OLAP에서 MAX-of-SUM 질의의 효율적인 처리 기법

(Efficient Processing of MAX-of-SUM Queries in OLAP)

정 희 정 † 김 동 욱 †† 김 종 수 ††† 이 윤 준 †††† 김 명 호 ††††
(Hee Jeong Cheong) (Dong Wook Kim) (Jong Soo Kim) (Yoon Joon Lee) (Myoung Ho Kim)

요 약 OLAP 분야에서 지금까지 연구되어온 영역 질의는 주어진 영역에 대한 집단 연산의 결과를 구하는 단순한 형태이다. 그러나 실제 데이타 분석 과정에서는 이러한 단순한 형태의 영역 질의뿐만 아니라, 집단 연산이 포함된 특정 조건을 만족하는 데이타 큐브 내의 영역을 찾는 형태의 확장된 영역 질의에 대한 필요성이 존재한다. 본 논문에서는 이러한 확장된 영역 질의 유형의 일반적인 형태를 정의하고, 이에 대한 대표적인 예인 'MAX-of-SUM 질의'의 효율적인 처리 기법을 제안한다. MAX-of-SUM 질의는 데이타 큐브 상에서 영역합(SUM)이 최대(MAX)가 되는 영역을 찾는 질의를 의미한다. 본 논문에서는 MAX-of-SUM 질의 처리 시 검색의 대상이 되는 영역들에 대한 SUM 연산의 결과값이 취할 수 있는 범위를 미리 예측하는 기법을 제안한다. 즉, 영역에 대한 SUM 값의 범위를 예측함으로써, 이들 중에서 최대값을 찾기 위해 실제로 계산하여야 하는 영역의 개수를 줄여 빠른 질의 처리를 보장한다.

Abstract Recent researches about range queries in OLAP are only concerned with applying an aggregation operator over a certain region. However, data analysts in real world need not only the simple range query pattern but also an extended range query pattern that finds ranges which satisfy a special condition specified by using several aggregation operators. In this work, we define the general form of the extended range query and propose an efficient processing method for the 'MAX-of-SUM' query, which is the representative form of the extended range query pattern. The MAX-of-SUM query finds the range which has the maximum range sum value in data cube where the size of the range is given. The proposed query processing method is based on the prediction of the scope of the range sum values. That is, the search space on the query processing can be reduced by using the result of the prediction, and hence, the query response time is also reduced.

1. 서 론

최근 들어 정보의 관리 및 분석의 중요성이 강조됨에 따라 데이타 웨어하우징(data warehousing)과 OLAP (On-Line Analytical Processing)이 의사 결정 지원 시스템(Decision Support System:DSS)의 중요한 요소로 떠오르고 있다. 데이타 웨어하우징이란, 조직 내의 각 데이타 저장소들로부터 추출한 데이타를 중앙의 단일 데이타베이스에 통합하여 저장한 '데이타 웨어하우스(data warehouse)'를 관리하는 기법을 의미한다. OLAP 시스템은 이러한 데이타 웨어하우스에 저장된 방대한 양의 데이타에 대하여, 의사 결정 과정에 도움을 주는 빠르고 직접적인 자료 분석 작업을 수행한다[1, 2, 3].

데이타 웨어하우스에는 일반적으로 수 백 기가 바이트(GB)에서 테라 바이트(TB)에 이르는 매우 방대한 양의 데이타가 저장된다. 또한 대부분의 의사 결정 지원 질의(decision support query)는 개별적인 레코드 정보의 검색이 아닌, 레코드 집합의 전체적인 정보와 경향

† 비 회 원 : 한국전자통신연구원 연구원
hjcheong@etri.re.kr
†† 비 회 원 : 새롬소프트 연구원
baksoo@seromesoft.com
††† 비 회 원 : 한국과학기술원 전산학과
jskim@dbserver.kaist.ac.kr
†††† 종신회원 : 한국과학기술원 전산학과 교수
yjlee@dbserver.kaist.ac.kr
mhkim@dbserver.kaist.ac.kr
논문접수 : 1999년 1월 18일
심사완료 : 1999년 10월 11일

분석 등을 수행하므로 하므로 많은 수의 집단 연산(aggregation)을 포함한다. 이와 같이 데이타의 양이 방대하고 수행되는 질의가 복잡하기 때문에, 대부분의 의사 결정 지원 질의는 처리 시 많은 시간이 소요된다. 그러나 OLAP 시스템은 의사 결정 과정에 즉시 반영될수 있는 직접적인 데이타 분석을 수행해야 하므로, 빠른 질의 응답 속도의 보장이 필수적이다. 이를 위하여 OLAP 분야에서는 자주 수행되는 질의의 처리에 필요한 정보를 미리 계산하여 저장하고, 실제 질의 처리 시 저장된 정보를 활용하여 질의 응답 속도를 줄이고자 하는 선계산(pre-computation) 기법이 활발히 연구되고 있다[4, 5, 6, 7].

OLAP 환경에서는 데이타를 효과적으로 시각화하고 여러가지 다른 관점에서의 분석을 효율적으로 수행하기 위하여, 데이타 큐브(data cube)라 불리는 다차원 데이타 모델을 사용한다[8]. 데이타 큐브의 애트리뷰트는 분석의 대상이 되는 '측정 애트리뷰트(measure attribute)'와 측정 애트리뷰트 값을 유일하게 식별하는 '차원 애트리뷰트(dimension attribute)'로 분류된다. 다차원 데이타 모델에서 데이타 큐브는 d-차원 배열로 간주될 수 있으며, 이 경우 데이타 큐브의 각 셀(cell)은 d-차원 애트리뷰트에 의하여 지정되는 측정 애트리뷰트 값을 갖는다.

일반적으로 의사 결정을 위한 데이타 분석은 다차원 데이타에 대한 집단 연산의 처리를 필요로 한다. 지금까지 OLAP 분야에서 연구된 질의 유형은 크게, 데이타 큐브 내의 하나의 셀에 대한 접근 만으로 결과를 얻을 수 있는 '단일값 질의(singleton query)'[9, 10, 11]와 각 차원에 대한 구간이 지정된 다차원 영역에 대해서 집단 연산을 수행하는 '영역 질의(range query)'[12]로 구분할 수 있다. 그러나 효과적인 의사 결정 지원을 위해서는 이러한 기본적인 질의 유형 이외에도, 좀 더 복잡하지만 데이타 분석 과정에서 유용한 질의 유형에 대한 효율적인 처리 기법이 제시되어야 한다.

