韓國 OA學會 論文誌
第3卷 第4號, 1998. 12.

98-3-4-1-1

형성 뷰를 이용한 효율적인 시간지원 질의 처리 기법

정 경 자*

Efficient Temporal Query Processing using Materialized View

Kyeong Ja Jeong*

요 약

시간지원 데이터베이스는 시간의 흐름에 따라 변화된 자료를 모두 저장하므로 질의 처리 시스템은 많은 양의 정보를 처리하여야 한다. 본 연구에서는 시간지원 질의 처리 시스템에서 질의의 관련성 검사 알고리즘을 이용한 효율적인 질의 처리 기법을 제시한다. 질의 관련성 검사 알고리즘은 사용자가 입력한 베이스 릴레이션에 대한 검색 질의와 시스템 카달로그에 저장된 뷰 정의의 실행 트리와의 질의 관련 여부를 조사하고, 뷰 정의와 관련된 검색 질의는 뷰에 관한 실행 트리로 변형하는 질의 변형 과정을 수행한다. 그 결과 시간지원 질의 처리 시스템에서는 질의 대상이 되는 튜플의 수를 감소시켜 질의 처리의 성능을 향상시킬 수 있다.

Abstract

Temporal Databases store all of informations by time varying, so the temporal query processor has to process very large information. Therefore, we propose an efficient method of query processing by using the relevance checking algorithm of input query and view definition. The relevance checking algorithm of query investigates relevance between the input query of user about base relation and the execution tree of view definition stored in system catalog. And related input query with view definition have a process of the query translation to the execution tree of view. So temporal query processor is able to increase performance of query processor by reducing the number of tuple.

* 충청대학 멀티미디어과 조교수
논문접수 : 98.11.2. 심사완료 : 98.12.3.

I. 서론

시간지원 데이터베이스에서는 시간의 변화에 따른 자료의 이력 사항을 모두 저장하므로 질의 처리의 대상이 되는 자료가 매우 방대하다[1,2,3]. 그러므로 시간지원 데이터베이스 관리 시스템에서는 자료를 보다 효율적으로 처리하기 위한 기법이 요구된다. 본 연구에서는 형성 뷰를 이용한 시간지원 질의 처리 시스템의 효율성을 향상시키고자 한다.

뷰는 베이스 릴레이션으로부터 유도된 릴레이션으로 데이터베이스의 일반 사용자가 생성할 수 있는 릴레이션이며 뷰를 생성하는 방법에 따라 비형성 뷰(unmaterialized view)와 형성 뷰(materialized view)로 나눌 수 있다[4,8]. 비형성 뷰는 뷰에 대한 테이블을 생성하지 않고 뷰 정의만을 시스템 카달로그에 저장하여 뷰에 대한 질의가 입력될 경우 이를 베이스 릴레이션에 대한 질의로 변형하여 수행한다. 그러나, 형성 뷰는 자주 접근되는 시점의 자료를 베이스 릴레이션과 동일하게 뷰에 대한 스키마 정보, 상태 정보, 테이블 값을 소유하여 질의 처리의 대상이 되는 자료의 수를 감소시켜 질의 처리의 효율성을 향상시킬 수 있다[2,3].

형성 뷰를 이용한 효율적인 질의 처리 기법은 형성 뷰 구성시 뷰의 실행 트리를 시스템 카달로그에 저장하여 사용자가 입력한 검색 질의와 뷰의 실행 트리와의 질의 관련성 검사를 수행한다. 그 결과 베이스 릴레이션에 대한 검색 질의가 뷰의 실행 트리와 관련된 질의인 경우 검색 질의를 뷰에 대한 질의로 변형하여 대상이 되는 튜플의 수를 감소시켜 질의 처리의 성능을 향상시킬 수 있다.

본 연구를 수행하므로서 시간지원 질의 처리 시스템에서 얻을 수 있는 기대 효과는 다음과 같다. 첫째, 자주 접근되는 시점의 자료를 형성 뷰로 관리하므로 질의 처리 대상이 되는 수를 줄일 수 있다. 둘째, 형성 뷰의 실행 트리를 시스템 카달로그에 저장하여 사용자가 입력한 검색 질의는 이 실행 트리와의 질의 관련성 검사를 수행하여 베이스 릴레이션에 대한 질의를 뷰에 대한 질의로 변형하여 질의 처리 대상이 되는 자료의 수를 줄여 질의 처리의 효율성을 향상시킬 수 있다.

