제18회 한국정보처리학회 추계학술발표대회 논문집 제9권 제2호 (2002.11)

분산 데이터베이스 환경에서 고객 관리를 위한 실체화된 뷰 유지 방법론

이현창*
*경인여자대학 전산정보시스템학과
e-mail:hclee@kic.ac.kr

AMaterialized View Maintenance Methodology for Customer Management in a Distributed Database Environment

Hyun-Chang Lee*
*Dept of Computer Information System, Kyung-In Women's College

요 약

일반적으로 고객 관리를 위한 고객 데이터는 운영 시스템 환경 여건상 다양한 분산 데이터베이스 시스템에 저장되어 있다. 이와 같이 분산 저장된 데이터들로부터 고객들의 향후 경향이나 추세 분석 등 의사 결정에 필요한 데이터로 활용하고자 할 때는 데이터베이스에 저장된 대량의 데이터가 고객 분석에 적합한 형태로 구성되어 서비스 되어야 한다. 이에 적절한 구조가 데이터 웨어하우스 구조이며, 데이터 웨어하우스는 분산 저장된 각각의 소스들로부터 발생된 변경 정보들을 실시간으로 데이터 웨어하우스에 반영되어야한다. 이렇게 함으로써 정확한 의사 결정을 수행할 수 있게 된다. 이에 본 논문에서는 분산 컴퓨팅 환경에서 고객 관리를 정확하고 효과적으로 이루어질 수 있도록 기본 소스에서 발생된 데이터 변경을 웨어하우스에 실시간으로 전달하여 정확한 데이터를 유지할 수 있는 방법론을 제시한고자 한다. 또한 제시된 방법의 실험 평가 결과를 간략하게 도시하여 나타내었다.

1. 서론

오늘날 컴퓨팅 환경은 제한된 범위 내에서 서로 유용한 기능을 제공하고 있으나 네트워크 상에서는 서로 독립적으로 운용되고 있기 때문에 효율적인 상호 운용이 이루어지지 않고 있다[1]. 이로 인하여 각 업체들은 분산환경에서 고객 서비스 증대와 효과적인 고객 관리를 위한 분석을 위해서 시스템 구축 및 운용에 많은 비용이 요구되고 있으며, 이를 극복하기 위한 대안으로 데이터 웨어하우스 구축이 필요하게 되었다.

데이터 웨어하우스는 주제 지향적(subject oriented)이며, 통합적(integrated)이며, 시간 변화(time varying)와 비휘발성(non volatile) 데이터 집합인 뷰(view)로서 주로 조직의 의사 결정이나 데이터 마이닝 등의 질의와 분석을 지원하는데 사용된다[4]. 근래에는 데이터 웨어하우스 데이터 집합인 실체 뷰(materialized view)가 더욱 중요하게 다루어지고 있다[2].

실체 뷰에 대한 대부분의 연구는 뷰 생성에 사용된 소스 테이블에 변경이 일어날 때 실체 뷰를 점진적으로 변경하는 기법이다[3]. 실체 뷰에 관한 기존 연구 방법의 환경에서 각 소스는 실체 뷰에 관한 관리 방법을 알고있으며, 질의에 따른 실체 뷰와의 관련성을 알고 있다는 가정 하에서 출발한다. 그러나 데이터 웨어하우스 환경에서 소스는 뷰에 관한 정보를 알지 못한다[4]. 이로 인하여 소스의 변경이 뷰에 바로 적용될 수 없으며, 데이터 웨어하우스의 부정확한 실체 뷰의 원인이 된다. 그러므로 실체 뷰와 소스를 정확하게 일치될 수 있도록 최신 정보로 변경해야 하는 문제가 발생하게 되며[6] 이를 해결하

기 위한 방법이 필요하게 되었다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 분산 환경에서 통합되는 방법과 실체 뷰에 관한 연구 방법에 관하여 살펴본다. 제 3장에서는 본 논문에서 제시하는 분산 환경하에서 실체 뷰 유지 관리 방법에 대해 설명한다. 4장에서는 제안된 내용에 대한 결과 분석과 5장에서 결론으로 맺는다.

2. 관련연구

일반적으로 소스에서 발생된 갱신 정보가 실체 뷰에 정확하게 전달되기 위해서는 실체 뷰와 소스사이에 몇 단계를 거치게 된다[3,4]. 첫째로, 정확한 실체 뷰를 유지하기 위해서 소스는 발생한 갱신 정보를 웨어하우스 측에 알린다. 둘째, 웨어하우스에서는 소스에서 보내온 정보가 실체 뷰를 유지하는 다른 관련 테이블들과의 죠인 관계성을 알아보기 위해서 다시 소스에 질의를 보낸다. 셋째, 소스에서는 웨어하우스로부터 보내온 질의가 실체 뷰와 관련된 다른 소스 테이블들과 관련성이 존재하는지 검사해 보아야 한다. 넷째, 상기 과정이 평가 단계이며, 평가된 결과는 실체 뷰 유지를 위해서 웨어하우스에 보내게 될 내용(answers)이며, 실체 뷰는 이를 바탕으로 정확한 데이터를 유지할 수 있게 된다. 다음 그림 1.에 각 단계를 그림으로 도시하였다.