지금까지 연구되어 온 영역 질의는 주어진 영역에 대한 집단 함수의 수행 결과를 구하는 형태의 질의를 의미한다. 그러나 실제 데이타 분석 과정에서는 이러한 단순한 형태의 영역 질의 뿐만 아니라, 집단 함수가 포함된 특정 조건을 만족하는 데이타 큐브 내의 영역을 찾는 형태의 확장된 영역 질의에 대한 필요성이 존재한다. 즉, 이와 같이 좀 더 복잡한 형태의 질의를 사용하여, OLAP 시스템에서의 보다 효과적인 의사 결정 지원을 보장할 수 있다. 본 논문에서는 이러한 확장된 영역 질의 유형의 일반적인 형태를 정의하고, 이에 대한 대표적인 예인 'MAX-of-SUM 질의'에 대한 효율적인 처리 방안을 제안한다. MAX-of-SUM 질의는 데이타 큐브 상에서 영역합(SUM)이 최대(MAX)가 되는 영역을 찾는 질의를 의미한다. 또한 본 논문에서는 MAX-of-SUM 질의의 처리 기법을 응용하여, 다른 집단 함수가 사용된 일반적인 유형의 질의에 대한 효율적인 처리 기법도 쉽게 얻을 수 있음을 보인다.

본 논문은 다음과 같은 순서로 구성되어 있다. 2장에서는 본 논문에서 다루고자 하는 문제에 해당하는 확장된 영역 질의를 정의한다. 3장에서는 문제를 해결하기 위해 사용되는 선계산(pre-computation) 연산자 Max_Avg와 Min_Avg를 정의하고, 이를 이용한 질의 처리 기법을 제안한다. 4장에서는 실험을 통해 본 논문에서 제안된 기법의 성능을 보이고, 성능 향상을 위한 지침을 제시한다. 마지막으로 5장에서는 결론을 맺고 추후 연구 방향을 제시한다.

2. 문제 정의

OLAP 분야에서의 질의 처리에 관한 기존 연구들은 주로 데이타 큐브 상의 지정된 영역에 대한 집단 함수의 값을 계산하는 영역 질의에 초점을 맞추고 있다. 영역 질의에 관한 대표적인 연구인 [12]에서는, 주어진 영역에 대한 집단 연산 SUM과 MAX를 효율적으로 처리하기 위한 기법을 제안하였다. 즉, 접두합 배열(prefix-sum array) 및 계층 트리 구조 등과 같은 특수한 자료 구조 상에 선계산 결과를 저장함으로써, 지정된 영역에 대한 집단 연산을 빠른 시간에 계산하는 기법을 제안하였다.

기존 연구들이 대상으로 하고 있는 영역 질의는 지정된 데이타 집합에 대한 전체적인 정보를 얻을 수 있다는 점에서 의사 결정 과정에 많은 도움을 줄 수 있다. 그러나 OLAP 환경에서의 보다 효과적인 데이타 분석 및 의사 결정 지원을 위해서는 이와 같이 단순한 영역 질의 뿐만 아니라, 좀 더 복잡한 형태의 확장된 영역 질의의 필요성이 존재한다. 즉, 전체 데이타 큐브 상에서, 집단 함수가 사용된 특정한 조건을 만족하는 영역을 찾는 질의를 사용하여 의사 결정 과정을 보다 효과적으로 지원할 수 있다. 현재까지 대부분의 연구는 단순한 영역 질의의 처리를 대상으로 하고 있으며, 이러한 새로운 형태의 영역 질의 유형에 대한 연구는 아직까지 조사된 바 없다.

확장된 영역 질의의 의미를 좀 더 명확히 하기 위하여 그림 1을 살펴보자. 그림 1은 각 상장사의 월별 증권 거래량을 나타내는 데이타 큐브이다. 이와 같은 데이타

큐브 상에서 "3개월 간의 거래량이 최대인 상장사에 대하여 최대의 거래량과 그 기간을 찾아라."와 같은 질의를 생각해 볼 수 있다. 이러한 질의는 검색될 영역의 위치가 지정되지 않고 단지 영역의 크기만이 주어질 때, 3개월 간의 최대 거래량이라는 특정한 조건을 만족하는 영역 및 해당 영역에서의 총 거래량을 구한다. 이 때, 거래량은 월별로 저장되어 있으므로 주어진 질의의 결과를 구하기 위해서는 먼저 모든 3개월 간의 구간에 대한 거래량을 합(SUM)하고, 이들 중의 최대값(MAX)을 구해야 한다. 즉, 주어진 질의는 대상이 되는 모든 영역에 대하여 특정한 계산(SUM/COUNT/AVG)을 수행한 결과에 대해 대표값(MAX/MIN)을 찾는 형태로서, 두 개의 집단 연산이 연속적으로 사용된다. 본 논문에서는 이와 같이 두 집단 연산을 연속적으로 사용하여 명시된 특정 조건을 만족하는 영역 및 해당 영역에서의 집단 연산 결과를 계산하는 형태의 질의에 대한 효과적인 처리 방안을 제안한다.

상장사

기간	A	B	C	D	E
3	120	420	80	10	70
4	170	200	130	20	90
5	220	90	200	20	185
6	250	20	70	30	120
7	310	30	50	400	75

그림 1 월별 증권 거래량이 기록된 2차원 데이타 큐브의 예

문제를 정의하기에 앞서 먼저 데이타 큐브와 영역의 정의를 살펴보자. n_i가 i차원의 도메인(domain)의 크기를 나타낸다고 할 때, k-차원 데이타 큐브 D는 $n_1 \times n_2 \times \cdots \times n_k$의 크기를 갖는 k-차원 배열로 표시된다(단, $i \leq k$). k-차원 데이타 큐브 D에서 임의의 셀 $D(p_1,p_2,\cdots,p_k)$는 배열의 원소 $(p_1,p_2,\cdots,p_k)$에 대응되는 값을 나타낸다. 또한, l_i와 u_i가 각각 i차원의 영역에 대한 임의의 시작값과 끝값을 의미한다고 할 때, k차원 데이타 큐브 D의 임의의 영역은 $Range(l_1{:}u_1,l_2{:}u_2,\cdots,l_k{:}u_k)$로 표현되며, 이는 $l_i \leq p_i \leq u_i$ 인 데이타 큐브의 셀 $D(p_1,p_2,\cdots,p_k)$의 집합으로 구성되는 영역을 의미한다[12]. 본 논문에서는 데이타 큐브의 일반적인 n 개의 차원 중 두 개의 차원만이 질의에 참여한다고 가정한다. 이후에서는 본 논문에서 제안하는 기법의 이해를 쉽게 하기 위해 2차원 데이타 큐브를 가정하여 설명한다.

이제, 크기가 $n_1 \times n_2$인 2차원 데이타 큐브 D에 대해, 찾고자 하는 영역의 크기가 R로 주어지는 '확장된 영역 질의'는 다음과 같이 정의될 수 있다.

