韓國 컴퓨터情報學會 論文誌
第12卷 第4號, 2007. 9.

2007-12-4-3-3

장바구니 분석용 OLAP 큐브 구조의 설계

유 한 주*, 최 인 수**

Designing OLAP Cube Structures for Market Basket Analysis

Han-Ju Yu *, In-Soo Choi **

요 약

고객이 제품을 구매할 때에는 항시 구매패턴이 생기기 마련인데, 이러한 구매패턴을 찾아 나가는 과정을 장바구니 분석이라 부른다. 장바구니 분석은 Microsoft Association Algorithm에서는 두 가지 단계로 구성되어 있는데, 첫 번째 단계는 빈발항목집합을 찾아내는 과정이고, 두 번째 단계는 첫 번째 단계에서 찾은 빈발항목집합을 근거로 하여 이들의 중요도를 비교하는 단순한 계산과정이다. 빈발항목집합을 찾아내는 첫 번째 단계는 장바구니 분석에 있어서 핵심부분임에도 불구하고, OLAP 큐브에 적용할 때에는 추적분석이 불가능해지거나 허구의 빈발항목집합이 생성되는 등 여러 문제가 발생하게 된다. 본 연구에서는 장바구니 분석에 있어서 추적분석을 가능하게 하고 실제의 빈발항목집합만을 생성시키는 새로운 OLAP 큐브 구조의 설계법을 제안하고 있다.

Abstract

Every purchase a customer makes builds patterns about how products are purchased together. The process of finding these patterns, called market basket analysis, is composed of two steps in the Microsoft Association Algorithm. The first step is to find frequent item-sets. The second step which requires much less time than the first step does is to generate association rules based on frequent item-sets. Even though the first step, finding frequent item-sets, is the core part of market basket analysis, when applied to Online Analytical Processing(OLAP) cubes it always raises several points such as longitudinal analysis becomes impossible and many unpractical transactions are built up. In this paper, a new OLAP cube structures designing method which makes longitudinal analysis be possible and also makes only real customers' purchase patterns be identified is proposed for market basket analysis.

▸ Keyword : 장바구니 분석(Market Basket Analysis), 빈발항목집합(Frequent Item-sets), 큐브구조(Cube Structures), 추적분석(Longitudinal Analysis)

• 제1저자 : 유한주
• 접수일 : 2007. 8.2, 심사일 : 2007. 8.3, 심사완료일 : 2007. 8.23
* 숭실대학교 산업·정보시스템공학과 박사과정
** 숭실대학교 산업·정보시스템공학과 교수
※ 본 연구는 숭실대학교 교내 연구비 지원으로 이루어졌음.

I. 서 론

기업이 고객과 상호교류하는 방식은 과거에 비해 많이 바뀌고 있다. 오늘날은 더 이상 고객과의 지속적인 거래가 보장되지 않는 시대이다. 이 때문에 기업은 고객을 더 잘 이해하고 고객이 원하는 것에 더 빨리 대응해야 할 필요성을 어느 때보다도 더 절실하게 느끼는 것이다. 고객관계관리(customer relationship management)는 기업과 고객 간의 상호교류를 관리하는 프로세스를 말한다. 현대의 기업에서는 고객의 구매형태에 대한 정보를 담고 있는 자사의 거래 관련 데이터베이스가 아주 중요시 되고 있다. 왜냐하면 구매형태에 관한 정보가 기업이 어떠한 서비스를 고객에게 제공해야 하는지를 결정하는데 있어서 가치가 있는 정보이기 때문이다.

OLAP(On-line Analytical Processing)과 데이터마이닝(data mining)은 고객관계관리를 위한 상호보완적인 기술이다[5,10,12]. OLAP은 사용자가 의사결정에 필요한 지식을 찾아내기 위해 대량의 비즈니스 데이터를 쉽게 분석할 수 있도록 도와주는 데이터베이스 도구이며, 사용자 위주의 관점으로 설계된 계층형 데이터베이스이다[1,7,11,16].

OLAP 큐브는 잘 구축된 데이터베이스라고 말할 수 있다. 관계형데이터베이스를 기반으로 하는 큐브에는 판매동향, 제품조합, 고객세분화와 같은 감추어진 패턴들이 포함되어 있다. 그러나 차원에는 수백만개의 구성원이 있고, 사실 테이블에는 수십억개의 거래처리기록이 있는 등 현대의 OLAP 큐브는 거대하다. 이러한 거대 OLAP 큐브에서 유용한 정보를 찾아내기는 무척 힘겹게 된다. 따라서 이렇게 거대한 큐브로부터 고객의 구매 패턴과 같은 CRM적인 정보를 찾기 위해서는 데이터마이닝 기술을 활용할 필요가 생기는 것이다[10,12,14,15]. OLAP과 데이터마이닝이 상호보완적이라고 전술한 것이 바로 이 때문이다.

