Kỷ yếu Hội nghị Khoa học Quốc gia lần thứ IX "Nghiên cứu cơ bản và ứng dụng Công nghệ thông tin (FAIR'9)"; Cần Thơ, ngày 4-5/8/2016
DOI: 10.15625/vap.2016.00080

SINH TỰ ĐỘNG TRIGGER TRÊN NGÔN NGỮ T-SQL HỖ TRỢ ANCHOR MODELING TRONG SQL SERVER

Nguyễn Trần Quốc Vinh[1], Huỳnh Xuân Hiệp[2], Trần Đăng Hưng[3], Hoàng Ngọc Hiển[4], Nguyễn Văn Vương[1]

[1] Trường Đại học Sư phạm, Đại học Đà Nẵng
[2] Trường Đại học Cần Thơ
[3] Trường Đại học Sư phạm Hà Nội
[4] Trường Đại học Bạc Liêu

ntquocvinh@ued.udn.vn, hxhiep@ctu.edu.vn, hungtd@hnue.edu.vn, hnhien@blu.edu.vn, nvvuong@ued.udn.vn

TÓM TẮT— *Phương pháp thao tác dữ liệu (INSERT, UPDATE, DELETE) truyền thống chỉ cho phép truy cập dữ liệu ở thời điểm hiện tại. Anchor Modeling cho phép truy cập đến dữ liệu toàn cảnh ở bất kỳ thời điểm nào trong quá khứ thông qua các khung nhìn và các hàm do người dùng định nghĩa. Các khung nhìn và hàm này sử dụng các truy vấn SELECT bao gồm phép nối ngoài và hàm gộp MAX. Về ý tưởng, khung nhìn chỉ mục hoá của SQL Server có thể hỗ trợ để cải thiện hiệu năng. Tuy nhiên, SQL Server không hỗ trợ khung nhìn chỉ mục hoá với phép nối ngoài và hàm MAX. Công trình này đề xuất nghiên cứu can thiệp mã nguồn của Anchor Modeling để (1) sinh các bảng dữ liệu chứa kết quả thực thi các khung nhìn cuối và các hàm; (2) sinh tự động các trigger trong ngôn ngữ T-SQL thực hiện cập nhật gia tăng đồng bộ kết quả được chứa trong các bảng đó. Khi đó, người dùng có thể đọc dữ liệu ở một thời điểm trong quá khứ hoặc hiện tại từ các bảng thay vì thực thi các khung nhìn và hàm. Kết quả thử nghiệm theo nhiều kịch bản khác nhau cho thấy cách tiếp cận của nghiên cứu này mang lại hiệu quả cao.*

Từ khóa— *Anchor modeling; can thiệp mã nguồn; dữ liệu trong quá khứ; sinh tự động trigger; T-SQL*

I. GIỚI THIỆU

Dữ liệu tiến hoá là dữ liệu thường xuyên có sự thay đổi về nội dung, cấu trúc, các ràng buộc, biểu diễn, nguồn gốc và tính xác thực. Cơ sở dữ liệu (CSDL) thông thường lưu trữ dữ liệu và cho phép tìm kiếm trên đó. CSDL thời gian (temporal database) là CSDL lưu trữ dữ liệu trong quá trình tiến hoá (under evolution) và cho phép tìm kiếm lịch sử trên đó. Cụ thể, CSDL thời gian cho phép truy cập đến dữ liệu ở thời điểm cuối cùng, một thời điểm bất kỳ trong quá khứ hoặc truy cập đến lịch sử thay đổi dữ liệu trong một khoảng thời gian nào đó. Dữ liệu phụ thuộc vào thời gian và thời điểm thay đổi dữ liệu phải được ghi lại.

Anchor Modeling (AM) [1; 2] kết hợp chuẩn hoá lược đồ quan hệ và tính cạnh tranh (emulation) để cung cấp kỹ thuật mô hình hoá CSDL linh hoạt (agile) cho dữ liệu tiến hoá. AM cung cấp hệ thống các nguyên tắc [3], theo đó, mô hình thu được có thể được triển khai theo mô hình dữ liệu quan hệ một cách đơn giản. Kết quả trực tiếp là CSDL thời gian đạt chuẩn 6NF.

AM cung cấp kỹ thuật mô hình hoá dữ liệu tiến hoá, công cụ mô hình hoá trực quan trong chế độ tương tác, khả năng sinh tự động các kịch bản để chuyển từ sơ đồ thiết kế được sang CSDL vật lý trong mô hình dữ liệu quan hệ. AM hỗ trợ nhiều hệ quản trị (HQT) CSDL khác nhau như Oracle, SQL Server, PostgreSQL… AM cũng sinh tự động các khung nhìn và các hàm giúp truy cập đến dữ liệu ở thời điểm cuối cùng, dữ liệu ở thời điểm bất kỳ trong quá khứ hoặc truy vấn thông tin thay đổi dữ liệu trong một khoảng thời gian hay toàn bộ quá trình thay đổi của CSDL. Các đối tượng dữ liệu đặc biệt này trên thực tế sử dụng phép nối ngoài, truy vấn lồng đồng bộ hoặc truy vấn lồng bao gồm phép gộp nhóm và hàm MAX như là bảng ảo [1; 2; 4]. Rõ ràng, các truy vấn này có thể yêu cầu tài nguyên của hệ thống rất lớn.