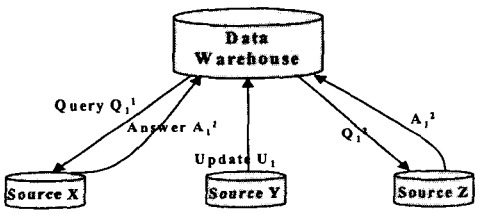

그림 1. 분산 시스템환경에서 뷰 유지 단계

분산 시스템 환경에서 고객 관리를 위한 두 개 이상의 시스템들을 통합하는 경우 중요하게 고려되어야 할 사항으로 사용자에게 분산 시스템 환경이라는 사실을 인지하지 않아도 충분히 원격지 시스템의 정보를 이용하여 원하는 결과를 도출할 수 있도록 정확하고 신속한 처리를 해주어야 하는 것이다.

실체 뷰를 정확하게 유지하기 위한 기존의 방법으로 실체 뷰를 재계산하는 방법이다. 이 방법은 과도한 통신 비용이 발생하기 때문에 어려우며 이에 대한 해결 접근 방법은 [5]에서 찾을 수 있다. 분산 환경하에서 대표적인 방법으로는 Strobe 알고리즘과 Sweep 알고리즘을 들수 있다[5,6]

3. 실체 뷰 유지를 위한 방법

먼저, 본 실체 뷰 유지 방법론에 관한 설명은 알고리즘을 설명하기보다 이해를 돕기 위해서 [3]에서처럼 소스와 웨어하우스에서 발생하는 사건으로 나누었듯이 본 연구에서도 사건(event) 위주로 설명한다. 소스에서 발생되는 갱신 정보는 다른 사이트와의 관련성 여부에 따라 실체 뷰인 웨어하우스에 반영되는지를 결정해야 한다. 그렇기 때문에 각 사이트에 갱신 정보와 관련된 질의처리를 하여야 하며, 소스측에서 발생되는 사건들과 웨어하우스 측에서 발생되는 사건들의 처리 내용이 서로 상이하게 처리되어진다. 다음은 사건들의 처리 종류를 살펴본다.

소스와 웨어하우스에서 발생되는 사건들은 각각 원자성을 가지며, 하나의 사건 내에서는 순서대로 처리가 이루어진다고 가정하며, 그림 1을 참조한다.

■ 데이터 소스측에서 발생되는 사건들

- S_req_i 는 발생한 갱신(update i) 정보 U_i를 수행한후 실체 뷰에 반영하기 위해서 갱신 정보를 웨어하우스로 보낸다.
- S_eva_i 는 웨어하우스로부터 양방향으로 보내어진 질의를 받은 소스에서 소스에 존재하는 기존의 기본 릴레이션을 이용하여 질의 평가를 수행하여 임시 결과 릴레이션을 생성한다. 생성된 임시 릴레이션을 다시 웨어하우스에 보낸다.

상기와 같은 사건들이 소스측에서 발생하며 다음은 웨어하우스 측에서 발생되는 사건들이다.

■ 데이터 웨어하우스에서 발생되는 사건들

- W_vie_i 는 소스에서 보내온 갱신 정보 U_i를 받아서 갱신 정보 리스트에 등록한 후 갱신 정보 U_i에 대해 질의 Q_i^1, Q_i^2를 생성한다. 생성된 질의는 소스 사이트 i를 기준으로 다른 양방향으로 보내어 질의 결과를 기다린다.

· W_res_i 는 W_vie_i 사건을 처리하기 위해서 보내진 질의Q_i^1, Q_i^2 에 대한 결과 릴레이션 A_i^1, A_i^2를 받아서 조인하며, 조인된 결과를 실체 뷰에 반영하도록 한다. 또한 갱신 정보 리스트에서 갱신정보 U_i를 삭제한다.

소스에서 발생된 갱신 정보는 동시성을 위해서 웨어하우스에 발생 순서대로 갱신정보 리스트에 저장된다. 또한, 본 논문에서 사용되는 실체 뷰에 대한 정의는 다음 V에서처럼 [4]와 같이 정의하였으며. 표현은 다음과 같다.

$$V = \prod_{proj} (\sigma_{cond} (r1 \ x \ r2 \ x \ldots x \ rn))$$

proj는 애트리뷰트 이름들의 집합이며, cond는 불린(boolean) 수식이며, r_1, r_2, ... , r_n은 서로다른 테이블 리스트이다[4].

발생된 갱신 정보들의 순서를 유지하기 위한 리스트는 실체 뷰 유지 알고리즘인 [3]에서 다루어졌던 소스/웨어하우스 사건 발생에 따른 점진적 알고리즘을 바탕으로 이루어졌다. 본 논문에서의 갱신 처리는 상기 사건들의 순서와 같다. 단지, 웨어하우스에서 임시 결과 릴레이션을 받았을 때 갱신 정보 순서 리스트에서 처리중인 갱신 정보보다 이전에 발생한 갱신정보가 존재하면 불일치가 발생하게 되므로 대기 상태가 되며, 이전에 발생된 갱신정보가 존재하지 않는다면 연산을 계속 수행하게 된다. 수행된 연산 정보는 리스트에서 삭제되며, 수행은 완료된다.