데이터마이닝은 수집한 데이터에서 아직 찾아내지 못한 가치 있는 패턴을 찾는 것을 목적으로 하는 모든 기술을 말한다. 데이터마이닝 기술 중 연관규칙이라고 불리는 장바구니분석(market basket analysis)은 거대 데이터에서 특정 연관규칙을 찾아내는 기술을 말한다. 장바구니 분석은 판매되고 있는 제품들 중에 동시에 구매될 가능성이 높은 제품들을 파악하는 분석 방법이다. 기업은 동시구매성이 높은 제품을 고객에게 효과적으로 제시함으로써, 고객이 보다 편리하게 제품을 선정하고 구매할 수 있도록 도와 줄 수 있다. 또한 경우에 따라서는 동시구매가능성이 높은 제품의 특성을 잘 고찰하여 신제품을 기획하고, 고객에게 제공함으로써 고객만족과 고객관계를 향상시킬 수 있게 된다[5,13,17]. 그러나 앞구성원 기반으로 설계되어 있는 현재의 OLAP 큐브구조를 바탕으로 하여 장바구니분석이라고 하는 데이터마이닝을 수행할 경우 이상에서 언급한 CRM적인 목표를 달성하는데 어려움이 있음을 저자들은 깨닫고 있다. 구체적으로 마이크로소프트 연관알고리즘(Microsoft Association Algorithm)을 현 OLAP 큐브구조에 적용시킬 경우 CRM에서 중요시 되는 추적분석(longitudinal analysis)을 할 수 없게된다는 사실을 확인하고 있다. 따라서 본 연구에서는 장바구니분석을 올바로 수행하기 위해 OLAP 큐브구조를 새롭게 설계하는 방안을 제시하고자 하는 데에 목적을 두고 있다.

II. 기존 OLAP에서의 큐브구조와 장바구니 분석

2.1 기존 OLAP 큐브 구조

OLAP은 사용자가 의사결정에 필요한 지식을 찾아내기 위해 대량의 비즈니스 데이터를 쉽게 분석할 수 있도록 도와주는 차원과 사실테이블로 구성된 큐브에 기반 하는 분석 도구이다. 다차원 데이터베이스의 기본적인 구조인 스타 스키마에서 다차원 모델은 사실(fact) 데이터와 차원(dimension) 데이터로 구성되며, 사실 데이터로 구성된 사실 테이블을 중심으로 여러 개의 차원 테이블이 뻗어져 나오는 형태를 취하고 있다[1,2,6,10]. 예제로 활용할 구매내역서를 살펴보면 (그림 1)과 같다. 첫 번째 구매내역(transaction)에는 1번 고객(customer)이 2006년 1월 1일에 제품집합(itemset) ab(제품 a를 2개, 제품 b를 1개)를 동시에 구매했으며, 총 지불액(extended price)이 410이라는 정보가 나타나 있다.

Transaction No.	Customer	Date	Itemset	Quantity	Extended Price
1	1	2006.01.01	ab	a=2, b=1	410
2	1	2006.01.02	c	c=2	200
3	2	2006.01.02	ca	c=1, a=1	220
4	3	2006.01.03	b	b=2	300
5	3	2006.01.04	b	b=2	300

그림 1. 구매내역
Figure 1. An example of transaction details

(그림 1)의 구매내역을 바탕으로 기존 방법에 따라 OLAP 큐브 스키마를 작성해보면 (그림 2)와 같다[1,6,7]. (그림 1)의 첫 번째 구매내역 즉, 제품 a와 b를 동시에 구

매한 내역을 사실 테이블에서 두 개의 행으로 구분하여 표현된 것을 확인할 수 있을 것이다. 세 번째 c와 e를 동시에 구매한 내역 역시 두 개의 별도의 행으로 구성되어 있고 나머지 세 개의 구매내역 즉, 두 번째 구내내역 c와 네 번째 b, 다섯 번째 b를 구매한 내역은 각각 하나의 행으로 구성되어 전체 5개의 구매내역이 7개의 행으로 구성되어 있음을 확인할 수 있을 것이다.