Khung nhìn thực (materialized view) cho phép tăng đáng kể tốc độ thực thi truy vấn [5-8], có thể được sử dụng để giải quyết vấn đề này. Tuy nhiên, khung nhìn thực còn gọi là khung nhìn chỉ mục hoá trong SQL Server không cho phép nối ngoài, phép gộp nhóm và hàm gộp MAX [8].

Nghiên cứu này thực hiện điều chỉnh mã nguồn của công cụ mô hình hoá AM để:

- Sinh các bảng thực khung nhìn cuối (BTKNC) thay vì sinh các khung nhìn cuối (latest view) khi được người dùng lựa chọn. Các BTKNC này sẽ lưu trữ kết quả thực thi của khung nhìn cuối.
- Sinh các bảng thực hàm theo thời điểm (BTHTTĐ) thay vì sinh các hàm theo thời điểm (point-in-time function) khi được người dùng lựa chọn. Các hàm theo thời điểm này sẽ lưu trữ kết quả thực thi của hàm theo thời điểm.
- Sinh tự động các bẫy sự kiện (trigger) trên tất cả các bảng liên quan để thực hiện cập nhật gia tăng đồng bộ BTKNC và BTHTTĐ ứng với mỗi giao tác thay đổi dữ liệu thông qua thao tác thêm mới bản ghi trong AM (các thao tác thêm, sửa, xoá trong CSDL thông thường) trên các bảng đó.
- Cuối cùng, xây dựng các kịch bản đánh giá được xây dựng, thực thi và so sánh giữa giải pháp đề xuất và trường hợp CSDL AM nguyên bản.

Trong phần 2, tác giả trình bày về AM, khung nhìn cuối và hàm theo thời điểm trong AM. Phần 3 bao gồm nội dung về điều chỉnh mã nguồn của hệ thống mô hình hoá AM, sinh mã các BTKNC, BTHTTĐ và các trigger

thực hiện cập nhật gia tăng đồng bộ các BTKNC và BTHTTĐ. Phần 4 bao gồm nội dung về thử nghiệm và đánh giá kết quả thu được.

II. ANCHOR MODELING

A. Mô hình hoá dữ liệu với Anchor Modeling

AM cung cấp cơ chế mở rộng không huỷ (non-destructive extensibility mechanisms), theo đó cho phép khả năng quản trị thay đổi dữ liệu rất mạnh và mềm dẻo [1]. Các thao tác dữ liệu thông thường làm cho người dùng chỉ có thể truy cập đến phiên bản dữ liệu cuối cùng, không thể truy cập đến các phiên bản dữ liệu trong quá khứ. Đặc biệt, các thao tác cập nhật (update) dữ liệu xoá giá trị cũ và thay vào đó bằng giá trị mới, thao tác xoá (delete) dữ liệu thực hiện xoá hoàn toàn bản ghi từ CSDL. Với AM, lịch sử hoá được thực hiện bằng cách sử dụng thời điểm thay đổi dữ liệu. Thời điểm này bắt đầu một khoảng thời gian kết thúc bởi việc thêm mới một bản ghi một thể hiện của thực thể cùng định danh với thời điểm thay đổi trễ hơn. Thay vì thay giá trị cũ bằng giá trị mới hay xoá bản ghi, AM giữ nguyên bản ghi cũ, thêm mới một bản ghi và đánh dấu các phiên bản bản ghi theo thời gian. Chẳng hạn, cho bản ghi có định danh ID nào đó. Bản ghi này có 3 phiên bản theo các mốc thời gian t_1, t_2, t_3. Sau này, khi cần truy cập đến dữ liệu vào thời điểm $t<t_1$, ta cần xác định tmax = MAX(t_i) với $t_i <= t$ theo ID và xác định bản ghi/giá trị ở thời điểm t theo ID và tmax.

Quan trọng hơn, các tác giả của AM đưa ra phương pháp mô hình hoá dữ liệu chính quy và độc lập với công nghệ mới. Với cách tiếp cận linh hoạt, AM cho phép tái cấu trúc các CSDL, đặc biệt kho dữ liệu, một cách dễ dàng. AM cũng cung cấp hệ thống công cụ phát triển CSDL, từ mức khái niệm với môi trường trực quan, mềm dẻo linh hoạt cho đến khả năng sinh các kịch bản để tạo CSDL vật lý trong nhiều HQT CSDL khác nhau như Oracle, SQL Server, PostgreSQL...