논문의 효율적인 구성을 위해 II장에서는 형성 뷰의 관련 연구를 설명하며 제 III장에서는 시간지원 데이터베이스에서 형성 뷰 구성을 설명하고 제 IV장에서는 형성 뷰를 효율적인 시간지원 질의 처리 시스템을 설명한다. 제 V장에서는 질의 예로 질의 관련성 검사의 효율성을 설명하고 마지막으로 제 VI장에서는 현재 수행된 연구를 마무리하며 결론을 맺는다.

II. 관련 연구

베이스 릴레이션은 데이터베이스 관리자의 스키마 구성을 통해 만들어진 릴레이션이다. 그러나, 뷰는 이미 생성된 릴레이션 또는 또다른 뷰에 의해 유도되는 유도 릴레이션으로 일반 사용자가 조작 가능한 릴레이션이다[5].

뷰의 분류는 뷰 정의에 의한 질의 처리 결과를 저장할 것인지의 여부에 따라 두가지로 나누어진다. 첫째, 비형성 뷰 기법으로 뷰에 대한 실제 테이블 값을 소유하지 않고 뷰에 대한 질의가 입력되면 이를 베이스 릴레이션에 대한 질의로 변형하여 처리하는 기법이다[6,7]. 둘째, 뷰는 베이스 릴레이션과 동일하게 뷰의 스키마, 테이블 등을 유지하는 형성 뷰가 있다. 과거의 데이터베이스 환경에서는 시스템의 성능 문제로 비형성 뷰를 많이 사용하였으나 최근 하드웨어 가격의 저렴화와 대용량 메모리의 출현으로 형성 뷰 기법에 관한 연구가 활발하다[7,8].

형성 뷰 기법은 다음과 같은 조건이 만족되는 데이터베이스 환경에서 효율적인 뷰 관리 기법이 된다[6].

① 뷰에 대한 질의의 수가 베이스릴레이션에 대한 질의의 수보다 많은 경우
② 베이스릴레이션의 크기가 클 경우
③ 뷰를 구성 조건의 선택요소(selectivity factor)가 적은 경우
④ 뷰에 대한 베이스릴레이션의 변경 연산이 적은 경우
⑤ 뷰에 대한 질의가 대부분 검색 연산인 경우

형성 뷰에 관한 연구는 뷰를 구성하는 방법과 베이스

릴레이션의 변경 정보를 뷰에 효율적으로 전파하는 뷰의 갱신 기법에 대한 연구가 대부분이다. 최근에는 형성 뷰를 이용한 응용분야에 대한 연구가 이루어지고 있다.

형성 뷰의 구성과 뷰 갱신에 관한 연구는 다음과 같다. Roussopoulos[7]는 뷰 테이블 구성을 색인으로 저장하여 기억장소의 절약과 뷰 갱신 속도를 향상시키고자 하였다.

Hanson[3]은 질의 수정화, 즉시 뷰 유지, 그리고 지연 뷰의 유지 관리에 대한 관계 대수적 표현과 이들의 성능 비교를 수행하였다. 지연 뷰 유지와 즉시 뷰 유지에 관한 SPJ(Select-Project-Join) 수식의 관계 대수적 표현과 뷰 구성시 중복 문제를 처리하기 위해 counting 기법을 사용하였다.

Blakeley[14]는 관계 대수 표현으로 뷰 정의와 뷰 갱신 연산을 처리하도록 하였다. 베이스 릴레이션에 변경 연산이 발생하였을 경우 이 변경 연산이 뷰에 영향을 주는지 여부를 검사하는 알고리즘을 구성하여 변경 정보를 뷰에 전파시 불필요한 연산을 제거하기 위해 입력 질의와 뷰 정의를 이용하여 관련성 검사를 통해 갱신에 대한 과부하를 줄였다.

Horwitz & Teitelbaum[9]은 속성 문법을 이용하여 관계형 데이터베이스에서 프로그램 언어 기반의 규칙 생성에 대한 모델을 이용한 뷰 갱신에 대한 알고리즘을 제시하였다.

형성 뷰를 이용한 응용 분야에 대한 연구는 다음과 같다. Zhou, Zhuge, Gupta[12,11,2] 등은 데이터 웨어하우징에서 다중 데이터베이스에 상주된 객체들을 수집하여 객체 통합을 위해 형성 뷰를 이용하였다.

Levy[8]는 cojunctive 뷰 정의로 conjunctive 질의의 응답 문제를 뷰를 이용하여 처리하였고, Chaudhuri[1]는 형성 뷰를 이용하여 집계 질의(aggregate query)의 최적화를 시도하였다.

또한 Lu[16]는 데이터 통합을 위한 모델 제시를 통하여 원격지와 근거리 릴레이션을 통합한 뷰에서 부분적으로 형성 뷰를 이용하였다.

