in [t_{t_b}, t_{t_b} + \varepsilon)\ r^{C^L} \in L(t_t))\}.$$

Предикат $P^{C^L}(c_1^L, c_2^L, t_{v_b})$ имеет вид:

$$P^{C^L}(c_1^L, c_2^L, t_{v_b}) = \exists \varepsilon > 0, \delta > 0 : \forall t_v^- \in (t_{v_b} - \varepsilon, t_{v_b})\ \forall t_v^+ \in [t_{v_b}, t_{v_b} + \delta)$$

$$\big(c_1^L(t_v^+) = \emptyset\big) \wedge \big(c_2^L(t_v^+) = c_1^L(t_v^-) \cup c_2^L(t_v^-)\big).$$

Отношение атрибутов в множестве L также имеет схожий вид:

$$R^{A^L} = \big\{r^{A^L} = \langle a_1^L, a_2^L, c^L, t_{v_b}, t_{t_b}\rangle \in A^L \times A^L \times C^L \times T^2 \mid P^{A^L}(a_1^L, a_2^L, c^L, t_{v_b}) \wedge$$

$$\wedge (\exists \varepsilon > 0 : \forall t_t \in [t_{t_b}, t_{t_b} + \varepsilon)\ r^{A^L} \in L(t_t))\big\}.$$

Предикат $P^{A^L}(a_1^L, a_2^L, c^L, t_{v_b})$ имеет вид:

$$P^{A^L}(a_1^L, a_2^L, c^L, t_{v_b}) = \exists \varepsilon > 0, \delta > 0 : \forall t_v^- \in (t_{v_b} - \varepsilon, t_{v_b})\ \forall t_v^+ \in [t_{v_b}, t_{v_b} + \delta)$$

$$\big(a_1^L(t_v^+) = \emptyset\big) \wedge \big(a_2^L(t_v^+) = a_1^L(t_v^-) \cup a_2^L(t_v^-)\big) \wedge \big(a_2^L(t_v^+) \in A_{c^L}\big).$$

Отношение сущностей имеет вид, аналогичный R^{A^L}:

$$R^{E^L} = \big\{r^{E^L} = \langle e_1^L, e_2^L, c^L, t_{v_b}, t_{t_b}\rangle \in E^L \times E^L \times C^L \times T^2 \mid$$

$$P^{E^L}(e_1^L, e_2^L, c^L, t_{v_b}) \wedge (\exists \varepsilon > 0 : \forall t_t \in [t_{t_b}, t_{t_b} + \varepsilon)\ r^{E^L} \in L(t_t))\big\}.$$

Предикат $P^{E^L}(e_1^L, e_2^L, c^L, t_{v_b})$ имеет вид:

$$P^{E^L}(e_1^L, e_2^L, c^L, t_{v_b}) = \exists \varepsilon > 0, \delta > 0 : \forall t_v^- \in (t_{v_b} - \varepsilon, t_{v_b})\ \forall t_v^+ \in [t_{v_b}, t_{v_b} + \delta)$$

$$(a_1^L(t_v^+) = \emptyset) \wedge (e_2^L(t_v^+) = e_1^L(t_v^-) \cup e_2^L(t_v^-)) \wedge (e_2^L(t_v^+) \in E_(c^L)).$$

Пусть L^{E^j} — множество связей между сущностями множеств E^j и E^L. Отношение, задающее связи между сущностями множеств E^j и E^L, имеет вид:

$$R^{L^{E^j}} = \big\{r^{L^{E^j}} = \langle l_1^{E^j}, l_2^{E^j}, e^j, e^L, t_{v_b}, t_{t_b}\rangle \in L^{E^j} \times L^{E^j} \times E^j \times E^L \times T^2 \mid$$

$$P^{L^{E^j}}(l_1^{E^j}, l_2^{E^j}, e^j, e^L, t_{v_b}) \wedge (\exists \varepsilon > 0 : \forall t_t \in [t_{t_b}, t_{t_b} + \varepsilon)\ r^{L^{E^j}} \in L(t_t))\big\}.$$

Предикат $P^{L^{E^1}}(l_1^{E^j}, l_2^{E^j}, e^j, e^L, t_{v_b})$ имеет вид:

$$P^{L^{E^j}}(l_1^{E^j}, l_2^{E^j}, e^j, e^L, t_{v_b}) = \exists \varepsilon > 0, \delta > 0 : \forall t_v^- \in (t_{v_b} - \varepsilon, t_{v_b})\ \forall t_v^+ \in [t_{v_b}, t_{v_b} + \delta)$$

$$\gamma\big(e^j(t_v^+), e^L(t_v^+)\big) \wedge \big(l_1^{E^j}(t_v^+) = \emptyset\big) \wedge \big(l_2^{E^j}(t_v^+) = l_1^{E^j}(t_v^-) \cup l_2^{E^j}(t_v^-)\big).$$

На основе данных отношений может осуществляться хранение совместных данных для решения задачи дедупликации сущностей. В целях решения задачи дедупликации, все сущности, создаваемые в каждой информационной системе, должны проходить процедуру идентификации. Если в качестве ответного значения от функции идентификации возвращается идентификатор существующего объекта, то данный идентификатор присваивается сущности в L.