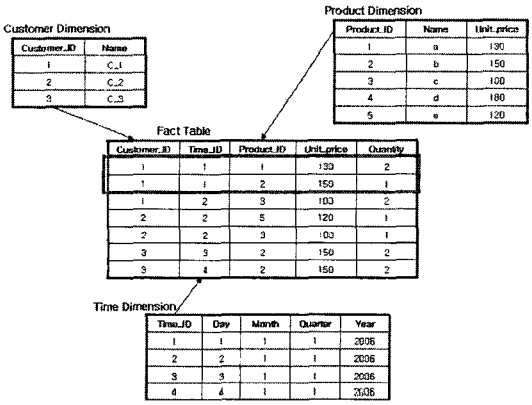


그림 2. 스타스키마
Figure 2. Star schema

(그림 2)의 스타 스키마를 바탕으로 MS SQL Server 2005 Analysis Service를 이용하여 OLAP 큐브를 작성해보면 (그림 3)과 같다.

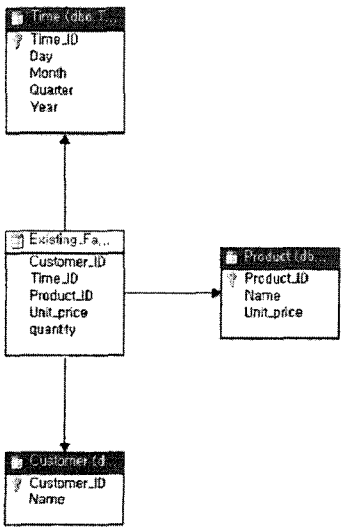


그림 3. 기존 OLAP 큐브 구조
Figure 3. Existing OLAP cube structures

2.2 장바구니 분석

대형 할인점, 백화점 또는 인터넷 쇼핑몰에서는 고객이 구매한 수많은 구매내역을 모아 두었다가 이를 분석하여 마케팅 활동에 활용하는 것이 보통이다. 연관성 분석이라고도 부르는 장바구니 분석(market basket analysis)은 매장에서 제품을 진열하거나 끼워 팔기에 이용 가능하며 주로 고객의 거래 데이터를 분석하여 제품 사이의 연관성을 알아내고 있다[13,15]. 연관규칙은 보통 a→b로 나타낸다. a→b는 어떤 고객이 a 라는 제품을 구매하면 b 라는 제품도 구매하게 된다는 것을 의미한다. 여기서, a는 반드시 하나의 제품만을 가리키지는 않으며 여러 가지 제품도 가리키는 즉, 제품조합이 되게 된다. 그러나 b는 대부분의 경우 한 가지 제품이 된다. a→b라는 규칙에 관련된 제품을 전체 고객 중에 상당 비율을 차지하는 고객이 구매하고, 동시에 a 제품을 구매한 고객 중에서 상당 비율의 고객이 b 제품을 구매해야만 a→b라는 연관규칙을 마케팅 활동에 활용할 수 있는 것이다. 이렇게 찾아낸 연관규칙에 대해 다시 중요도(impotance)값을 계산하여 큰 값을 갖는 연관규칙만을 골라내게 된다. 이러한 장바구니분석은 고객이 구매한 제품과 연관성이 높은 제품을 추천하거나, 백화점이나 소매점에서 효과적인 제품 진열방법을 모색하고자 할 경우, 또는 패키지 제품을 개발하고자 할 경우에 사용할 수 있다.

본 연구에서는 Microsoft SQL Server 2005에서 제공하는 연관 알고리즘(association algo-

rithm)을 이용하여 (그림 1)의 구매내역 예제에 대하여 장바구니 분석을 수행하였다. 마이크로소프트 연관 알고리즘에 의해 발생한 제품집합(항목집합)은 (그림 4)와 같다.

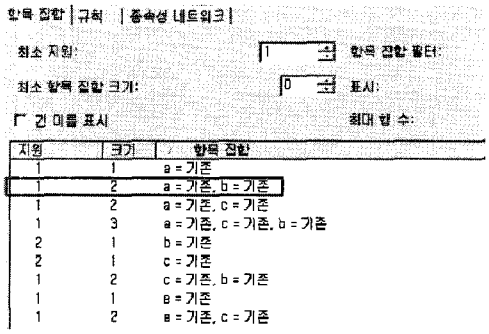


그림 4. 제품집합(항목집합)
Figure 4. Itemsets

(그림 4)에서 지원(support)은 해당 제품집합이 발생하는 횟수이며, 크기는 제품집합을 이루는 제품구성원의 숫자를 말한다. 한 예로, 제품집합 {a, b}는 두 개의 제품으로 이루어졌으며 이 제품집합을 구매하는 즉, 제품 a와 b를 동시에 구매한 구매내역이 1번 있었다는 사실을 나타내고 있다.

제품집합에 대한 지원과 크기를 구했다면 해당 제품집합으로부터 발생하는 연관규칙의 중요도[10]가 얼마나 높은가를 알아보아야 한다. 마이크로소프트 연관규칙에 의해서 구한 각각의 연관규칙의 중요도는 (그림 5)와 같다.