이들 연구 분야는 기존의 데이터베이스 환경에서 뷰 처리에 관한 연구와 새로운 데이터베이스 분야에 형성 뷰를 적용한 경우에 대한 것이다. 그러므로 본 연구에서는 이들 연구를 기반으로 시간지원 데이터베이스에서 형성 뷰를 이용한 질의 효율성 향상측면에 대한 연구가 된다.

시간지원 데이터베이스 시스템에서는 베이스 릴레이션이 크기가 시간의 흐름에 따라 증가하고, 베이스 릴레이션에 대한 갱신 연산이 거의 발생하지 않으므로 형성 뷰 기법으로 뷰를 관리하는 것이 적합하다[6,10].

III. 시간지원 형성 뷰 구성

3.1 개요

시간지원 데이터베이스에서는 시간의 변화에 따라 생성되는 자료를 모두 저장하므로 데이터베이스에서 처리할 자료의 양이 방대해진다. 그러므로 접근 빈도가 높은 자료를 형성 뷰로 구성하여 질의 처리에 드는 비용을 줄일 수 있다. 형성 뷰로 뷰를 유지 관리하기 위해서는 베이스 릴레이션의 변경 정보를 뷰에 전파하는 전략과 뷰 자체적인 갱신 여부에 관한 관리 기법이 요구된다. 그림 1은 시간지원 형성 뷰의 관리 기법이 된다.

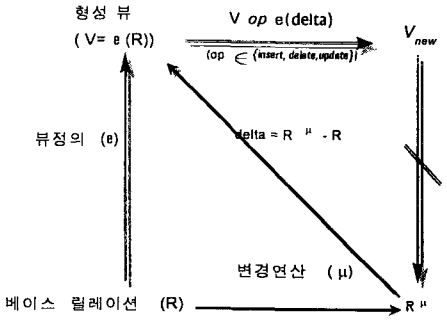

그림 1. 형성 뷰 관리
Fig. 1 Management Materialized View

[그림 1]과 같이 뷰 V는 베이스 릴레이션 R로 부터 뷰 정의식 e에 의해 유도되는 릴레이션으로 식(1)과 같이 표현하며, 베이스 릴레이션의 변경 연산 μ는 뷰의 변경을 요구한다. 변경 연산에 의해 변경된 베이스 R 릴레이션 R^{μ}와 변경전 베이스 릴레이션 의 차분 정보는 전파연산 $p\rightarrow$에 의해 뷰의 갱신을 유도하며 식(2)와 같이 표현된다.

$$V = e(R) \qquad \text{..... 식(1)}$$

$$|R - R^{\mu}|_{p\rightarrow} V_{new} \qquad \text{..... 식(2)}$$

형성 뷰는 비형성 뷰와는 다르게 실제 뷰에서 값을 소

유하고 있으므로 베이스 릴레이션의 변경 정보가 뷰에 전파되어 뷰의 갱신을 수행하여야 하므로 갱신 연산이 많이 발생하는 응용 분야에서는 비효율적이다. 또한 베이스 릴레이션의 변경 정보를 뷰에 전파하기 위한 방법이 요구되는데 본 연구에서는 뷰 구성 단계에서 만들어진 뷰의 실행 트리를 시스템 카달로그에 저장하여 뷰 갱신에 이용하며 이 실행 트리를 질의 최적화 단계에 질의 관련성 검사에 이용된다. 뷰 갱신에 관한 내용은 본 논문의 범위를 벗어나므로 구체적으로 기술하지는 않는다.

3.2 형성 뷰 지원을 위한 의미 확장

데이터베이스에서 뷰를 지원하기 위해서는 기존의 릴레이션에 뷰에 관한 의미 확장이 요구된다. 뷰는 사용자 측면에서 생성할 수 있는 릴레이션으로 베이스 릴레이션과 동일하게 취급될 수 있다. 시간지원 질의 처리에서는 형성 뷰를 지원하기 위해 뷰에 관한 스키마 정보와 뷰의 상태 정보, 실제 값을 소유한다.

시간지원 데이터베이스에서 뷰는 베이스 릴레이션의 의미를 확장하여 표현할 수 있다. 뷰는 베이스 릴레이션과 동일하게 시간 값의 소유 여부에 따라 구분될 수 있다. 사건이 발생된 시간인 유효시간을 소유하는 이력 뷰와 시스템에 저장된 시간을 소유하는 거래시간 뷰 그리고 두 시간 값을 모두 소유한 이원시간 뷰와 시간 값을 소유하지 않는 스냅 뷰로 구분된다. 시간지원 뷰의 의미 표현은 다음과 같다[15].