정의 1 확장된 영역 질의:
2차원 데이타 큐브 D에 대하여 영역이 검색될 차원 i와 (i=1 또는 2) 임의의 영역 크기 R이 주어질 때, 확장된 영역 질의 Q는 다음과 같이 정의된다: Q는 $Range(s_1{:}s_1{+}R{-}1,\ idx_2{:}idx_2)$ 또는 $Range(idx_1{:}idx_1,\ s_2{:}s_2{+}R{-}1)$의 영역에 대하여 두 개의 집단 연산으로 명시된 조건을 만족하는 데이타 큐브 상의 영역과 해당 영역의 집단 연산 결과 값을 찾는 질의 Q를 의미한다. 즉,

$$Q = \{G_{\ 1 \leq s1 \leq n1-R+1,\ 1 \leq idx2 \leq n2}\ [F(Range(s_1{:}s_1{+}R{-}1,\ idx_2{:}idx_2)]\}$$
(i = 1인 경우)

또는

$$Q = \{G_{\ 1 \leq idx1 \leq n1,\ 1 \leq s2 \leq n2-R+1}\ F(Range(idx_1{:}idx_1,\ s_2{:}s_2{+}R{-}1)]\}$$
(i = 2인 경우)

여기에서 G는 집단 연산 MAX 또는 MIN을, F는 집단 연산 SUM, $COUNT$ 또는 AVG 중의 하나를 각각 나타낸다. 또한, s_i는 차원 i에서 찾고자하는 영역의 시작점을 나타내고, idx_j는 이 경우 다른 차원 j에 대한 색인을 의미한다.

앞서 정의한 확장된 영역 질의에서 COUNT나 AVG는 SUM의 경우에 대한 약간의 변형을 통해 결과를 얻을 수 있으며, MIN은 셀 값의 부호를 바꾼 후 MAX에 대한 처리 기법을 그대로 적용할 수 있다.[1] 따라서, 본 논문에서는 대상 영역에 대한 SUM을 수행한 후 MAX를 수행하는 질의 유형에 대해서만 논하고자 한다. 본 논문의 대상이 되는 MAX-of-SUM 질의는 다음과 같이 정의된다.

정의 2 MAX-of-SUM 질의:
2차원 데이타 큐브 D에 대하여 영역이 검색될 차원 i와 (i=1 또는 2) 임의의 영역 크기 R이 주어질 때, MAX-of-SUM 질의 Q는 다음과 같이 정의된다: Q는 $Range(s_1{:}s_1{+}R{-}1,\ idx_2{:}idx_2)$ 또는 $Range(idx_1{:}idx_1,\ s_2{:}s_2{+}R{-}1)$의 영역에 대하여 두 개의 집단 연산 SUM과 MAX로 명시되는 조건을 만족하는 데이타 큐브 상의 영역과 해당 영역의 집단 연산 값을 찾는 질의 Q를

1) 이 경우 셀의 모든 값은 동일 부호라고 가정한다. 셀의 값에 서로 다른 부호의 값이 존재하는 경우에는 코드화 기법등을 통하여 해결할 수 있다.

의미한다. 즉,

$Q = \{MAX_{1<s1<n1-R+1,\ 1<idx2<n2}\ [SUM(Range(s_1{:}s_1{+}R{-}1, idx_2{:}idx_2)]\}$ (i = 1인 경우)

또는

$Q = \{MAX_{1<idx1<n1,\ 1<s2<n2-R+1}\ [SUM(Range(idx_1{:}idx_1, s_2{:}s_2{+}R{-}1)]\}$ (i = 2인 경우)

	A	B	C	D	E
3월 ~ 5월	510	710	410	50	335
4월 ~ 6월	640	310	400	70	395
5월 ~ 7월		140	320	450	380

그림 2 그림 1의 증권 데이타에 대한 연속된 3개월 동안의 총 거래량

3. MAX-of-SUM 질의 처리 기법

본 논문에서 제안하는 MAX-of-SUM 질의 처리 기법의 기본적인 내용은 검색의 대상이 되는 영역들에 대한 SUM 연산의 결과값이 취할 수 있는 범위를 미리 예측하는 것이다. 즉, 영역에 대한 SUM 값의 범위를 미리 예측함으로써, 이들 중에서 최대값을 찾기 위해 실제로 계산하여야 하는 영역의 개수를 줄여 빠른 질의 처리를 보장한다.

3.1 기본 성질

그림 1의 데이타 큐브에서 "3개월 간의 거래량이 최대인 상장사에 대하여 최대의 거래량과 그 기간을 구하라."라는 질의를 처리하는 경우를 생각해보자. 이에 대한 가장 손쉬운 방법은 데이타 큐브 상의 모든 가능한 영역에 대해서 연속된 3개월 간의 거래량을 합을 모두 구하고, 이들 중 최대값을 찾는 방법이다. 그림 2는 연속된 3개월 간의 모든 영역에 대하여 거래량을 합한 결과와 실제 최대값을 나타내고 있다. 즉, 질의의 수행 결과는 영역 [5월, 7월]과 이 영역에서의 거래량의 합 780 (그림에서 어둡게 표시된 값)이 된다. 그림 2의 데이타를 기반으로, 연속된 3개월의 기간에 대해 각 상장사의 증권 거래량의 합이 가질 수 있는 값의 범위를 그래프로 표현하면 그림 3과 같이 표현된다.

그림 3에서 상장사 A의 거래량의 합은 최소 510 이상인데 반하여, 상장사 C, D 및 E는 거래량의 합의 최대값이 510보다 작음을 알 수 있다. 즉, 상장사 C, D 및 E는 절대로 A보다 큰 거래량의 합을 갖지 못하므로, 질의의 결과로 선택될 수 없다. 이와 같이 각 상장사가 가질 수 있는 영역합의 범위를 미리 예측할 수 있다면, 질의 처리 시 실제로 계산을 수행하여야 하는 데이타의 양을 줄일 수 있고, 따라서 빠른 질의 응답 속도를 보장할 수 있다. 본 논문에서는 이러한 속성을 활용할 수 있도록, 영역합의 범위를 미리 예측할 수 있는 측정자(measure)와 이에 기반한 질의 처리 기법을 제안한다.

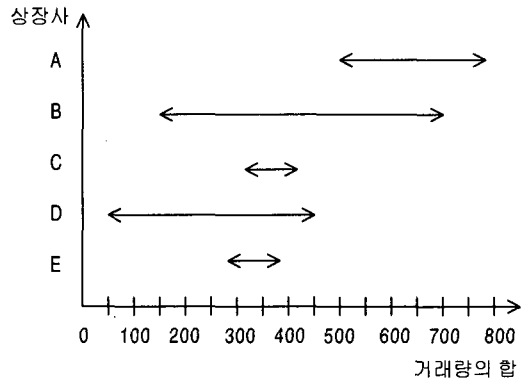

그림 3 연속된 3개월 간의 각 상장사별 거래량의 합이 가질 수 있는 값의 범위

3.2 영역합 측정자

크기가 $M \times N$ 인 2차원 데이타 큐브 상에서, 크기가 M인 차원이 영역합의 주체가 되고(그림 1의 예제에서는 '상장사'), 크기가 N인 차원이 영역합이 계산되는 차원(그림 1에서는 '기간')이라고 가정하자. 이 데이타 큐브를 2차원 배열 $A[M][N]$이라고 간주하면, 영역합을 예측하기 위한 측정자 Max_Avg, Min_Avg는 다음과 같이 정의된다.