Использование данной модели позволяет обеспечить асинхронное согласование модификаций. При этом решение о возможности использования записей принимается системами на основе установленных правил в зависимости от состава согласований. Для данной модели множества L в работе предложены алгоритмы управления совместными данными.

5. Реализация моделей в СУБД Microsoft SQL Server

Для реализации предложенных реляционных моделей средствами стандартных СУБД требуется поддержка последними универсального или неопределенного типа данных, такого как sql_variant в Microsoft SQL Server. При этом для осуществления простого отката транзакций, как на уровне сервера баз данных, так и на уровне сервера приложений, посредством удаления только соответствующей записи транзакции требуется использование ограничений внешних ключей между таблицей транзакций и остальными таблицами.

Недостаточная приспособленность средств языка SQL к обработке темпоральных данных приводит к необходимости использования соединений типа self-join, где для упрощения написания запросов требуется использование вспомогательных функций, а для повышения производительности при их обработке – соответствующих индексов.

Для прозрачного внедрения темпоральной технологии организации данных в существующие системы требуется использование представлений, замещающих собой соответствующие использовавшиеся ранее таблицы, для обработки запросов модификации данных — триггеров типа “instead of”:

1. Выполняется генерация SQL-дампа исходной таблицы базы данных;
2. Создаются темпоральные таблицы Элементов и Атрибутов;
3. Удаляется исходная таблица;
4. Создается представление (VIEW) с тем же именем, что и исходная таблица;
5. Для созданного представления создаются триггеры INSTEAD OF INSERTUPDATEDELETE;
6. Выполняется восстановление данных посредством запуска скрипта SQL-дампа.

Экспериментальная проверка предложенной моделей темпоральной организации данных проводилась посредством сравнения ранее известной модели полнотемпоральной организации данных с полутемпоральной моделью на базе одного отношения (совпадающего по функциональности с полнотемпоральной моделью), полутемпоральной моделью на базе двух отношений (с расширением операций над сущностями) и полутемпоральной моделью на базе четырех отношений (дополнительно обеспечивающую темпоральность метаданных).

В первую очередь была произведена вставка записей с шагом в 10 000 сущностей. Изменение размера баз данных представлено на диаграмме.

–

Затем была произведена операция обновления значений атрибутов (в соответствии с установленными правилами). Изменения размера базы данных после каждых 10 000 обновлений (одно обновление каждой записи) изображено на диаграмме.

—

Таким образом, полутемпоральная модель на основе одного отношения (значений) во всех случаях эффективнее полнотемпоральной модели на 8–12

При сравнении полнотемпоральной модели и полутемпоральной модели на основе 2 отношений (сущностей и значений) имеет значение соотношение количества новых сущностей и обновлений. Полутемпоральная модель на основе двух отношений более эффективна в случаях, когда количество обновлений в 2,46 раза больше количества сущностей.

При сравнении полнотемпоральной модели и полутемпоральной модели на основе 4 отношений (классов, атрибутов, сущностей и значений) имеет значение соотношение количества новых сущностей, обновлений и среднего количества обновляемых атрибутов. Если k — отношение количества сущностей к количеству обновлений, z — среднее количество одновременно обновляемых атрибутов каждой сущности, то полутемпоральная модель на основе четырех отношений более эффективна, когда выполняется неравенство:

$$z < 2,754 - 3,548k.$$

Поскольку z всегда не меньше 1, то, должно выполняться соотношение:

$$k < (2,754 - 1)/3,548 \approx 0,494.$$

То есть на каждую сущность в среднем должно быть больше двух обновлений. Также, очевидно, что z всегда меньше 2,754.

Список литературы

1. Date C.J., Darwen H. Databases, types and the relational model: the third manifesto (3rd ed.). — Addison-Wesley. — Reading, MA., 2006 — ISBN 0-321-39942-0.

2. Дейт К.Дж. Введение в системы баз данных (8-е изд.) / Издательский дом "Вильямс", М. — 2005. — ISBN 5-8459-0788-8.

Collaborate data management using temporal data and metadata

© S. N. Lizin[2]

Abstract. DBMS and enterprise frameworks vendors compete for developing new functionalities of their products. But many problems, such as collaborate data processing or data lifecycle management, can be solved more simply and more effective using data management technologies.
Key Words: databases, data, temporal, transactional, DBMS, informational systems.

[2]Associate professor, Mordovian State Pedagogical Institute after M.E.Evsevev, Saransk; sergey.lizin@gmail.com.