뷰 = [{스냅,거래시간,이력,이원시간}×거래시간×]$^{+}$
[뷰스키마×거래시간]*×
[[스냅상태×거래시간]+
[이력상태×거래시간]]*×
[뷰의실행트리]$^{+}$

뷰의 의미 표현에서와 같이 뷰의 종류는 시간지원 여부에 따라 4종류로 분류되며 뷰는 베이스 릴레이션과 동일하게 뷰의 스키마 정보와 뷰의 실제 값 및 뷰의 실행 트리를 저장한다. 뷰의 실행 트리는 사용자가 입력한 검색 질의와의 질의 관련성 검사에 이용된다. 또한 베이스 릴레이션의 변경 정보를 뷰에 전파하는 과정에서도 뷰의 실행 트리가 사용된다.

3.3 형성 뷰의 실행 트리

형성 뷰의 실행 트리는 뷰 정의에 관한 관계 대수식으로 질의 최적화 과정을 거쳐 생성된 관계 수식으로 시스템 카달로그에 저장하게 된다.

형성 뷰의 실행 트리의 각 노드는 뷰를 생성하는데 사용되는 연산자인 추출(μ), 선택(σ), 유효시간(δ), 롤백(ρ), 조인($\bowtie$) 및 교차곱($\times$)의 연산자로 구성된다. 이들 연산자 중에서 유도 연산자와 롤백 연산자는 시간 연산을 처리하기 위한 것으로 유도 연산자는 사건이 발생된 시간을 나타내는 유효시간을 위한 연산자이며 롤백연산자는 자료가 데이터베이스에 수록된 거래시간을 처리하기 위한 연산자가 된다.

실행 트리를 구성하는 시간지원 관계 대수 연산자에서는 기존의 관계 연산자에는 존재하지 않는 값의 동등성을 검사하는 과정이 추가된다. 값의 동등성 검사는 임의의 튜플들이 시간 값을 제외한 모든 속성이 동일한 지를 검사하는 과정이며 그 결과에 따라 시간 연속적인 튜플의 경우 튜플을 하나로 결합하며 그렇지 않을 경우 별도로 유지된다.

Ⅳ. 형성 뷰를 이용한 시간지원 질의 처리

4.1 개요

뷰는 베이스 릴레이션의 정보에서 임의의 조건에 맞는 정보로 구성되므로 뷰를 구성하는 튜플은 베이스 릴레이션에 비해 그 양이 매우 적다. 특히, 조인 연산이 포함된 질의를 뷰로 구성할 경우 질의 처리 비용을 많이 감소시킬 수 있다.

시간지원 질의 처리 부분에서 뷰의 이용은 뷰 정의를 시스템 카달로그에 저장하여 입력된 질의가 이미 생성된 뷰 정의와 일치할 경우 질의의 실행 트리를 해당 뷰에 관한 실행 트리로 변형하므로 질의 처리 비용을 줄일 수 있다. 일반적인 질의 처리 과정은 질의의 구문 및 의미분석 단계, 실행트리 생성 단계, 질의 최적화, 그리고 실행 단계로 구성된다. 본 연구에서는 질의 최적화 과정에서 입력된 질의의 뷰의 관련성 검사 단계의 추가로 입력 질의가 뷰와 관련된 질의인 경우 질의의 실행 트리를 뷰에 관한 실행 트리로 변형하여 질의 처리 비용을 감소하고자

한다. 뷰를 이용한 변형된 시간지원 질의 처리 과정은 [그림 2]와 같다.

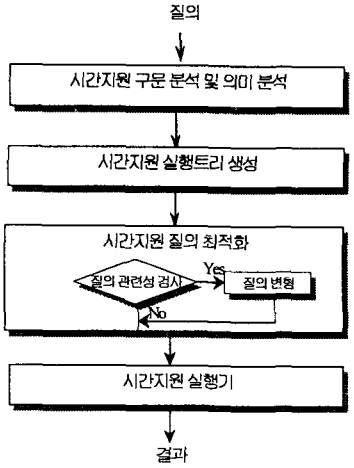

그림 2. 시간지원 질의 처리 과정
Fig. 2 Temporal Query Processing

[그림 2]와 같이 시간지원 질의 처리 과정은 사용자가 질의를 입력하면 입력된 질의의 구문 및 의미 분석을 수행하고 정확한 질의인 경우 시간 관계 대수 연산자로 구성된 시간지원 실행 트리를 생성한다. 시간지원 실행 트리는 시간지원 질의 최적화 과정에서 질의 처리 비용을 최소화할 수 있는 질의 실행 트리로 변형되며, 이 과정에서 입력된 질의가 뷰와 관련된 질의 여부를 검사하는 과정이 수행된다.