Nhìn chung, AM mang lại đầy đủ các lợi ích [2] của CSDL đạt dạng chuẩn 6NF: i) Quản lý và xử lý được dữ liệu tiến hoá, đảm bảo tính toàn vẹn của dữ liệu theo thời điểm bất kỳ; ii) tăng vòng đời của CSDL so với vòng đời trung bình 5 năm; iii) đơn giản hoá các khái niệm mô hình hoá (modeling concepts), giảm thiểu hoá lỗi mô hình hoá; iv) cho phép phát triển theo mô-đun và tăng trưởng từng bước; v) chuyển đổi sang CSDL vật lý đơn giản; vi) cho phép và hỗ trợ công cụ sinh tự động các kịch bản tạo CSDL vật lý cũng như các khung nhìn, các hàm cần thiết; vii) đặc biệt, chỉ quét những dữ liệu cần thiết trong trong quá trình xử lý truy vấn nhờ mỗi cột được lưu trữ riêng lẻ trong từng bảng; viii) không chấp nhận NULL nên khắc phục được vấn đề dữ liệu thưa thớt trong CSDL thông thường. Nếu dữ liệu được lưu trữ theo từng thuộc tính trong từng bảng riêng biệt, khi cần truy vấn dữ liệu từ nhiều bảng ta phải thực hiện phép nối. Phép nối và phép gộp nhóm là hai phép toán đắt đỏ nhất của truy vấn chọn dữ liệu trong mô hình dữ liệu quan hệ. Tuy nhiên, theo các thử nghiệm trên lượng dữ liệu lớn (hàng chục GB với hàng chục triệu bản ghi) [1], cho thấy việc phát triển CSDL quan hệ theo mô hình AM không gây ra vấn đề. Đạt được điều này khả năng là nhờ chi phí tài nguyên tiết kiệm được nhờ không phải quét qua dữ liệu liên quan các thuộc tính không cần thiết cho truy vấn đủ bù đắp cho phần thực thi phép nối.

Ngoài ra, AM cho hiệu năng tốt nhờ khai thác các các đặc điểm hạn chế của CSDL dữ liệu thông thường [2], i) trên thực tế nhiều thuộc tính chấp nhận giá trị NULL và nhiều trường thường chứa giá trị NULL trong CSDL; ii) thường số lượng giá trị khác biệt nhỏ hơn rất nhiều so với tổng số lượng các giá trị dữ liệu; iii) các giá trị cho khoá hoặc từ định danh chỉ chiếm phần nhỏ so với toàn bộ dữ liệu; iv) có nhiều khoá; v) có nhiều thuộc tính; vi) truy vấn chọn lựa dữ liệu thường chỉ sử dụng một vài trường trong toàn bộ bản ghi chứa nhiều giá trị.

Các khái niệm nền tảng trong mô hình bao gồm neo (anchor), giới hạn (knot), thuộc tính (attribute), thuộc tính hằng (static attribute), thuộc tính biến thiên (historized attribute), thuộc tính hằng giới hạn (knotted static attribute), thuộc tính biến thiên giới hạn (knotted historized attribute). Các loại mối quan hệ (tie) bao gồm quan hệ hằng (static tie), quan hệ biến thiên (historized tie), quan hệ hằng giới hạn (knotted static tie), quan hệ biến thiên giới hạn (knotted historized tie) [1].

Trong mối tương quan với mô hình thực thể - mối quan hệ rất phổ biến, có thể hiểu, anchor là tập thực thể. Anchor biểu diễn ID của thực thể. Giới hạn đại diện cho một tập lực lượng nhỏ các giá trị cố định theo thời gian, thường là những giá trị dùng chung cho nhiều thực thể khác nhau. Chẳng hạn, giới hạn giới tính có các giá trị "nam", nữ". Thuộc tính biểu diễn các thuộc tính của tập thực thể, cụ thể là của anchor. Thuộc tính hằng là những thuộc tính chấp nhận giá trị không thay đổi theo thời gian, chẳng hạn ngày sinh của một công dân. Thuộc tính biến thiên là thuộc tính chấp nhận các giá trị có thể thay đổi theo thời gian và cần ghi lại sự thay đổi này, chẳng hạn, đơn vị công tác của một công dân. Thuộc tính hằng giới hạn biểu diễn mối quan hệ giữa thực thể và giới hạn, cụ thể thuộc tính chấp nhận một giá trị cố định nào đó từ tập số lượng nhỏ các giá trị; chẳng hạn, công dân có mã công dân CMT1 có giới tính nam. Thuộc tính biến thiên giới hạn biểu diễn thuộc tính của một thực thể có thể chấp nhận các giá trị khác nhau từ tập số lượng nhỏ các giá trị; chẳng hạn, công dân có thể được xếp vào các nhóm tuổi khác nhau theo thời gian.

Mối quan hệ mô tả liên kết giữa hai hoặc nhiều tập thực thể anchor với nhau, chính xác hơn, là loại mối liên kết giữa hai hay nhiều thực thể với nhau, và cũng có thể là với các giới hạn. Có thể hiểu mối quan hệ hằng là mối quan hệ không thay đổi theo thời gian, chẳng hạn, mối quan hệ cha – con sinh học giữa hai anchor công dân. Nó cũng có thể là ví dụ cho mối quan hệ hằng giới hạn, mối quan hệ sinh học giữa hai công dân: cha-con, mẹ-con, ông-cháu, bà-cháu, cô/dì/chú/bác – cháu và không quan hệ huyết thống. Mối quan hệ biến thiên giới hạn đại diện cho mối liên kết chấp nhận các giá trị khác nhau theo thời gian từ một tập số lượng nhỏ các giá trị giữa hai hoặc nhiều thực thể với nhau.

Chẳng hạn, mối quan hệ bạn bè, người yêu, vợ chồng giữa hai công dân với nhau. Mối quan hệ biến thiên là mối liên kết có thể chấp nhận các giá trị khác nhau theo thời gian từ tập số lượng lớn các giá trị.