정의 3 영역합 측정자 Max/Min_Avg

$$Max\_Avg[i] = MAX_{(e-s)\geq MSQ}\left(\sum_{j=s}^{e}\frac{A[i][j]}{e-s+1}\right)\ for\ all\ s,e\in\{1,\cdots,N\}$$

$$Min\_Avg[i] = MIN_{(e-s)\geq MSQ}\left(\sum_{j=s}^{e}\frac{A[i][j]}{e-s+1}\right)\ for\ all\ s,e\in\{1,\cdots,N\}$$

Max_Avg[i](또는 Min_Avg[i])는 영역합의 주체가 되는 차원의 임의의 원소 i에 대하여, 질의에서 주어질 수 있는 가능한 모든 영역의 크기에 대한 영역합의 평균값 중 최대값(또는 최소값)을 하나의 셀 단위로 나타낸 값이다.

정의 3에서 MSQ(Minimum Supported Query range size)는 질의에서 사용될 수 있는 최소 영역의 크기를 의미하며, 시스템에서 미리 설정하는 값이다. MSQ는 Max/Min_Avg의 계산 결과가 데이타 큐브 상

의 몇몇 특이한 셀 값에 의하여 영향을 받는 현상을 억제하는 역할을 담당한다. 즉, MSQ의 사용없이 가능한 영역의 최소 크기를 1이라 가정하면, Max_Avg와 Min_Avg는 값이 매우 크거나 매우 작은 특정한 셀에 의하여 왜곡될 가능성이 존재한다. 이와 반대로 MSQ가 크면 클수록 특정 셀 값들에 의한 영향을 적게 받으므로, 좀더 상세한 영역합의 범위를 예측할 수 있게 된다. 그러나 이 경우 질의 시 허용될 수 있는 영역의 크기는 MSQ 이상으로 제한되므로, 무한정 큰 값으로 MSQ를 설정하는 것도 불가능하다. 시스템이 좋은 성능을 나타내도록 적절한 MSQ 값을 설정하는 문제에 대해서는 4장에서 좀 더 자세히 논한다.

앞서 제시한 Max/Min_Avg의 정의를 기반으로 다음과 같은 성질을 얻을 수 있다.

소정리 1 *MSQ*≤*r*≤*N*을 만족하는 크기 *r*의 영역 *R*에 대하여, 각 원소 *i*의 영역합은 *[Min_Avg[i]×r, Max_Avg[i]×r]* 사이의 값을 갖는다. (단, *1*≤*i*≤*M*이고, *M*은 영역합의 주체가 되는 차원의 크기를, *N*은 영역합이 계산되는 차원의 크기를 나타낸다.)

소정리 1의 증명:

정의 3에 의하여 Max_Avg[i]는 원소 i에 대한 가능한 모든 영역합의 평균값 중 최대값을 나타낸다. 따라서, MSQ 이상의 크기 r인 임의의 영역 R에 대하여 다음이 항상 성립한다.

$$Max\_Avg[i] \geq \frac{R\text{의 영역합}}{r}$$

동일한 방법으로 Min_Avg[i]의 경우도 증명할 수 있다. □

즉, 임의의 질의 영역 크기 R이 주어지면, 원소 i에서의 크기 R인 영역합은 Max_Avg[i]×R 과 Min_Avg[i]×R 사이의 값을 갖게 된다. 따라서, 이들 Max_Avg와 Min_Avg 값을 미리 계산하여 놓으면, 질의의 영역 크기로 어떤 값이 주어지건 간에 해당 영역합이 취할 수 있는 값의 범위를 예측할 수 있다.

소정리 1의 결과로부터, 질의의 영역 크기가 주어지면 각 원소별 영역합이 취할 수 있는 값의 범위를 알 수 있으며, 이를 기반으로 질의 처리 시 실제 계산할 필요가 없는 원소들을 식별할 수 있다. 즉, 그림 4에서와 같이 원소 i의 영역합 중 최소값이 원소 j의 최대 영역합보다 크다면 원소 j는 원소 i보다 더 큰 영역합을 가질 수 없음을 알 수 있다.

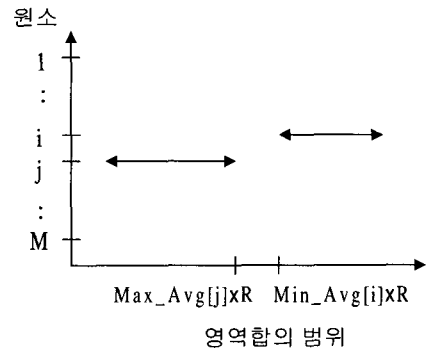

그림 4 [속성 1] 절대적 우위의 영역합 범위

소정리 2 [속성 1]

1≤*i, j*≤*M*인 *i, j* 에 대해서 *Min_Avg[i] > Max_Avg[j]* 이면, 질의에서 주어지는 모든 영역 크기에 대하여 원소 j는 원소 i보다 더 큰 영역합을 가질 수 없다.

소정리 2의 증명:

1≤*i, j*≤*M*인 *i, j* 에 대해서 *Min_Avg[i] > Max_Avg[j]* 이 성립한다면, 소정리 1에 의해서 다음이 항상 성립한다.

원소 *i*에서 영역 크기가 *R*인 임의의 영역합 ≥ *Min_Avg[i]*×*R*

Max_Avg[i]×*R* ≥ 원소 *j*에서 영역 크기가 *R*인 임의의 영역합

따라서 항상 다음이 성립함을 알 수 있다.

원소 *i*에서 영역 크기가 *R*인 임의의 영역합 > 원소 *j*에서 영역 크기가 *R*인 임의의 영역합 □

앞서 언급한 바와 같이, 소정리 2의 결과를 이용하여 질의의 결과에 참여할 수 없는 원소들을 미리 식별할 수 있다. 이 경우 질의 처리 시 무시될 원소들은 질의에서 사용되는 영역의 크기와 무관하게 미리 계산되어 저장된 Max_Avg와 Min_Avg 값의 비교 만으로 찾을 수 있다. 따라서, 소정리 2에 의한 무시될 원소들의 선정은 비작동 시간(off time)에 수행될 수 있으며, 실제 실행 시간(run time)에는 소정리 2에 의하여 판단할 수 없는 나머지 원소들에 대해서만 처리를 하면 된다.