[그림 2]의 시간지원 질의 처리 과정에서 뷰 정의에 대한 질의의 실행 트리는 검색 질의들 중에서 뷰와 관련된 질의 여부를 검사하고 베이스 릴레이션의 변경 연산을 뷰에 전파하기 위해 뷰 정의 규칙 테이블인 RuleOfView에 기록된다.

4.2 뷰 정의 규칙 테이블

뷰에 대한 실행 트리는 뷰 생성시 뷰에 관한 규칙을 테이블에 저장하며 이 테이블을 RuleOfView라 표현한다. RuleOfView 테이블은 질의 관련성 검사에 이용된다. 질의 관련성 검사는 사용자가 입력한 검색 질의의 수식이 저장된 뷰 정의와 일치하는 지의 여부를 검사하는 단계이며 이 결과 질의 관련성이 있을 경우 입력 질의를 뷰에 대한 질의로 변형하는 질의 변형 단계를 수행하여 질의 처리의 효율성을 높일 수 있다.

뷰 정의 규칙 테이블인 RuleOfView 테이블은 표 1과 같으며 이 테이블에 들어가는 정보는 "베이스릴레이션명", "뷰이름", "뷰수식"으로 구성된다. "뷰수식"정보는 뷰에 대한 일반조건, 추출변수, 유효시간 조건 및 거래시간 조건이 기록되며 이 정보는 베이스 릴레이션의 변경 정보를 뷰에 전파하기 위한 뷰 갱신에서도 이용된다.

표 1 뷰정의 규칙 테이블
Table 1. Rule Table of View Definition

베이스 릴레이션명	뷰 이름	뷰 수식			
		일반 조건 (σ)	추출변수 (π)	유효시간 조 건	거래시간조건
직 급 릴레이션	조 교 수	직급="조교수"	교원번호, 이름,소속	NULL	NULL
	승진_1996	NULL	교원번호, 이름,소속	overlap,1996	NULL
급 여 릴레이션	고수입급여	급여액>=450	교원번호, 소속,급여액	overlap,1997	NULL

[표 1]과 같이 뷰 정의에 대한 모든 정보를 RuleOfView에 저장하여 검색 질의 입력시 입력된 질의에 해당하는 릴레이션이 뷰를 소유한 경우 뷰 정의 테이블인 RuleOfView에 기록된 정보를 이용하여 질의 관련성 검사를 수행한다.

4.3 질의 관련성 검사

질의 관련성 검사는 입력된 질의가 베이스 릴레이션에 의해 유도된 뷰와 관련된 질의인지의 여부를 검사하는 단계로 베이스 릴레이션의 검색 질의를 뷰에 관한 검색 질의로 변형하므로서 검색 대상이 되는 자료의 수를 줄이고자 한다. 이 단계는 검색문의 조건과 뷰 정의시 뷰 실행 트리에서 추출한 뷰의 규칙 테이블인 RuleOfView 테이블을 참조하여 질의 관련 여부를 검사한다.

입력된 질의가 뷰와 관련된 질의인지의 여부 검사는 디스크에 저장된 뷰를 참조하지 않고 입력된 질의와 뷰 정의 테이블을 사용하여 검사하게 된다. 질의 관련성은 입력된 검색 질의를 관계 대수식으로 변형하여 뷰 정의 테이블인 RuleOfView 테이블에 저장된 뷰 정의와 비교하여 수행하게 된다.

다음의 정의는 뷰 정의 테이블인 RuleOfView의 뷰 정의 규칙을 표현한 것이다.

【정의】 뷰 정의 규칙은 V는

$$V = Q(R_N A, P, V, T)$$로 구성된다.

① $R_{N,N=1,i}$은 뷰의 베이스 릴레이션들을 나타낸다.

② A는 뷰를 구성하는 일반 속성 리스트

③ P는 뷰를 구성하기 위한 일반속성에 대한 조건식이 된다

$$P \in x(I_i)\ op\ x(I_j) \vee x(I_i)\ op\ c \vee x(I_i)\ op\ y(M_j)$$

(단, x,y는 뷰에 대한 베이스 릴레이션을 나타내고 Ii, Ij, 그리고 Mj는 베이스 릴레이션에 한 속성이며 c는 상수 값이 된다)

$$op \in \{=, >, <, \geq, \leq, \neq, \text{and}, or, not\}$$

④ V는 뷰를 구성하는 유효시간 조건을 의미한다

$$V \in \{x(v_i)\ op\ y(v_j) \vee x(v_i)\ op\ c\}$$
$$op \in \{overlap,\ equal,\ precede\}$$