AM phân biệt ba quan điểm về thời gian [4]. Thời gian thay đổi (changing time) dữ liệu đối với các thuộc tính biến thiên hoặc mối quan hệ biến thiên là khoảng thời gian các giá trị của chúng hoặc mối quan hệ là hợp lệ trong lĩnh vực ứng dụng đang được mô hình hoá. Trong mô hình anchor, thời gian thay đổi được mô tả thông qua một cột trong bảng, có tên gọi ValidTime và có kiểu dữ liệu ngày/giờ. Thứ hai, thời điểm xảy ra (happening time) đại diện cho thời điểm một sự kiện xảy ra trên thực tế trong lĩnh vực ứng dụng. Thời điểm này tự thân nó sẽ là một thuộc tính trong mô hình anchor. Thứ ba, thời gian ghi nhận (recording time) là dữ liệu về thời điểm thông tin được ghi nhận. Nghiên cứu này quan tâm hơn đến hai loại thời gian đầu tiên.

B. Truy cập dữ liệu

Ràng buộc của lĩnh vực ứng dụng như sau. Có nhiều cơ sở sân khấu văn nghệ nơi các tiết mục có thể được biểu diễn. Các chương trình sẽ xác định các tiết mục thực tế sẽ được biểu diễn trên sân khấu nào bởi nghệ sĩ nào. Sân khấu có hai thuộc tính, tên và địa điểm. Tên của sân khấu có thể thay đổi theo thời gian, nhưng địa điểm thì không. Chương trình chỉ có tên của nó là thuộc tính. Nghệ sĩ có tên là nghệ danh và có thể thay đổi theo thời gian. Họ có giới tính, và giả sử giới tính của mỗi nghệ sĩ là không thay đổi. Kỹ năng nghề nghiệp của mỗi nghệ sĩ có thể thay đổi theo thời gian và được ghi lại. Nghệ sĩ sẽ được chấm điểm cho mỗi tiết mục họ đã biểu diễn. Mỗi sân khấu có một chương trình riêng tại một thời điểm và tất nhiên chương trình ở mỗi sân khấu có thể thay đổi theo thời gian. Quan hệ huyết thống cha – con, mẹ - con giữa hai nghệ sĩ cũng được ghi lại. Một tiết mục là một sự kiện gắn liền với một vị trí và thời gian liên quan đến một sân khấu mà chương trình đang diễn ra bởi một hoặc nhiều nghệ sĩ. Người ta cũng quan tâm đến lượng khách ở mỗi tiết mục cũng như tổng doanh thu từ tiết mục đó.

Sau khi thiết kế mô hình Anchor trên công cụ Anchor Modeler, người dùng có thể sinh ra các đối tượng dữ liệu liên quan, cụ thể trong trường hợp thử nghiệm là trong SQL Server. Các khung nhìn cuối và hàm theo thời điểm được sinh ra như trên Hình 1 và Hình 2. Khung nhìn cuối cho phép truy cập đến dữ liệu ở thời điểm hiện tại theo thời gian thực, thời điểm lớn nhất. Hàm theo thời điểm cho phép truy cập đến dữ liệu ở thời điểm trong quá khứ được nhập vào là đối số của hàm, dữ liệu bao gồm các giá trị mới nhất ở thời điểm đó và thường nhỏ hơn giá trị thời điểm hiện tại theo thời gian thực.

```
1  create view lAC_part_PR_in_RAT_got as
2  select
3    [AC_PR_RAT].AC_ID_part,
4    [AC_PR_RAT].PR_ID_in,
5    [AC_PR_RAT].RAT_ID_got,
6    [RAT].RAT_Rating,
7    [AC_PR_RAT].AC_part_PR_in_RAT_got_ValidFrom
8  from
9    AC_part_PR_in_RAT_got [AC_PR_RAT]
10   left join
11   RAT_Rating [RAT]
12   on
13     [RAT].RAT_ID = [AC_PR_RAT].RAT_ID_got
14 where
15   [AC_PR_RAT].AC_part_PR_in_RAT_got_ValidFrom = (
16   select
17     max(sub.AC_part_PR_in_RAT_got_ValidFrom)
18   from
19     AC_part_PR_in_RAT_got [sub]
20   where
21     [sub].AC_ID_part = [AC_PR_RAT].AC_ID_part
22       and
23     [sub].PR_ID_in = [AC_PR_RAT].PR_ID_in
24         );
```

Hình 1. Khung nhìn cuối

```
1  create function pAC_Actor (@timepoint date) returns
   table
2  return select
3    [AC].AC_ID,
4    [AC GEN].GEN_ID,
5    [GEN].GEN_Gender,
6    [AC_NAM].AC_NAM_Actor_Name,
7    [AC_NAM].AC_NAM_ValidFrom
8  from
9    AC_Actor [AC]
10   left join
11   AC_GEN_Actor_Gender [AC_GEN]
12   on
13    [AC_GEN].AC_ID = [AC].AC_ID
14   left join
15   GEN_Gender [GEN]
16   on
17    [GEN].GEN_ID = [AC_GEN].GEN_ID

18   left join
19   AC_NAM_Actor_Name [AC_NAM]
20   on
21    [AC_NAM].AC_ID = [AC].AC_ID
22   and
23    [AC_NAM].AC_NAM_ValidFrom = (
24   select
25     max([sub].AC_NAM_ValidFrom)
26   from
27    AC_NAM_Actor_Name [sub]
28   where
29     [sub].AC_ID = [AC].AC_ID
30       and
31      [sub].AC_NAM_ValidFrom <= @timepoint
32    );
```