먼저 연관규칙 {b→a}의 확률이 0.5이고 중요도가 0.301이 되는 것을 설명하면 다음과 같다. 중요도를 구하기 위해서는 먼저 각 규칙에 대한 확률을 구하고 구한 확률을 기반으로 중요도를 계산해야 한다. 연관규칙 {b→a}의 중요도를 구하기 위해 필요한 값을 도식화하여 나타내면 (그림 6)과 같다.

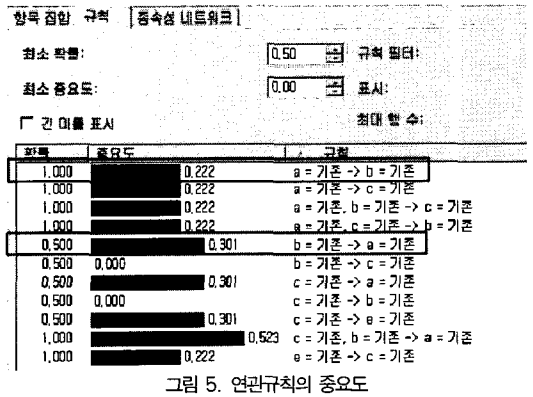


그림 5. 연관규칙의 중요도
Figure 5. Importance of association rules

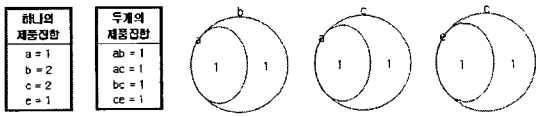


그림 6. 연관규칙 {b→a}의 중요도
Figure 6. Importance of a rule {b→a}

제품집합의 지원 숫자가 a=1, b=2, ab=1 이므로, (그림 6)의 오른쪽 집합의 형태를 취하게 되는 것을 확인할 수 있을 것이다. 또한 a=1, c=2, ac=1 이기 때문에 a가 c에도 속한다는 것을 알 수 있으며, c=2, e=1, ce=1 이므로, e가 c에 속한다는 것 역시 알 수 있을 것이다. (그림 4)와 (그림 6)의 데이터를 기반으로 연관규칙 {b→a}의 중요도(importance{b→a})를 구해보면 다음과 같다. 먼저 b를 구매하면서 a를 동시에 구매할 확률 즉, P(b→a)를 구한다. P(b→a)는 b와 a를 동시에 구매한 제품집합(ab=1개)의 숫자를 b를 구매한 제품집합(b=2개)의 숫자로 나눈 값이 된다(식 (1)참조).

$$P(b\rightarrow a)=\frac{1}{2}=0.5 \quad \cdots\cdots (1)$$

다음으로 이 확률을 b를 구매하지 않으면서 a를 구매할 확률로 나누면 된다. 즉, $P(\bar{b}\rightarrow a)$는 b를 구매하지 않았을 때 a를 구매한 경우(a=1)를 b를 구매하지 않는 제품집합의 수 4개(a=1, c=2, e=1 : 4)로 나눈 값이다(식 (2) 참조).

$$Importance(b\rightarrow a)=\log\frac{P(b\rightarrow a)}{P(\tilde{b}\rightarrow a)}$$

$$=\log\frac{\frac{1}{2}}{\frac{1}{4}}=\log\frac{0.5}{0.25}=\log 2=0.301 \quad \cdots\cdots (2)$$

연관규칙{a→b}의 확률이 1이고, 중요도가 0.222가 되는 것을 한번 더 확인해 보기로 한다. (그림 4)와 (그림 6)의 데이터를 기반으로 연관규칙{a→b}의 중요도(importance {a→b})를 구해보면 다음과 같다. 먼저 a를 구매하면서 b를 동시에 구매할 확률 즉, P(a→b)를 구한다. P(a→b)는 a와 b를 동시에 구매한 제품집합(ab=1)을 a를 구매한 제품집합(a=1)으로 나눈 값이 된다(식 (3)참조).

$$P(a\rightarrow b)=\frac{1}{1}=1 \quad \cdots\cdots (3)$$

다음으로 이 확률을 a를 구매하지 않으면서 b를 구매할 확률로 나누면 된다. 즉, $P(\bar{a}\rightarrow b)$는 a를 구매하지 않았을 때 b를 구매한 경우(b=2, bc=1 : 3)를 a를 구매하지 않는 제품집합(b=2, c=2, e=1 : 5)으로 나눈 값이다(식 (4) 참조). 여기서 a를 구매하지 않았을 때 b를 구매한 경우에서 ab=1이 들어 있으나 이는 b에 속하기 때문에 고려되지 않는다.