⑤ T는 뷰를 구성하는 거래시간 조건식을 의미한다

$$T \in \{x(t_i)\ op\ y(t_j) \vee x(t_i)\ op\ c\}$$
$$op \in \{overlap,\ equal,\ precede\}$$

위의 정의에 의해 질의 관련성 검사 알고리즘은 다음과 같다.

```
query_relationship(view_tree, inputQuery_tree)
{
/* 뷰정의와 입력 질의에서 프로젝션 노드를 추출 */
attrOfview = projectionNode of view definition;
attrOfinput_query = projectionNode of input query;
/* 뷰 정의와 입력 질의에서 선택 노드 추출 */
condOfview = selectionNode of view defintion;
condOfinput_query = selectionNode of input query;
/* 뷰 정의와 입력 질의에서 유효시간 값 추출 */
validOfview = derivationNode of view defintion;
validOfinput_query = derivationNode of input query;

/* 뷰 정의와 입력 질의에서 거래시간 값 추출 */
transOfview = rollbackNode of view definition;
transOfinput_query = rollbackNode of input query;

/* step 1 */
if ( (attrOfinput_query + condOfinput_query) ∉
attrOfview) then return False;

/* step 2 */
for(each of attribute of condOfinput_query)
{
/* 입력 질의 선택 노드에 속한 속성과 뷰 정의의 동일
속성을 추출하여 조건 수 검사 */
NumberOfattributeView = count(attribute);
attributeOfInput =
attributeOfMatch(condOfinput_query);
NumberOfattributeInput=count(attributeOfInput);
if (NumberOfattributeView <
NumberOfattributeInput) then return FALSE;
/* 조건 연산이 동등 연산이 아닌 경우 뷰 정의와 입력 질의의 조
건 범위 검사 */
if(opOfattribute && opOfattributeOfInput ∈
{<, <=, >, >=} )
if(Range(attribute) < Range(attributeOfInput))
then return False;
}
/* step 3*/
/* 뷰 정의와 입력 질의의 유효시간 범위 검사 */
if(durationOfvalid(validOfview) <
durationOfvalid(validOfinput_query)) then
return FALSE;
/* 뷰 정의와 입력 질의의 거래시간 범위 검사 */
if(durationOftrans(transOfview) <
durationOftrans(transOfinput_query)) then
return FALSE;
return TRUE;
}
```

그림 3. 질의 관련성 알고리즘
Fig. 3 Algorithm of Query relationship

그림 3의 질의 관련성 검사 알고리즘에서 step 1에서는 뷰 정의에 속한 속성과 검색 질의의 속성에서 검색 질의의 속성이 뷰 속성에 포함되었는지를 검사한다. step 2에서는 뷰 정의에 속한 임의의 속성에 대한 조건의 수가 검색 질의의 조건의 수와 비교하며 조건이 동등 조건이 아닌 경우 뷰 정의에 속한 속성의 범위가 검색 질의의 범위보다 큰지를 검사한다. 마지막으로 step 3에서는 뷰 정의와 검색 질의의 유효시간과 거래시간에 대한 범위를 검사하는 단계가 된다. 세 개의 단계를 거쳐 TRUE가 반환된 경우 입력된 검색 질의는 뷰 정의에 대한 질의로 변형할 수 있다.

4.4 질의 변형

질의 변형 단계는 4.3절의 질의 관련성 검사 알고리즘인 "query_relationship"의 수행 결과 TRUE로 판정된 질의에 대한 베이스 릴레이션을 뷰로 변형하는 단계로 시

간지원 질의 처리 시스템의 실행기는 이 변형된 실행 트리를 사용하여 질의 결과를 출력하게 된다.

질의 변형을 수행하므로서 얻을 수 있는 이점은 뷰는 베이스 릴레이션의 정보보다 상대적으로 적은 양을 소유하므로 디스크 접근 횟수를 줄여 질의 최적화를 유도할 수 있다. 특히, 죠인 연산 또는 교차 곱이 포함된 실행 트리에 대하여 뷰를 이용할 경우 질의 처리 비용이 절약된다.

V. 질의 관련성 검사의 질의 효율성 평가

이 논문에서는 검색 질의와 뷰 정의와 질의 관련성 검사를 수행하기 위해 시스템 카달로그에 저장된 뷰의 실행 트리를 사용한다. 입력된 검색 질의가 뷰 정의와 관련된 경우 질의 대상이 되는 자료의 수를 줄이므로 전체 질의 처리 비용을 줄일 수 있다.