Hình 2. Hàm theo thời điểm

Dễ nhận thấy các khung nhìn và hàm trên Hình 1 và Hình 2 bao gồm truy vấn lồng đồng bộ. Các truy vấn tạo khung nhìn và hàm này có thể được viết lại dưới dạng bao gồm truy vấn lồng như là bảng ảo, được xem là hiệu quả hơn truy vấn lồng đồng bộ. Trong mọi trường hợp, các truy vấn lồng này đều chứa hàm gộp nhóm và hàm gộp MAX. Mỗi khi truy cập đến khung nhìn cuối hoặc hàm theo thời điểm, các truy vấn trên Hình 1 và Hình 2 sẽ được thực thi, chắc chắn sẽ yêu cầu rất lớn đến tài nguyên của hệ thống.

III. GIẢI PHÁP ĐỀ XUẤT

A. Khung nhìn thực cho khung nhìn cuối, hàm theo thời điểm

Khung nhìn thực như là một dạng cache đặc biệt trong CSDL quan hệ. Khung nhìn thực lưu trữ kết quả thực thi truy vấn vào các BTKNC và BTHTTĐ như các bảng thông thường. Khi cần truy cập khung nhìn hoặc hàm, nếu kết quả đã được lưu trong khung nhìn thực thì kết quả đó sẽ được dùng để trả lời truy vấn của người dùng thay vì thực hiện truy vấn trên cơ sở tạo khung nhìn và hàm. Tất nhiên, tốc độ sẽ rất cao và yêu cầu tài nguyên hệ thống rất thấp, ngược lại với thực thi truy vấn từ đầu. Bao giờ cũng phải trả giá cho những lợi ích mang lại. Một khi dữ liệu trong các bảng liên quan thay đổi thì cũng cần cập nhật các BTKNC và BTHTTĐ để kết quả thực thi truy vấn được chứa trong nó nó trở nên là đúng đắn. Một số công trình [9-11] nghiên cứu các phương pháp phân tích hệ thống thông tin, phân tích CSDL để lựa chọn các truy vấn để xây dựng khung nhìn thực thông qua các tính toán phức tạp. Trên thực tế, việc lựa chọn truy vấn để thực hoá khung nhìn có thể dựa trên kinh nghiệm của quản trị viên hệ thống thông tin.

Nghiên cứu đề xuất lưu kết quả thực thi của khung nhìn cuối và hàm theo thời điểm vào trong các bảng khung nhìn thực cho các trường hợp lượng dữ liệu lớn, tần suất truy cập cao hoặc yêu cầu kết quả thực thi trong thời gian thực; nghĩa là khi lợi ích mang lại lớn hơn nhiều so với chi phí phải trả về mặt hiệu năng hoặc khi yêu cầu về tốc độ truy cập dữ liệu khung nhìn cuối hoặc hàm theo thời điểm là cấp thiết, theo thời gian thực. Trong phạm vi nghiên cứu này, người dùng sẽ quyết định lựa chọn khung nhìn cuối và hàm theo thời điểm để xây dựng khung nhìn thực thông qua lựa chọn cấu hình trong thời gian thiết kế mô hình CSDL bằng AM. Việc cập nhật các BTKNC và BTHTTĐ được thực hiện thông qua các trigger được sinh tự động cho mỗi sự kiện thêm mới, cập nhật bản ghi trong các bảng liên quan.

B. Sinh trigger thực hiện cập nhật

Với AM trong SQL Server, chỉ cần tạo và cập nhật bảng khung nhìn thực của khung nhìn cuối và hàm theo thời điểm khi có các sự kiện trên các bảng được sinh ra từ thuộc tính biến thiên, thuộc tính biến thiên giới hạn, quan hệ biến thiên và quan hệ biến thiên giới hạn. Một khi đã có các bảng khung nhìn thực là kết quả thực thi truy vấn bao gồm phép nối ngoài và hàm gộp MAX, có thể dùng công cụ có sẵn của SQL Server để tạo các khung nhìn chỉ mục hoá khác.

Cho mỗi bộ giá trị của khoá, trong trường hợp thử nghiệm trên Hình 1 và Hình 2 là ([AC_PR_RAT].AC_ID_part, [AC_PR_RAT].PR_ID_in) và ([AC].AC_ID), truy vấn lồng sẽ tìm được một giá trị MAX(ValidFrom) và từ đó xác định được giá trị cuối cùng của thuộc tính cho đến thời điểm ValidFrom.

Theo quan điểm của AM, thực tế trong CSDL (truyền thống) chỉ có thể hai thao tác dữ liệu thêm mới (insert) và cập nhật (update). Để thêm mới một bản ghi, tương ứng là một giá trị cho một thuộc tính của thực thể, trong bảng CSDL AM sẽ được thêm mới một bản ghi. Với các thuộc tính và quan hệ biến thiên có thể tồn tại thao tác cập nhật. Để cập nhật một bản ghi, tương ứng cho một thuộc tính của thực thể từ giá trị cũ về giá trị mới, một bản ghi mới sẽ được thêm vào bảng CSDL AM và bản ghi với giá trị cũ vẫn được giữa nguyên.