실제 실행 시간에서 사용될 수 있는, 질의 처리 시 무시될 원소의 선정 방법은 다음과 같다. 질의를 처리하는

과정에서 임의의 원소 i에 대한 크기 R의 영역합을 계산하여, 이들 중 최대값 MAX^i_R을 얻었다고 가정하자. 이 때, 아직 영역합이 계산되지 않은 또 다른 원소 j에 대하여, j의 임의의 영역합이 취할 수 있는 값의 범위가 MAX^i_R 보다 작다면, 원소 j는 질의의 결과로 선택될 수 없고 따라서 j에 대한 계산을 수행할 필요가 없음을 알 수 있다. 즉, 실제 계산 과정에서 임의의 원소 i에 대한 MAX^i_R이 확보된 경우, 가능한 영역합의 최대값이 MAX^i_R보다 작은 원소 들은 실제 검색 및 계산의 대상에서 제외된다. 그림 5는 이와 같은 관계를 나타내고 있으며, 구체적인 내용은 다음의 소정리 3과 같이 기술된다.

소정리 3 [속성 2]

원소 i에서 크기 R인 영역합들 중의 최대값 MAX^i_R에 대하여,

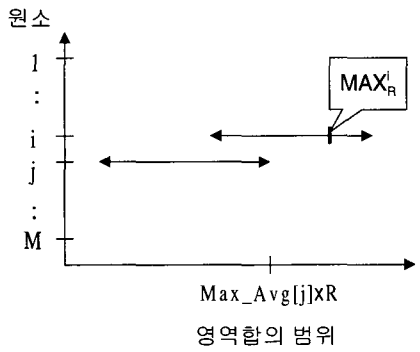

그림 5 [속성 2] 최대값의 우위에 따른 영역합 범위

$$MAX^i_R > Max\_Avg[j] \times R \text{ (단, } 1 \le i, j \le M)$$

이 성립하면 원소 j는 원소 i의 최대 영역합보다 더 큰 영역합을 가질 수 없다.

소정리 3의 증명:

$1 \le i, j \le M$ 인 i, j에 대하여, 소정리 1의 결과로부터 다음이 성립한다.

$Max\_Avg[j] \times R$ > 원소 j에서 영역 크기가 R인 영역합

또한 가정에 의하여 $MAX^i_R > Max_Avg[j] \times R$이 성립하므로, 다음의 식은 항상 성립한다.

MAX^i_R > 원소 j에서 영역 크기가 R인 영역합 □

이상에서와 같이 소정리 2와 소정리 3을 이용하여, MAX-of-SUM 질의 처리 시 검색 및 영역합 계산이 필요 없는 원소들을 식별할 수 있다. 이는 전체 데이타 큐브 중 실제 계산에 참여하는 데이타의 양을 줄여 질의 응답 속도를 향상 시킬 수 있음을 의미한다. 특별히 소정리 2에 제시된 기준은 질의에 무관하게 비작동 시간에 적용될 수 있으므로 매우 효과적이다. 다음 절에서는 이들 두 가지 기준을 사용한 MAX-of-SUM 질의 처리 알고리즘을 제시한다.

3.3 MAX-of-SUM 질의 처리 알고리즘

그림 6은 앞서 제시한 원소 식별 기준을 기반으로한 MAX-of-SUM 질의 처리 알고리즘을 나타낸다. 단계 (1)-(3)은 소정리 2를 적용하는 과정으로, 비작동 시간에 검색할 필요가 없는 원소들을 식별한다. 식별된 원소들은 "Searched"로 표시되어 추후 검색 과정에서 제외된다. 실행 시간에는 단계 (5)-(14)의 과정을 통하여 소정리 3의 조건에 해당되는 원소들을 찾는다. 이 과정에서 검색 및 영역합 계산을 수행하는 원소의 순서는 'Max_Avg에 대한 내림차순' 등과 같은 경험적 방법(heuristic method)을 적용할 수도 있다. 알고리즘의 수행 결과로는 최대의 영역합을 갖는 영역의 시작 위치 "Item_position"과 이때의 최대 영역합 "Max"가 얻어진다.

```
Procedure Search_MAX-of-SUM
MAX-of-SUM(Max_Avg[M], Min_Avg[M], range_size)
Max_Avg[M] : M개의 각 원소에 대한 Max_Avg 값
Min_Avg[M] : M개의 각 원소에 대한 Min_Avg 값
range_size : 질의에서 주어진 영역 크기
begin
(1) for(i=1;i≤M;i++)
(2)       for(j=1;j≤M;j++)
(3)             if(Min_Avg[i]>Max_Avg[j]) Mark item
                  j as "Searched";
(4) Max = 0, i = 0;
(5) while(∃(item which is not marked as "Searched"))
(6)       i++;
(7)       if(item k which has ith Max_Avg is not marked
            as "Searched")
(8)             Find  MAX^i_R and store position;
(9)             Mark item k as "Searched";
(10)            if(Max < MAX^i_R)
(11)                  Max =  MAX^i_R
(12)                  store position to Item_position;
(13)                  for(j=1;j<=M;j++)
(14)
if(Max_Avg[j]×R<Max) Mark item j as "Searched";
end
```

그림 6 MAX-of-SUM 질의 처리 알고리즘

정리 1 알고리즘 Search_MAX-of-SUM은 2차원 데이

타 큐브 D에 대한 MAX-of-SUM 질의에 대하여 올바른 질의 결과를 생성한다.

증명

Search_MAX-of-SUM 알고리즘은 데이타 큐브의 일부 원소만을 검색하여 질의에 대한 결과를 생성한다. 알고리즘의 내용에 의하여, 실제로 검색되는 원소들에 의한 결과가 올바른 질의 결과임은 자명하다. 또한, 검색 대상에서 제외되는 원소들은 앞의 소정리 2와 3에 의하여 최대의 영역합이 존재할 수 없는 원소임이 증명되었다. 즉, 최대 영역합이 존재할 가능성이 있는 모든 원소들에 대해서 실제 계산을 통한 질의 결과를 생성하였으므로, 그 결과는 항상 올바르다. □

4. 성능 평가

이 장에서는 실험을 통해 본 논문에서 제안한 Search_MAX-of-SUM 알고리즘의 성능을 평가한다. 실험은 2차원 데이타 큐브를 대상으로 수행되었으며, 제시된 실험 결과는 20번의 반복 실험에 의한 측정치들의 평균값이다.