다음 예제 질의를 베이스 릴레이션을 이용한 검색과 뷰를 이용한 검색을 수행하였을 경우

```
range of s is 급여
range of f is 직급
  retrieve(교원번호=s.교원번호,
        급여액=s.급여, 직급=f.직급)
    valid from begin of(f overlap s)
        to end of(f overlap s)
    where s.교원번호=f.교원번호
        and s.급여>=450
    when s overlap f
```

위의 검색문에 대한 관계 대수식은 다음과 같다.

$$\pi_{s.교원번호,s.급여,f.직급}(\delta_{(s\ overlap\ 1996)\ overlap\ f}(\sigma_{s.급여\geq 450}(급여 \bowtie 직급))) \quad \cdots 식(3)$$

위의 검색 질의에서 직급릴레이션과 급여릴레이션이 뷰를 소유하는 지를 뷰정의 규칙 테이블에 검사한다. 그 결과 급여릴레이션이 뷰를 소유하고 있고 뷰 정의 규칙이 식(4)와 같을 경우 릴레이션과 뷰 정의에 대한 질의 관련성 검사를 수행한다.

V고수입급여=Q(급여, {교원번호, 소속, 급여액}, {급여액≥350},{(overlap, 1996)}, NULL) ..... 식(4)

① 일반속성의 조건 수
- 조건식 수 :

N_v(급여액≥350) = N_q(급여액≥450)

즉, 형성 뷰의 조건과 입력 검색 질의 조건의 수가 1로 동일

② 속성 수
- 입력 질의 속성(QA) = {교원번호, 소속, 급여액}
- 뷰 정의 속성(VA) = {교원번호, 소속, 급여액}

QA ⊆ VA이므로 2번째 조건도 만족한다.

③ 일반 조건식의 범위
- $Rang_q$(급여액≥450)≤$Rang_v$(급여액≥350)

으로 뷰 정의의 조건 범위가 검색 질의 조건 범위가 더 크다.

④ 유효 시간 조건
- 입력 검색질의 유효시간 조건과 뷰 정의의 유효시간 조건이 모두 1996년에 발생한 사건을 나타내므로 두 조건이 일치

⑤ 거래시간 조건
- 거래시간 조건에 대한 명시가 없으므로 현재 유효한 버전에서 검색

위 예의 조건 검사에서는 ①②③④⑤의 모든 조건이 만족되므로 뷰의 질의 수식으로 변형이 가능하다.

그림 3의 알고리즘에서 정의한 질의 관련성 검사에 의해 식(3)을 뷰에 관한 관계 대수식인 식(5)로 변형하여 질의 처리를 수행한다. 식(5)에서 "h"는 뷰에 대한 튜플 변수가 된다.

$$\pi h.교원번호,h.급여,f.직급(\delta(h\ \ overlap\ \ 1996)\ overlap\ f(\sigma h.급여\geq 450(고수입급여 \bowtie 직급))) \quad 식 \cdots\cdots (5)$$

질의 변형으로 급여 릴레이션의 대상 튜플 수가 n이고 직급 릴레이션의 튜플 수가 m이며 뷰의 비율이 10%라 가정할 경우 식 (3)의 연산 대상이 되는 튜플 수는 n*m이고 고수입급여인 뷰로 변형된 식(5)는 n*m*0.1로 대

상이 되는 튜플 수가 축소되므로 입출력 비용이 감소되며 CPU 연산 비용도 감소된다.

VI. 결론

시간지원 데이터베이스에서는 시간의 흐름에 따라 변경되는 자료의 이력 사항을 모두 기록하므로 질의 처리 시스템에서 처리해야 할 자료가 매우 방대해진다. 그러므로 접근 빈도가 높은 뷰를 형성 뷰로 구성하여 베이스 릴레이션의 참조 없이 빠르게 데이터에 접근하여 질의 처리의 성능을 향상시킬 수 있다.

본 연구에서는 시간지원 질의 처리 시스템에서 형성 뷰를 이용한 질의 처리의 효율성을 향상시켰다. 형성 뷰를 구성하므로서 질의 처리의 향상은 다음과 같다. 첫째, 접근 빈도가 높은 자료를 뷰를 구성하므로 뷰와 관련된 질의에 응답 시간을 단축할 수 있다. 둘째, 형성 뷰 구성시 뷰에 대한 정의문을 시스템 카달로그에 저장하여 사용자가 검색 질의를 입력하였을 경우 해당 질의가 뷰 정의와 관련성을 검사하여 베이스 릴레이션에 대한 질의를 뷰에 대한 질의로 변형하므로서 질의 처리에 드는 비용을 줄일 수 있다.