$$Importance(a\rightarrow b)=\log\frac{P(a\rightarrow b)}{P(\tilde{a}\rightarrow b)}$$

$$= \log \frac{1}{\frac{3}{5}} = \log 1.67 = 0.222 \quad \cdots\cdots (4)$$

이상의 결과에서 연관규칙{a→b}의 중요도 보다 연관규칙{b→a}의 중요도가 더 높은 것을 확인할 수 있다. 따라서 {b→a}라는 연관규칙을 좀 더 신중하게 고려해야 한다고 본다. 이처럼 마이크로소프트에서 제공하는 연관규칙 알고리즘을 이용하면 각 제품집합간의 연관성을 확인 할 수 있게 된다. 업체들은 이 정보를 마케팅에 활용하여 보다 좋은 기대효과를 얻을 수 있을 것이다. 그러나 마이크로소프트 연관규칙 알고리즘을 적용시킬 때에는 여러 문제가 생기게 된다. 고객이 구매하지도 않은 구매내역을 실제로 구매한 것처럼 만들어 낸다는 것이다. 이러한 연유로 잘못된 분석을 하게 된다. 또 하나의 주요 문제점은 정해진 기간 동안 개인 고객이 여러 번 구매활동을 하더라도 이 알고리즘에서는 한번밖에 구매활동을 하지 않게 된다고 보는 것이다. 이러한 점 때문에 CRM에서 중요시 되는 추적분석이 불가능해지는 것이다. 3장에서 이상의 문제점에 대해서 자세히 알아보도록 하겠다.

III. 장바구니분석에 있어서의 현 OLAP 큐브의 문제점

3.1 마이크로소프트 연관규칙을 사용한 구매제품 조합 생성

구매내역 데이터베이스에서 연관규칙을 찾는다는 것은 사용자가 명시한 최소 지원과 최소 신뢰도보다 큰 지원과 신뢰도를 갖는 제품집합을 찾는 것이라고 말할 수 있다. 최소 지원보다 큰 지원을 갖는 제품집합은 자주 팔리는 빈발제품집합이라고 할 수 있다. 우리가 최소 지원에 관심이 있는 이유는 구매내역 데이터베이스에서 관심 있을 정도로 빈번하게 일어나는 제품집합만을 고려해야 하기 때문이다.

일반적으로 연관규칙 탐사는 빈발제품집합을 찾고, 빈발제품집합을 이용해 규칙을 생성하는 두 단계로 나누어진다. 마이크로소프트 연관규칙 알고리즘을 예로 들어 설명하면, 데이터베이스를 읽어가면서 각 단계마다 찾아낸 빈발제품집합이 다음 단계의 후보제품집합을 생성하게 된다. (그림 3)의 큐브 데이터는 (그림 7)과 같은데, 이 데이터가 후보제품집합을 생성하는 기초자료가 된다.

Name	Unit Price	Quantity	Existing Fact 카운트
a	130	2	1
b	450	5	3
c	200	3	2
e	120	1	1
총합계	900	11	7

그림 7. 큐브 데이터
Figure 7. Cube data

후보 제품집합생성 작업은 제품집합의 크기가 1인 것부터 시작한다. 제품집합의 크기가 1인 것으로 재구성한 표가 (그림 8)의 첫 번째 표이다. (그림 8)의 첫 번째 표는 (그림 7)의 데이터를 가져와서 재구성한 것인데 (그림 7)에서의 b의 카운트가 3인 것이 (그림 8)에서는 2로 바뀐 것을 알 수 있을 것이다. 여기에 대해서는 3.2에서 자세히 설명하기로 한다.

(그림 8)에서 a의 카운트가 1이라는 숫자는 a가 들어가 있는 구매내역이 한번이라는 의미를 가진다. 이 중 최소지원이 1미만인 제품집합을 제거하고 나머지 제품집합을 가지고 다음단계인 크기 2인 제품집합의 후보제품집합을 생성한다. 크기 2인 제품집합의 내용은 두 번째 그림과 같다. 최소지원이 1미만인 후보제품집합이 없으므로, 이 후보제품집합을 가지고 크기가 3인 후보제품집합을 만들면 세 번째 그림과 같다. 최소지원이 1미만인 {a,b,e}와 {b,c,e}를 제거하면 후보제품집합은 {a,b,c}가 된다. 더 이상의 후보제품집합을 얻어낼 수 없게 되면 알고리즘은 종료된다.