4.1 실험 환경

일반적으로 OLAP 시스템에서의 질의 처리 비용은 질의 처리 시 검색되는 데이타 큐브 내의 데이타의 양으로 표현될 수 있다. 본 실험에서는 질의 처리 시 검색되는 원소의 개수를 성능의 척도로 삼는다. 데이타 큐브상에서 검색되는 원소의 수는 질의 처리 시 검색되는 데이타의 양과 비례하므로, 검색되는 원소의 개수를 측정함으로써 질의 처리 비용을 추정할 수 있다. 현재까지 MAX-of-SUM 질의의 처리에 대한 기존 연구는 조사된 바 없다. 따라서, 본 논문에서는 선계산 기법을 사용하지 않고 모든 원소에 대하여 영역합을 직접적으로 수행하여 최대값을 찾는 일반적인 질의 처리 방식을 성능 비교의 대상으로 한다.

본 논문에서는 두 가지 서로 다른 분포를 갖는 데이타 큐브에 대하여 실험을 수행한다. 먼저 제안된 기법이 임의의 데이타 집합에 대하여 유용함을 보이기 위해, 데이타 큐브의 모든 셀 값이 균일 분포(uniform distribution)를 갖는 경우에 대한 실험을 수행한다. 그러나 실제 응용 환경에서의 데이타 집합은 각 원소마다 셀 값의 분포 범위가 서로 차이를 나타내는 경우가 빈번하다. 예를 들어, 도시별 강수량을 저장한 데이타 큐브의 경우, 제주 등과 같이 강수량이 많은 도시와 내륙의 강수량이 적은 도시는 그 강수량의 분포 범위에 있어서 많은 차이를 나타내게 된다. 본 논문에서는 이와 같은 환경을 나타내기 위하여, 데이타 큐브의 각 원소들은 서로 다른 값의 분포 범위를 갖고, 원소 내부에서는 균일한 분포를 갖는 경우에 대한 실험도 수행한다.

본 실험에서 사용되는 변수는 다음과 같다.

1. 질의가 가질 수 있는 최소 영역 크기(MSQ) : Max/Min_Avg를 계산할 때 사용되는 최소의 질의 영역 크기를 의미한다. 값이 크면 클 수록 특정 셀 값의 영향을 적게 받으므로 좀더 구체화된 영역합 범위를 나타낼 수 있다.
2. 질의 영역 크기(R) : 질의에서 주어지는 영역 크기를 나타낸다.

4.2 결과 및 분석

4.2.1 전체적 균일 분포의 데이타 큐브

그림 7은 256×256 데이타 큐브에 대한 실험 결과를 나타낸다. 그래프에서 x축은 질의에서 주어진 영역 크기 R을, y축은 실제 검색을 수행해야 하는 원소의 수를 나타낸다. 실험 결과는 위에서 아래 방향으로 각각 전체 데이타에 대하여 선계산 없이 직접 검색을 수행한 경우(그래프에서는 MSQ가 no인 경우로 표시), MSQ를 10으로 설정한 경우, MSQ를 20으로 설정한 경우, … , MSQ가 70으로 설정된 경우를 나타낸다. 실험 결과에서와 같이 본 논문에서 제안한 질의 처리 기법을 사용하는 경우, 기존의 전체 검색 기법을 사용하는 경우보다 검색해야 하는 데이타의 양이 현저히 감소함을 알 수 있다. 특히 MSQ가 커질수록, 그리고 질의에 주어지는 영역의 크기가 MSQ에 가까울수록 더 좋은 성능을 보임을 알 수 있다.

그림 8은 정방형이 아닌 데이타 큐브의 경우로서, 256×1024 데이타 큐브에 대한 동일한 내용의 실험 결과를 나타낸다. 앞의 실험에 비하여 검색 대상이 되는 원소 수가 4배 증가하였음에도 불구하고, 256×256 데이타 큐브와 비슷한 성능 향상을 보이고 있다. 즉, 원소의 수가 증가되어도 실제 검색해야 할 원소의 수는 총 원소 수에 대해 비슷한 비율로 유지됨을 알 수 있다. 이러한 결과를 바탕으로 데이타 큐브의 크기가 확장되는 경우, 예상되는 질의 응답 시간을 미리 예측할 수도 있다.

4.2.2 원소별 균일 분포의 데이타 큐브

대부분의 실제 응용 환경의 데이타는 각 원소마다 셀 값의 분포 범위가 서로 다른 경우가 빈번하다. 예를 들어, 도시별 월별 강수량 데이타의 경우 제주가 50mm-250mm 정도의 월평균 강수량을 나타내는데 반하여, 대구는 20mm-200mm 정도의 월평균 강수량을 나타낸다. 따라서 각 원소마다 다른 셀 값의 범위를 가

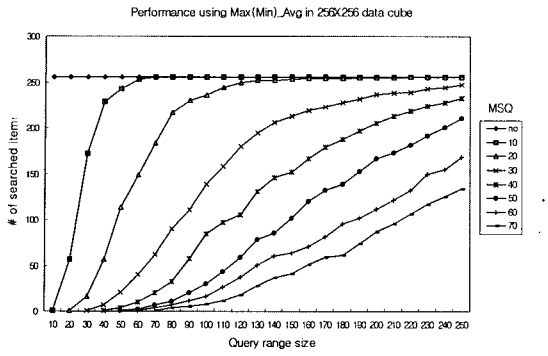

그림 7 256×256 데이타 큐브(전체 균일 분포)

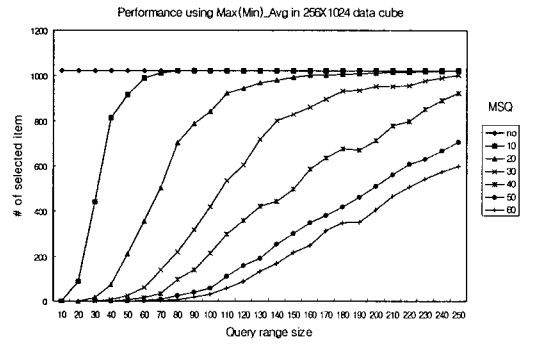

그림 8 256×1024 데이타 큐브(전체 균일 분포)

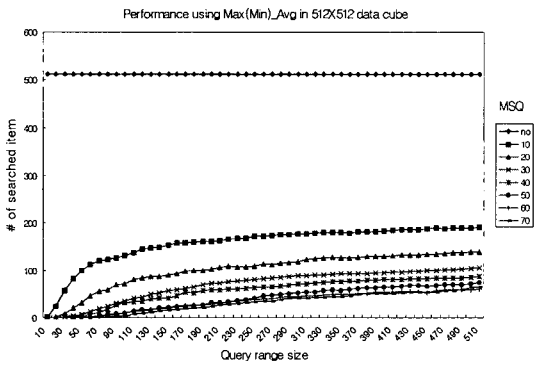

그림 9 512×512 데이타 큐브(원소별 균일 분포)

지면서, 원소 내부에서는 균일한 분포를 갖는 경우에 대한 실험을 수행했다.