시간지원 질의 처리 시스템은 시간 구문 및 의미 분석 단계, 시간지원 실행 트리 생성 단계, 질의 최적화 단계와 실행기로 구성된다. 이들 단계에서 입력된 질의의 뷰 관련성 검사는 질의 최적화 단계에서 수행된다. 질의 최적화 단계에서 뷰 관련성 검사를 수행하여 뷰 정의와 관련된 질의는 질의 변형 과정을 수행한다. 질의 변형 결과 임의의 릴레이션의 튜플 수가 n이고, 이 베이스 릴레이션이 뷰를 소유하고 있으며 베이스 릴레이션의 튜플에 대한 뷰의 비율이 30%라 할 경우 질의 대상이 되는 테이블의 크기는 n*0.3의 크기로 질의 대상이 되는 튜플의 수를 감소시킬 수 있다. 앞으로의 연구 과제는 시간지원 질의 처리시 뷰의 색인을 고려한 질의 처리 기법이 된다.

참고문헌

[1] Surajit Chaudhuri, Ravi Krishnamurthy, Spyros Potamianos, and Kyuseok Shim, "Optimizing Queries with Materialized Views," IEEE Transactions on Knowledge and Data Engineering, 1995, pp. 190-200.

[2] A. Gupta, H. V. Jagadish, and I. S. Mumick, "Data Integration using Self-Maintainable Views," Technical Memorandum, AT&T Bell Laboratories, Nov. 1994.

[3] Eric N. Hanson, "A Performance Analysis of View Management Strategies," In Proceedings of the ACM SIGMOD International Conference on Management of Data, 1987, pp. 440-453.

[4] K.J. Jeong,"Maintenance of materialized View in Temporal Query Processing System,", Ph.D thesis, the Chungbuk National University, 1998.

[5] Date, C.J., "An Intoduction to Database Systems," Additiona-Wesley Pub. Co., Inc., 1986.

[6] K. H. Ryu, "Execution Tree for Incremental View Materialization of Temporal Database," Journal of the KISS , Vol. 20, No. 8, 1993.

[7] Nicholas Roussopoulos,"An Incremental Access Method for ViewCache: Concept, Algorithms, and Cost Analysis," ACM TODBS, Vol. 16, No. 3, Sep. 1991, pp. 535-563.

[8] A. Y. Levy, A. O. Mendelzon, Y. Sagiv, and D. Srivastava, "Answering Queries Using View," In Proc. ACM Symp. on

Principles of Database Systems, 1995.
[9] Susan Horwitz and Tim Teitelbaum, "Generating Editing Environments Based on Relations and Attributes," ACM Transaction on Programming Languages and Systems, 8(4), Oct. 1986.
[10] K. J. Jeong, D. H. Kim, Y. B. Lee, K. H. Ryu, "Query Processing of Temporal Database," Database Review, Vol. 3, No. 2, Database Research of the KISS, 1996.
[11] Yue Zhuge,Hector Garcia-Molina, and Jennifer Widom,"View Maintenance in a Warehousing Environment," ACM SIGMOD International Conference on Management of Data, 1995, pp. 316-327.
[12] G. Zhou, R. Hull, R. King, J-C. Franchitti, "Using Object Matching and Materialization to Intergrate Heterogeneous Databases," In Proceeding of 3rd International Confernce on Cooperative Information System, 1995, pp. 4-18.
[13] Adiba M.E and B.G Lindsay, "Database Snapshots," VLDB Oct., 1980, pp. 86-91.
[14] J. A. Blakeley, N. Coburn and P. -A. Larson, "Updating Derived Relations: Detecting Irrelevant and Automously Computable Updates," ACM TODS, 14(3), 1989, pp. 369-400.
[15] Stefano Ceri and Jennifer Widom, "Deriving Production Rules for Incremental View Materialization," 17th VLDB, 1991, pp. 577-589.
[16] J. Lu, G. Moerkotte, J. Schu, and V. S. Subrahmanian, "Efficient Maintenance of materialized Mediated Views," In Proceedings of the ACM SIGMOD International Conference on Management of Data, 1995, pp. 340-351.

저 자 소 개

정경자

1984.3 - 1988.2 충북대학교 전산통계학과 졸업(이학사)
1991.3 - 1993.2 충북대학교 전자계산학과 졸업(이학석사)
1993.3 - 1998.2 충북대학교 전자계산학과 졸업(이학박사)
1995.3 - 현재 충청대학 멀티미디어과 조교수
관심분야: 시간지원 데이터베이스, 정보검색, 멀티미디어 DB, 시공간 데이터베이스