Có thể thực hiện cập nhật bảng khung nhìn thực cho khung nhìn cuối như sau. Cho mỗi bản ghi từ tập bản được thêm mới vào bảng gốc (bảng tham gia vào truy vấn tạo khung nhìn cuối): i) xác định khoá; ii) xác định bộ giá trị khoá và giá trị thuộc tính; iii) cập nhật giá trị thuộc tính trong bảng khung nhìn thực với điều kiện bộ giá trị khoá trùng với bộ giá trị khoá trong bản ghi mới.

Với các BTHTTĐ, cần bổ sung kiểm tra và thực hiện quá trình cập nhật tương tự như cho BKNC nếu tham số thời điểm đầu vào của hàm lớn hơn hoặc bằng giá trị ValidFrom.

Thuật toán cập nhật khung nhìn thực đơn giản nên thuật toán sinh mã nguồn các trigger triển khai các thuật toán đó cũng đơn giản. Cho mỗi bảng được sinh ra từ thuộc tính biến thiên, thuộc tính biến thiên giới hạn, quan hệ biến thiên và quan hệ biến thiên giới hạn có tham gia vào truy vấn tạo khung nhìn thực: i) xác định các trường, kiểu dữ liệu; ii) sinh phần đầu (header) của trigger; iii) sinh các biến trung gian; iv) sinh mã thực hiện cập nhật khung nhìn thực; v) sinh mã kết thúc trigger. Trên **Hình 3** là thuật toán sinh trigger cập nhật BTHTTĐ. Cho trường hợp BTKNC sẽ không bao gồm sinh mã kiểm tra ở dòng 10 **Hình 3**. Đầu vào của thuật toán là mô hình CSDL AM. Đầu ra là các trigger thực hiện cập nhật BTHTTĐ và BTKNC trên mỗi sự kiện thêm mới trong các bảng được sinh ra từ các thuộc tính trong mô hình AM (dòng 3).

```
1  foreach thực thể trong danh sách do
2    if thực thể khác rỗng then
3      foreach thuộc tính trong danh sách thuộc
                                  tính của thực thể do
4        if thuộc tính khác rỗng then
5          Sinh mã SQL xóa trigger nếu đã tồn tại
6          Sinh mã phần đầu trigger sự kiện thêm mới
7          Sinh mã khai báo biến
8          Sinh mã cursor cho bảng inserted
9          foreach bản ghi từ cursor do
10           if ValidFrom <= TimePoint then
11             if thuộc tính là loại knot then
12               Sinh mã SELECT lấy giá trị knot
13             end if
14             if tồn tại bản ghi trong BTHTTĐ cho thuộc
                                  tính của thực thể tương ứng then
15               Sinh lệnh cập nhật BTHTTĐ
16             else
17               Sinh lệnh thêm mới bản ghi vào BTHTTĐ
18             end if
19           end if
20         end foreach
21         Sinh lệnh đóng cursor
22         Sinh phần kết thúc trigger
23       end if
24     end foreach
25   end if
26 end foreach
```

Hình 3. Sinh trigger cập nhật BTHTTĐ

Trong SQL Server, mỗi lệnh SQL thực hiện thêm mới, cập nhật một tập các bản ghi trong các bảng liên quan. Vì thế, thuật toán sinh mã nguồn vòng lặp duyệt qua từng bản ghi trong tập này (dòng 8, 9).

Mã nguồn thực hiện sinh bảng khung nhìn thực được bổ sung vào tệp mã nguồn của AM CreateAnchorPerspectiveLatest.js. Mã nguồn thực hiện sinh các triggers thực hiện cập nhật gia tăng đồng bộ các khung nhìn thực được bổ sung vào tệp mã nguồn CreateAttributeTrigger.js. Tác giả đã sử dụng các đối tượng của các lớp từ thư viện AM phục vụ cho việc sinh mã bao gồm schema, anchor, attribute. Các biến được sử dụng bao gồm $anchor.capsule, $anchor.name, $anchor.mnemonic, $anchor.identityColumnName, $attribute.anchorReferenceName, $anchor.identity, $attribute.dataRange, $attribute.changingColumnName, $attribute.timeRange, $attribute.knotValueColumnName, $knot.dataRange, $attribute.knot.identity, $attribute.valueColumnName.

IV. THỬ NGHIỆM VÀ ĐÁNH GIÁ

Quá trình thực nghiệm được tiến hành theo các bước như sau: i) Cài đặt hệ thống thực nghiệm; 2) xây dựng xây dựng CSDL theo phương pháp AM; iii) sinh dữ liệu và cho vào các CSDL thực nghiệm; iv) sinh khung nhìn cuối, hàm theo thời điểm và đo đạt; v) sinh các BTKNC, BTHTTĐ, sinh các trigger và tiến hành đo đạt; vii) tiến hành so sánh và đánh giá kết quả thu được ở bước iv) và bước v).

A. Hệ thống thử nghiệm

Hệ thống thử nghiệm là máy tính xách tay trên nền Windows 10, HQT CSDL SQL Server 2014; phần cứng bao gồm CPU Intel Core i5 4 lõi 1.7GHz, RAM 4 GB DDR3 và SSD Intel 240 GB. Hệ thống được cài đặt trong chế độ máy chủ CSDL chuyên biệt. Điểm hạn chế của hệ thống là chỉ dùng một ổ cứng cho cả phần dữ liệu chính thức và bộ nhớ tạm của HQT CSDL. Microsoft yêu cầu cấu hình thử nghiệm đối với HQT CSDL phải bao gồm ít nhất 2 ổ cứng, dữ liệu chính thức và bộ nhớ tạm phải được cấu hình trên hai ổ cứng khác nhau để đảm bảo tốc độ thực thi truy vấn.