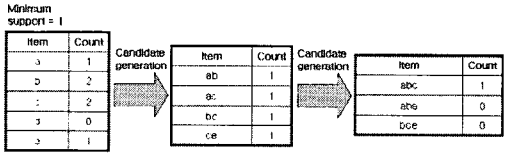


그림 8. 연관규칙 탐사
Figure 8. Finding association rules

(그림 8)에서 볼 것 같으면 크기 1인 유효 제품집합은 {a},{b},{c},{e} 4개가 발생되고, 크기 2인 유효 제품집합은 {a,b},{a,c},{b,c},{c,e} 4개가 생성되며, 크기 3인 유효 제품집합이 {a,b,c} 1개가 생성됨을 알 수 있다. 이러한 연유로 마이크로스프트 연관규칙을 사용하면 (그림 4)에서와 같은 9개의 제품집합이 생성되는 것이다.

3.2 잎 구성원 기반의 OLAP 큐브구조

현 관계형데이터베이스(relational database) 구조에서는 일반적으로 테이블이 릴레이션(relations)이 되는 것을 원칙으로 하고 있다. 릴레이션이 되기 위한 조건을 살펴보면 (그림 9)와 같다[8].

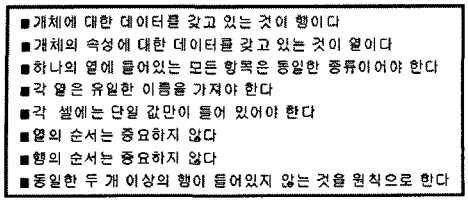


그림 9. 릴레이션으로서의 요구 조건
Figure 9. Characteristics of relations

테이블이 릴레이션이 되기 위해서는, 첫째 테이블의 행들이 개체에 대한 데이터를 저장해야 하고, 테이블의 열들이 그 개체들의 특성에 대한 데이터를 저장해야 한다. 또한 릴레이션에서는 한 열의 모든 값들이 동일한 종류에 속해야 한다. 예를 들어 어떤 릴레이션에서 첫 번째 행의 두 번째 열이 FirstName을 가진다면, 그 릴레이션에서 모든 행의 두 번째 열은 FirstName을 가져야 한다. 또한 열의 이름들은 유일해야 한다. 즉, 동일한 릴레이션에 속한 두 열은 같은 이름을 가질 수 없다. 릴레이션의 각 셀(cell)은 단일한 값 또는 항목만을 가지며 다중 항목은 허용되지 않는다. 그리고 행의 순서와 열의 순서는 중요하지 않다. 마지막으로 주어진 테이블이 릴레이션이 되기 위해서는 동일한 행이 존재하면 안 된다. SQL문장은 중복 행을 가진 테이블을 생성하기도 한다. 그러나 이러한 행의 중복은 SQL 조작문의 결과로서만 발생하는 것이지, 데이터베이스에 저장되도록 설계된 테이블에서는 결코 행이 중복되어서는 안 된다[8]. 관계형데이터베이스를 기반으로 하는 현 OLAP 큐브 구조 역시 이상의 릴레이션의 원칙을 따르게 되는 것이다.

다차원 데이터베이스의 기본적인 구조인 스타 스키마에서 다차원 모델은 2장에서 언급한바와 같이 사실 데이터와 차원 데이터로 구성된다. 단순한 형태의 스타스키마는 (그림 2)에서처럼 하나의 정규화 된 사실 테이블과 이에 연결된 다수의 차원테이블로 구성되는데 이런 스타 스키마는 실제 사용자가 쉽게 이해할 수 있고, 단순하게 모델링 할 수 있게 때문에 결과적으로 사용자가 차원에 대해 쉽게 질의를 작성할 수 있으며, 데이터베이스는 사용자의 질의에 빠른 속도로 응답할 수 있게 되는 것이다. 하나의 사실 데이터와 여러 개의 차원 데이터는 각기 차원 데이터의 기본키(primary key)로 연결되며, 하나의 사실 데이터는 여러개의 관련된 차원의 기본키의 조합 즉, 복합키로써 구별된다. 여기서 기본키의 조합이란 외래키(foreign key)의 조합 즉, 각 차원의 잎구성원(leaf member)의 조합이 된다. 각 행이 가지는 의미는 이미 2장에서 밝힌바 있다.