그림 9는 이에 대한 실험 결과를 나타낸다. 이 경우 각 원소마다 서로 다른 셀 값의 분포 범위를 가지므로, 각 원소가 갖는 영역합의 범위 또한 서로 다른 분포를 갖게 된다. 따라서 본 논문에서 제안한 원소 식별 기준이 더욱 효과적으로 적용될 수 있으며, 결과적으로 데이타 큐브의 전체가 균일 분포를 따르는 경우보다 더 우수한 성능을 나타낸다.

4.2.3 결과 분석

앞 절에서 제시한 실험 결과는, MAX-of-SUM 질의 처리 시 본 논문에서 제안한 기법을 사용하면 전체 데이타 큐브에 대한 검색을 수행하지 않고, 일부만을 검색하여 질의 응답 시간을 크게 줄일 수 있음을 보이고 있다. 즉, 실험 결과를 나타내는 그림 7,8 및 9에서 알 수 있듯이, N×N 데이타 큐브에 대하여 N개의 원소를 모두 검색할 필요 없이 보다 적은 수의 원소만을 검색함으로써 결과를 구할 수 있다.

본 논문에서 제안한 기법은 각 원소가 취할 수 있는 영역합 범위의 차이를 이용하여, 검색될 필요가 없는 원소들을 식별한다. 따라서 각 원소의 영역합들이 분포할 수 있는 범위가 좁을수록 더 좋은 성능을 나타낸다. 영역합의 분포가 좁은 범위를 갖는 경우로는 MSQ가 큰 값으로 설정된 경우와 질의로 들어오는 영역 크기가 MSQ에 가까운 경우 등을 생각할 수 있다. MSQ가 크면 클수록 Max/Min_Avg의 계산 시 특정 셀 값의 영향을 적게 받으므로, 영역합의 범위를 좀더 효과적으로 예측할 수 있다. 그러나 MSQ가 커지면 커질수록 지원할 수 있는 질의의 범위는 축소되므로, 적절한 MSQ를 설정하는 것은 전체 시스템의 성능 향상을 위해 매우 중요하다. MSQ 값을 설정하는 효과적인 기준은 4.3절에서 좀 더 자세히 설명한다.

앞의 실험 결과에서 질의로 주어지는 영역의 크기가 커질 수록 성능이 저하됨을 알 수 있다. 이는 영역합의 예측 범위가 Max/Min_Avg×R로 표현되기 때문이다. 즉, 질의의 영역 크기 R이 커질 수록, 예측된 영역합의 범위는 더 커지게 되고 이는 본 논문에서 제안한 원소 식별 기준의 적용에 방해 요소로 작용한다.

본 논문에서 제안한 기법을 이용하기 위해서는 선계산 시간에 모든 원소에 대한 Max/Min_Avg 값을 미리 계산하여 저장해 두어야 한다. 따라서 M×N 데이타 큐브에 대해서 O(M)의 추가 공간을 필요로 하지만, 전체 데이타의 크기에 비하면 비교적 적은 부담에 해당한다.

4.3 MSQ(Minimum Supported Query range size)

MSQ의 값은 본 논문에서 제안된 기법의 성능에 큰 영향을 미친다. 앞에서 언급한 바와 같이, MSQ가 클수록 좋은 성능을 나타내지만, MSQ 보다 작은 영역 크기

의 질의를 처리할 수 없는 문제점이 존재한다. 따라서, 시스템에서 사용되는 질의의 영역 크기 등을 고려하여 적절한 크기의 MSQ 값을 설정하는 것이 시스템 성능을 결정하는 중요한 요소가 된다.

본 논문에서는 이와 같은 문제를 해결하기 위해서 하나 이상의 MSQ를 도입하는 방법을 제안한다. 즉, 최소 질의 영역 크기에 대해 설정된 기본 MSQ 이외에, 질의의 분포에 따라 추가적인 MSQ를 설정한다. 만약 질의에서 빈번히 사용되는 영역 크기를 통계적 분석 방법 등을 통하여 미리 알 수 있다면, 해당 영역 크기와 유사한 값의 MSQ를 추가할 수 있다. 그림 10은 256×256 데이타 큐브에서 10을 기본 MSQ 값으로 하고, 70에 또 하나의 MSQ를 설정한 경우의 실험 결과를 나타낸다. 이는 응용 환경에서 가능한 최소의 질의 영역 크기는 10이지만, 영역 크기 70에 대한 질의가 빈번히 사용되는 경우에 대한 MSQ 설정 방법으로 볼 수 있다. 그림에서와 같이 두 개의 MSQ를 사용하는 경우, 하나의 MSQ를 사용하는 경우보다 더 낮은 성능을 나타냄을 알 수 있다.

하나의 MSQ를 추가로 사용하는 경우, O(M)의 공간을 추가로 사용하게 된다. 그러나 이는 전체 데이타 큐브의 크기인 M×N에 비해 그다지 크지 않다. 또한 비작동 시간에 여러 MSQ들에 대한 Max/Min_Avg를 구하는 경우, 가장 작은 MSQ에 대한 Max/Min_Avg를 구하는 과정에서 그보다 큰 모든 MSQ들에 대한 Max/Min_Avg 값을 구할 수 있으므로, 계산 시간 상의 오버헤드도 크지 않다. 따라서 여러 개의 MSQ를 사용하면 상대적으로 적은 공간을 사용하면서도 질의 처리 면에서 효과적인 성능 향상을 기대할 수 있다.

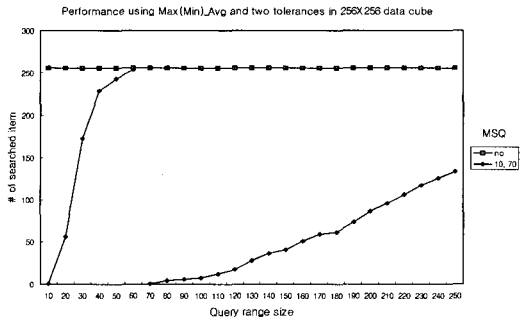

그림 10 256×256 데이타 큐브(두 개의 MSQ 사용)

5. 결론 및 추후 연구 과제

최근 들어 조직의 의사 결정 과정의 중요성이 부각됨에 따라, OLAP 시스템에 의한 빠르고 직접적인 데이타 분석의 필요성이 대두되고 있다. OLAP 시스템은 빠른 질의 응답 속도를 보장하면서, 데이타 웨어하우스 등에 저장된 데이타에 대한 요약되고 압축된 정보를 제공할 수 있는 의사 결정 지원 시스템의 핵심 요소이다. 일반적으로 OLAP 시스템에서는 빠른 질의 응답 속도를 보장하기 위해, 빈번히 사용되는 질의에 대한 선계산을 비작동 시간에 수행하고, 실제 질의 처리 시에는 선계산 결과를 활용해 질의 처리 시간을 줄이는 기법이 보편적으로 사용된다.