B. Dữ liệu thử nghiệm

Công trình [1] đã xây dựng được bộ dữ liệu, kịch bản khoa học, phong phú với 288 tổ hợp dựa trên 4 kịch bản khác nhau và đã tiến hành thử nghiệm so sánh hiệu năng giữa CSDL quan hệ thông thường và CSDL Anchor trong SQL Server. Nghiên cứu này kế thừa kịch bản số 4 từ nghiên cứu đó, sinh CSDL xây dựng các bảng thực và trigger để tiến hành thực nghiệm và so sánh thời gian thực thi truy vấn trên khung nhìn cuối và hàm theo thời điểm đối với trường hợp số phiên bản tối đa của mỗi thể hiện là 2. Khung nhìn cuối và hàm theo thời điểm dựa trên truy vấn không bao gồm điều kiện chọn lựa trong mệnh đề WHERE.

Bảng 1. Cấu hình sinh dữ liệu và thử nghiệm

Tham số				
(a)	Số phiên bản của mỗi thể hiện (bản ghi)	2		
(b)	Mức độ thưa thớt của dữ liệu	50%		
(c)	Tỉ lệ giới hạn so với thuộc tính	1/3		
(d)	Số lượng thuộc tính của mô hình	12		
Các tổ hợp thử nghiệm				
(e)	Số lượng bản ghi	100 ngàn	1 triệu	10 triệu
(f)	Số lượng thuộc tính được truy vấn	1	Một nửa	Tất cả

C. Kết quả và đánh giá

Tổ hợp các cấu hình từ (e) và (f) cho ra 9 trường hợp thử nghiệm có kết quả như trên Bảng 2. Các CSDL có kích thước từ 25MB đến 11.6GB. Đối số cho các hàm theo thời điểm được chọn làm sao để có thể lựa chọn khoảng 85% số lượng bản ghi so với khung nhìn cuối.

Bảng 2. Thời gian thực thi truy vấn

	Truy vấn (s)																	
	Khung nhìn cuối									**Hàm theo thời điểm**								
	1	2	3	4	5	6	7	8	9	1	2	3	4	5	6	7	8	9
AM	0,2	0,6	0,9	1,9	5,4	8,8	14,1	55,2	114,2	0,19	0,7	1,1	1,9	5,7	9,1	14,3	59,2	146,1
KNT	0,1	0,2	0,2	3,6	3,8	4,1	6,5	6,8	6,9	0,1	0,1	0,2	3,1	3,9	4,0	6,1	6,2	6,4

Thời gian cần thiết để thực thi truy vấn truy cập đến khung nhìn cuối và hàm theo thời điểm cao gấp hàng chục lần so với thời gian cần thiết để truy cập đến các bảng khung nhìn thực. Với mô hình gồm 12 thuộc tính trong cấu hình thử nghiệm, AM cho CSDL bao gồm 13 bảng. Khi truy vấn tất cả các thuộc tính, 13 bảng này phải được nối với nhau bằng phép nối ngoài. Nếu dùng KNT, chỉ cần truy cập đến một bảng duy nhất.

Có thể thấy rằng, với số lượng bản ghi lớn đến 10 triệu, số lượng thuộc tính 12 có thể tạo nên CSDL AM có kích thước lớn đến 11,6GB nhưng thời gian cần thiết để truy cập đến khung nhìn cuối và hàm theo thời điểm đều rất thấp. Tốc độ cao này có thể đạt được nhờ CSDL đạt dạng chuẩn 6 và đặc biệt là cấu hình hệ thống thử nghiệm dựa trên ổ cứng SSD cho tốc độ rất cao. Thời gian cần thiết để thêm mới một bản ghi vào các bảng AM trong trường hợp có mặt các trigger cập nhật KNT và không có trigger đều rất thấp, xấp xỉ 0ms. Đây cũng là điểm hạn chế của nghiên cứu này.

Thông thường các HQT CSDL hoạt động trên các hệ thống bao gồm HDD có tốc độ truy cập thấp hơn hàng chục lần so với SSD. Nếu thử nghiệm được thực hiện trên hệ thống bao gồm HDD thì có thể các số liệu thu được phản ánh thực tế tốt hơn.

V. KẾT LUẬN

Anchor Modeler là một công cụ mô hình hoá CSDL cho phép tạo các mô hình dữ liệu thời gian sử dụng 6NF. CSDL AM lưu lại tất cả các phiên bản dữ liệu, cho phép truy cập đến dữ liệu ở bất kỳ thời điểm nào trong quá khứ với tốc độ truy cập rất cao. Các thực nghiệm với gần 300 kịch bản khác nhau trên SQL Server cho thấy tốc độ cao hơn so với CSDL đạt 3NF được thiết kế thông qua mô hình thực thể - mối quan hệ được sử dụng rộng rãi.