(그림 2)에 대한 LC(Located Contents) 스키마를 살펴보면 식 (5)와 같다[11].

$$(\text{Customer.} \otimes \text{Product.} \otimes \text{Time.}) \sim \text{Unit\_Price, Quantity} \quad \cdots\cdots (5)$$

Customer.은 (그림 2)에서의 고객차원의 각각의 구성원 즉, C_1, C_2, C_3를 말한다. Product.은 제품 5개의 제품 각각을, Time.은 4개의 시간을 말한다. (Customer. ⊗ Product. ⊗ Time.)은 60 조합(Customer 3 × Product 5 × Time 4) 각각을 의미하고, ~은 이 60개의 각각의 조합에 Unit_Price, Quantity가 매치된다는 것을 의미한다. 어느 일정한 기간 동안 판매된 제품을 가지고 그 제품을 구매한 고객에 대한 분석과 각 고객의 구매 패턴을 분석하는 OLAP과 데이터마이닝에서는 분석기간이 중요한 분석의 요소로 자리 잡고 있다. 식 (5)의 LC 스키마로 구성되어 있는 OLAP 큐브를 마이크로소프트 연관규칙을 사용하여 장바구니 분석을 수행하게 되면 식 (6)의 LC 스키마 구조를 가지게 된다.

$$(\text{Customer.} \otimes \text{Product.} \otimes \text{Time.all}) \sim \text{Unit\_Price, Quantity} \quad \cdots\cdots (6)$$

식 (6)에서의 Customer.과 Product. 은 식 (5)에서의 Customer.과 Product.과 같다. Time.all은 식 (5)에서의 Time.이 의미하는 것하고는 다른 의미를 가진다. Time.이 의미하는 4개의 시간이 아니라, Time 차원의 최상위 수준인 Year 수준 자체를 의미한다. 다시 말하자면 각 날짜별 데이터가 아니라 각 날짜의 데이터들이 속해있는 최상위 수준인 2006년 이라는 하나의 기간을 의미한다. 따라서 식 (6)의 (Customer. ⊗ Product. ⊗ Time.all)은 15 조합(Customer 3 × Product 5 × Time 1)을 가지게 된다. ~은 이 15개의 각각의 조합에 Unit_Price, Quantity가 매치된다는 것을 의미한다. 따라서 식 (6)이 궁극적으론 다음의 식 (7)과 같은 의미를 갖는다고 볼 수 있다.

$$(\text{Customer.} \otimes \text{Product.}) \sim \text{Unit\_Price, Quantity} \quad \cdots\cdots (7)$$

식 (7)에서의 Customer.과 Product.은 식 (5)와 식 (6)에서의 의미와 같다. 식 (7) 역시 15개의 조합(Customer 3 × Product 5)을 가진다. 그렇다고 해서 식 (6)이 식 (7)과 같다라고 말할 수는 없다. 두 식에는 그 의미에서부터가 큰 차이를 가지고 있다. Time.all이 들어가 있는 식 (6)은 시간에 따라 결과가 변할 수 있다는 시간적

분석축의 의미를 내포하고 있지만, 식 (7)처럼 시간에 대한 차원이 존재하지 않는 스키마는 시간에 대한 분석축을 고려하지 않는다는 말이 된다.

현 OLAP 체제 하에서 식 (5)로서 표현되었던 큐브 데이터를 마이크로소프트 연관규칙을 사용하여 장바구니분석을 수행하면 식 (6)과 식 (7)에서와 같은 결과를 산출하게 되는데 현재 이 방식에는 다음과 같은 큰 두 가지 문제점이 발생하게 된다. 첫 번째 문제점은 일정 기간 동안 동일 고객이 동일 제품을 시기를 달리해 가면서 여러 번 구매한 경우에 있어서, 식 (5)를 사용할 것 같으면 구매시기별로 각각 다른 기본키의 조합이 생기지만, 구매시기에 대한 분석점이 하나가 되는 식 (6)이나 구매시기에 대한 분석점을 고려하지 않는 식 (7)을 사용할 것 같으면 단지 한 개의 기본키 조합밖에 생기지 않는다는 것이다. 이점을 (그림 2)를 예를 들어 설명하기로 한다.