상기와 같은 결과를 얻기 위해서 순서적으로 발생 사건들을 유지하기 위한 저장 장치 "SEQ" 를 사용하며, SEQ는 소스에서 발생한 모든 갱신 정보를 발생 순서대로 포함하며 순서화(serializability)를 유지함으로서 발생되는 사건들의 발생순서에 따른 결과값이 정확하게 얻을 수 있도록 한다. 또한 여러 테이블에 대해 갱신 연산 결과가 정확하게 유지될 수 있도록 한다.

4. 성능 분석

이장에서는 본 논문에서 제안된 고객 정보의 정확하고 효율적인 관리를 통한 의사결정을 위해서 기존 실체 뷰 유지 방법과 본 연구와의 비교를 통한 분석을 제시한다.

본 논문에서 제시하는 SEQ 유지 알고리즘에 대한 성능평가를 위해서 기존에 다중 소스 데이터 웨어하우스에서 점진적으로 뷰 유지 수행 알고리즘으로 잘 알려진 Strobe 알고리즘과 Strobe의 제약 사항을 많이 완화시켜서 좋은 성능을 보였던 Sweep 알고리즘들과 성능 비교를 수행한 결과 다음과 같은 결과를 얻었다.

알고리즘	구조	갱신에따른 메시지비용	특성
ECA	중앙집중	O(1)	원격보상처리 기하급수적인메시지크기
Strobe	분산 환경	O(n)	키 유지 요구
SWEEP	분산 환경	O(n)	DW에서 지역적으로보상 서버의 로드 증가
SEQ	분산 환경	O(n/2)	분산질의를 통한 메시지비용감소

각 알고리즘별 구조는 보상 알고리즘의 경우 중앙집중식 환경에서 수행되었으며, 갱신에 따른 메시지 비용은 O(1)이었다. Strobe 알고리즘은 분산 환경에서 적용가능한 유지 알고리즘으로서 O(n)의 갱신에 따른 메시지 비용 결과가 있었다. 그러나 Strobe 알고리즘에서는 키를 요구하며 기타 다른 부분에서도 많은 제약 사항이 따랐다. Sweep 알고리즘에서는 Strobe의 단점을 보완하여 키등의 제약사항을 제거하면서도 갱신에 따른 메시지비용을 유지하였다. 본 논문에서 제시한 알고리즘에서는 각 소스에 보내진 질의를 양분화 시킴으로서 메시지 크기와 처리 시간을 감소시킴으로서 빠른 응답 처리시간을 가져왔다.

5. 결론

오늘날 네트워크 환경과 기술 발전에 힘입어 제한된 범위 내에서 서로 유용한 기능들을 제공하고 있는 기존의 하드웨어와 소프트웨어들을 네트워크 상에서 서로 독립적이 아닌 통합적으로 운영하여 고객의 요구에 최적화된 서비스의 필요성이 증가하고 있다. 이에 대해 본 연구에서는 분산된 환경에서 독립적으로 처리되어지는 고객 데이터를 분석에 적합한 구조인 실체 뷰에 정확하게 유지될 수 있도록 소스에서 발생된 데이터의 즉각적(immediate) 반영 방법으로

신속하고 정확하게 적용할 수 있는 방법을 제시하였다. 이와 같은 방법의 제공은 소스데이터와 실체 뷰 사이에 발생할 수 있는 불일치를 제거함으로서 정확한 데이터로 고객 서비스를 수행할 수 있을 것으로 기대된다.

참고문헌

[1] A. Leinwand and K. F. Conroy, "Network Management" Addison-Wesley Publishing Company, Inc. pp. 17-36, 1996.

[2] Latha S.Colby, Akira Kawaguchi, Daniel F.Lieuwen, Inderpal Singh Mumick and Kenneth A. Ross. Supporting Multiple View Maintenance Policies. In Proceedings of the ACM SIGMOD International Conference on Management of Data pages 405-416. 1997.

[3] J. A. Blakeley, P. Larson, and F. W. Tompa. Efficiently Updating Materialized Vews. Proceedings of ACM SIGMOD 1986 International Conference, pages 61-71, Washington, D.C., May 1986.

[4] Y. Zhuge, H. Garcia-Molina, J. Hammer, and J. Widom. View Maintenance in a Warehousing Environment. Proc. of the ACM SIGMOD Conference, pages 316-327, San Jose, California, May 1995.

[5] D. Agrawal, A. El Abbadi, A. Singh, T. Yurek. Efficient View Maintenance at Data Warehouses. Proceedings of ACM SIGMOD 1997.

[6] Y. Zhuge, H. Garcia-Molina, Janet L. Wiener. The Strobe Algorithms for Multi-Source Warehouse Consistency. In Proceedings of the International Conference on Parallel and Distributed Information Systems, December 1996.