Nghiên cứu này thực hiện sinh các BTKNC và BTHTTĐ lưu kết quả thực thi các khung nhìn cuối và các hàm thực theo thời điểm khi người dùng lựa chọn. Các thực nghiệm với 9 kịch bản khác nhau trên lượng dữ liệu lớn từ 100 ngàn đến 10 triệu bản ghi cho thấy tốc độ truy cập của giải pháp đề xuất cao hơn hàng chục lần so với giải pháp nguyên bản của AM.

Về ý tưởng, có thể sử dụng khung nhìn chỉ mục hoá của SQL Server để đảm bảo dữ liệu được chứa trong các BTKNC và BTHTTĐ luôn được đúng đắn đáp ứng bất kỳ thay đổi dữ liệu nào trong các bảng AM. Tuy nhiên, SQL Server không hỗ trợ khung nhìn chỉ mục hoá trên cơ sở các truy vấn bao gồm phép nối ngoài và hàm MAX. Hệ thống các trigger được sinh tự động trên tất cả các sự kiện thay đổi dữ liệu. Các trigger thực hiện cập nhật gia tăng đồng bộ các BTKNC và BTHTTĐ. Chênh lệch thời gian thực hiện thêm bản ghi giữa trường hợp có trigger và không có trigger là không đáng kể.

TÀI LIỆU THAM KHẢO

[1]. Rönnbäcka L., Regardtb O., Bergholtzc M., Johannessonc P., Wohedc P., "Anchor modeling — Agile information modeling in evolving data environments", *Data & Knowledge Engineering*, 69(12), 2010, pp. 1229.

[2]. Rönnbäck L., "Anchor Modeling – A Technique for Information under Evolution", <http://www.anchor modeling.com/wp-content/uploads/2011/05/Anchor-Modeling-Open-AMW.pdf> (Truy cập: 03/06/2016).

[3]. R¨onnb¨ack L., Regardt O., Bergholtz M., Johannesson P., Wohed P., "From Anchor Model to Relational Database", <http://www.anchormodeling.com/wp-content/uploads/2010/09/AM-RDB.pdf> (Truy cập: 20/4/2013).

[4]. Rönnbäck L., "Three Concepts of Time in Anchor Models", <http://www.anchormodeling.com/wp-content/uploads/2010/08/Three_Concepts_of_Time_in_Anchor_Models.pdf> (Truy cập: 17/5/2016).

[5]. Nguyễn Trần Quốc Vinh, "Viết lại truy vấn để sử dụng khung nhìn thực có hàm thống kê trong PostgreSQL", *Kỷ yếu HTKH quốc gia Nghiên cứu cơ bản và ứng dụng công nghệ thông tin*, FAIR-VIII-2015, 2015, trang 760-767.

[6]. Nguyễn Trần Quốc Vinh, Trần Trọng Nhân, "Nghiên cứu xây dựng mô-đun sinh tự động mã nguồn trigger trong ngôn ngữ C thực hiện cập nhật gia tăng, đồng bộ các khung nhìn thực trong PostgreSQL", *HTKH Quốc gia về Nghiên cứu cơ bản và ứng dụng Công nghệ thông tin (FAIR)*, VII-2014, 2014, pp. 440-448.

[7]. Nica A., "Incremental maintenance of materialized views with outerjoins", *Inf. Syst.*, 37(5), 2012, pp. 430-442.

[8]. Microsoft, "Create Indexed Views (SQL Server 2016)", <https://msdn.microsoft.com/en-us/library/ms191432.aspx?f=255&MSPPError=-2147217396> (Truy cập: 27/5/2016).

[9]. Gupta H., Mumick I. S., "Selection of views to materialize in a data warehouse", *Knowledge and Data Engineering, IEEE Transactions on*, 17(1), 2005, pp. 24-43.

[10]. Кунгурцев А. Б., Нгуен Ч. К. В., "Анализ возможности применения материализованных представлений в информационных симтемах ", *Тр. Одесск. политехн. ун–та, Украина*, 2(20), 2003, pp. 102-106.

[11]. Кунгурцев А. Б., Нгуен Ч. К. В., "Метод анализа информационной системы для применения материализованных представлений ", *Холодильна Техніка і Технология. Одеса, Украина*, 2(94), 2005, pp. 102-105.

AUTOMATIC TRIGGER GENERATING IN T-SQL FOR SUPPORTING OF ANCHOR MODELING IN SQL SERVER

Nguyen Tran Quoc Vinh, Huynh Xuan Hiep, Tran Dang Hung, Hoang Ngoc Hien, Nguyen Van Vuong

ABSTRACT*— Relational database created using entity-relationship model with traditional methods of data manipulation allows to access only to the last instances of data. Anchor Modeling database allows to access to data in any points of times through views and used defined functions, including latest data. Those views and functions are created based on the select queries with outer joins and aggregate function MAX. Ideally, materialized views can be used to support and improve the performance. SQL Server does not allow to create indexed views with outer joins and MAX function. This research suggests to intervene the source codes of Anchor Modeler for (1) automatic generating the materialized view tables that contain query result of latest views and point-in-time functions; (2) automatic generating the triggers on all data manipulating events for all under-lying tables for synchronized incremental updating of materialized views. Therefor, users can access to latest data or data at any past time point through those tables instead of executing the views and functions. The experiments show the positive effect of the suggested approach.*

Keywords*— Anchor modeling; source code intervention; historized data; automatic trigger generating; T-SQL.*