현재까지의 OLAP 분야에서의 질의 처리에 관한 연구는 주로 지정된 영역에 대하여 집단 연산의 결과를 구하는 영역 질의에 한정되어 왔다. 그러나, OLAP 시스템에서의 보다 효과적인 의사 결정 지원을 위해서는 의사 결정에 도움을 줄 수 있는 다양한 질의 유형에 대한 처리 기법을 제공하는 것이 필수적이다. 실제 자료 분석 과정에서는 기존의 영역 질의 뿐만 아니라, 특정한 조건을 만족하는 데이타 큐브 상의 영역 및 해당 영역에서의 계산 결과를 구할 필요성이 존재한다.

본 논문에서는 이와 같은 확장된 형태의 영역 질의의 개념을 정의하고, 이에 대한 처리 기법을 제안하였다. 본 논문에서 다루는 질의는 영역의 크기만이 주어질 때, 연속적인 두 개의 집단 연산으로 명시된 특정 조건을 만족하는 영역과 해당 영역에서의 집단 연산 결과 값을 구하는 형태의 질의이다. 본 논문에서는 데이타 큐브 상의 각 차원 원소에 대하여, 질의에 명시된 조건을 구성하는 집단 연산의 결과값이 취할 수 있는 범위를 예측하는 방법을 제안하였다. 즉, 이러한 예측값을 활용하여 실제 질의 처리 시 검색될 데이타의 양을 줄임으로써, 질의 응답 속도를 향상 시킨다.

본 논문에서는 실험을 통하여, 제안된 기법이 질의 처리 시 좋은 성능을 나타냄을 보이고 성능 향상을 위한 시스템 설정 방안도 제시하였다. 본 논문에서 제안된 기법은 2차원 데이타 큐브에 대하여 효과적으로 수행된다. 3차원 이상의 데이타 큐브에 대한 확장된 영역 질의의 의미와 질의 처리 기법의 적용 방법 등에 대해서는 좀 더 많은 연구가 필요하다.

참 고 문 헌

[1] E.F. Codd, S.B. Codd, C.T. Salley. "Providing olap(on-line analytical processing) to user-analysts: An it mandate". Technical report, 1993.

[2] S. Chaudhuri and U. Dayal. "An Overview of Data Warehousing and OLAP Technology". ACM

SIGMOD Record, Vol.26, No.1, pp.65-74, 1997.
[3] "The Case for Relational OLAP". Technical report, MicroStrategy White Paper, 1995.
[4] E. Baralis and S. Paraboschi and E. Teniente. "Materialized view selection in a multidimensional database". In Proceedings of the 23rd VLDB Conference, pp.156-165, 1997.
[5] H. Gupta and V. Harinaarayan and A. Rajaraman and J.D. Ullman. "Index selection for olap". In Proceedings of the 13th ICDE, pp.208-219, 1997.
[6] V. Harinarayan and A. Rajaraman and J.D. Ullman. "Implementing data cubes efficiently". In Proceedings of the ACM SIGMOD Conference, pp.205-227, 1996.
[7] A. Shukla and P.M. Deshpande and J.F. Naughton and K. Ramasamy. "Storage estimation for multidimensional aggregates in the presence of hierarchies". In Proceedings of the 22nd VLDB Conference, pp.522-531, 1996.
[8] S. Sarawagi, R. Agarwal, A. Gupta. "Modeling multidimensional databases". In Proceedings of the 13th ICDE, pp.232-243, 1997.
[9] S. Agarwal, R. Agrawal, P.M. Deshpande, A. Gupta, J.F. Naughton, R. Ramakrishnan, S. Sarawagi. "On the computation of multidimensional aggregates", In Proceedings of the 22nd VLDB Conference, pp.606-521, 1996.
[10] J. Gray, A. Bosworth, A. Layman, H. Pirahesh. "Data cube: A relational aggregation operator generalizing group-by, cross-tabs and sub-totals". In Proceedings of the ACM SIGMOD Conference, pp.152-159, 1997.
[11] Y. Zhao, P.M. Deshpande, J.F. Naughton. "An array-based algorithm for simultaneous multidimensional aggregates". In Proceedings of the ACM SIGMOD Conference, pp.159-170, 1997.
[12] C.T. Ho, R. Agrawal, N. Megiddo, R. Srikant. "Range Queries in OLAP Data Cubes". In Proceedings ACM SIGMOD International Conference on Management of Data, pp.73-88, 1997.

정 희 정

1997년 한국과학기술원 전산학과 졸업(학사). 1999년 한국과학기술원 전산학과 졸업(공학석사). 1999년 ~ 현재 한국전자통신연구원 연구원. 관심분야는 OLAP, 데이타 웨어하우스, 데이타 마이닝 등임.

김 동 욱

1992년 한국과학기술대학 전산학과 학사. 1994년 한국과학기술원 전산학과 석사. 1999년 한국과학기술원 전산학과 졸업(공학박사). 1999년 3월 ~ 현재 새롬 소프트재직. 관심분야는 능동데이타베이스 시스템, 고성능 저장 시스템 등임.

김 종 수

1995년 한국과학기술원 전산학과 학사. 1997년 한국과학기술원 전산학과 석사. 1997년 ~ 현재 한국과학기술원 전산학과 박사과정 재학중. 관심분야는 시간지원 데이타베이스, OLAP

이 윤 준

1977년 서울대학교 계산통계학과 졸업. 1979년 한국과학기술원 전산학과에서 석사학위 취득. 1983년 France, INPGEN-SIMAG에서 박사학위 취득. 1983년 ~ 1984년 France, IMAG 연구원. 1984년 ~ 현재 한국과학기술원 전산학과 부교수. 1989년 MCC(미) 초빙연구원. 1990년 CRIN(불) 객원교수. 관심분야는 데이타베이스 시스템, 정보검색, 실시간 데이타베이스 등임.

김 명 호

1982년 서울대학교 컴퓨터공학과 학사. 1984년 서울대학교 컴퓨터공학과 석사. 1989년 MICHIGAN 주립대 연구원. 1989년 ~ 1993년 KAIST 부교수. 1993년 ~ 현재 KAIST 부교수. 1992년 ~ 1993년 개방형 컴퓨터 통신 연구회(OSIA) 분산 트랜잭션 처리 분과위(TG-TP)의장. 1993년 ~ 1994년 한국통신기술협회(TTA) 분산 트랜잭션처리 실무 위원회 의장. 관심분야는 분산데이타베이스, 분산트랜잭션, 멀티미디어 데이타베이스