(그림 2)에서는 1번 고객이 1월 1일에 제품 b를 1개, 3번 고객이 1월 3일에 제품 b를 2개, 그리고 역시 3번 고객이 1월 4일에 제품 b를 2개 구매한 것으로 되어 있다. 이들 구매내역에 대해 MS 연관규칙을 사용하여 장바구니분석을 수행해 보면 (그림 4)와, (그림 6)의 왼쪽 두 개의 그림에서와 같은 결과를 얻게 된다. 실제로는 b를 3번(1번 고객이 1번, 3번 고객이 2번)구매 하였으나, (그림 6)에서는 b=2로 되어있다. 이는 1번 고객이 한번 제품 b를 구매하였고, 3번 고객이 제품 b를 한번 구매 하였다고 해석한 결과로 얻어진 값이다. 즉, 3번 고객의 제품 b 구매에 대한 기본키 조합이 구매시기를 고려하면 두 개가 되는데 비해서, 구매시기를 고려하지 않으면 한 개가 된다는 뜻이다. 두 번째 문제점은 동일 고객이 여러 시간에 걸쳐 여러개의 다른 제품을 구매하였다고 하더라도 현재의 방식은 시간에 대한 분석점을 고려하지 않기 때문에 고객이 한 번에 여러 제품을 구매한 것으로 취급하게 된다. 이는 결과적으로 사실에는 없는 제품 조합들을 생성하게 되는 원인이 된다. (그림 1)에서 볼 것 같으면 1번 고객이 1월 1일에 제품 a와 b를 동시에 구매하였고, 1월 2일 제품 c를 구매한 것으로 되어 있는데, MS Analysis Service 체제에서는 이러한 사실을 (그림 2)의 사실 테이블에서와 같이 3개의 행으로 나타내게 된다. 시간차원을 고려하지 않는 MS 연관규칙을 사용하여 구매제품 조합을 생성해 보면, a, b, c, ab, ac, bc, abc의 7개 제품조합이 생기게 된다. 실제로는 ab와 c의 2개 제품조합 밖에 없었으나 (그림 4)에서와 같이 허구의 5개 제품조합(a, b, ac, bc, abc)이 생기게 되는 것이다. 이상의 두 가지 문제점 때문에 구매내역이 없어지기도 하고 구매내역이 생기기도 하는 정보손실의 상황하에서 분석가는 분석을 할 수 밖에 없게 되는 것이다.

OLAP과 데이터마이닝에서는 추적분석이 아주 중요한 부분을 차지하고 있다. 추적분석이란 고객의 활동 정보를 종합적으로 수집, 분석하여 기업의 마케팅 담당자들에게 보다 정교하고 종합적인 고객의 이용정보를 제공하는 프로세스를 말한다. 이 추적분석을 수행하기 위해서는 시간적인 속성이 필요하다. 고객이 어느 시기에 어느 제품을 선호하는지를 시간주기적으로 분석하여 마케팅에 활용할 수 있어야 한다. 그러나 현재 관계형 데이터베이스를 기반으로 하는 OLAP 큐브를 마이닝하고자 할 때에는 이상에서 언급한바와 같이 일정 분석 기간 내에서 한명씩의 고객만 존재하고, 이 고객들의 구매동향은 변하지 않는다고 하는 잘못 해석되는 상황이 발생하기 때문에 추적분석이 불가능해지는 것이다.

이상의 문제점이 생기게 되는 출발점은 현재 OLAP이 잎구성원 기반으로 설계되어 있다는 점이라고 볼 수 있다. 구체적으로 (그림 7)에서와 같이 큐브 데이터는 잎구성원 a, b, c, e별로 집계되어 있고, 잎구성원간의 조합별로는, 예를 들면 {a,b}, {a,b,c}별로는 집계할 수 없게끔 설계되어 있는 OLAP의 큐브구조 때문에 이상 언급한 모든 문제가 야기된다고 생각한다.

IV. 새로운 OLAP 큐브 구조 설계

4.1 제품집합 기반의 OLAP 큐브 생성 알고리즘

3장에서 현재 OLAP 큐브 구조는 잎구성원 기반으로 설계되어 있기 때문에 구매한 제품집합에 대한 연관성을 확인할 수 있는 방법이 없다고 설명한 바 있다.

먼저 본 연구에서 정의하고 있는 잎구성원 기반, 잎구성원 자체 기반, 잎구성원상호조합 기반이란 세 가지에 대해서 설명하고자 한다. a, b, c, d, e 의 다섯 제품이 있다면 제품차원의 구성원을 a, b, c, d, e 다섯으로 삼는 것을 잎구성원 기반 설계라 부르고, a, b, c, d, e 다섯 제품을 각기 4개들이(4pack)와 8개들이(8pack)로 묶어서 판매하는 경우 a.4Pack, a.8Pack, ... , e.4Pack, e.8Pack 등으로 구성원을 각기 4개들이, 8개들이 식으로 10개로 삼는 것을 잎구성원자체조합 기반의 설계라고 정의한다. 만약에 a 제품과 b 제품을 묶어서 판매하는 등 여러 제품을 서로 묶어서 판매하는 경우, 예로 a, ab, bc, de 등을 구성원으로 삼는 것을 잎구성원상호조합 기반의 설계로 정의하고 있다. 잎구성원 기반

설계에 대해서는 2, 3장에서 설명한바 있기에, 본 난에서는 잎구성원 자체조합 설계부터 설명하기로 한다.

어떤 회사에서 2개의 제품 a, b를 생산하고 있으며, 각 제품에 대해 4개들이, 8개들이의 두 가지 다른 패키지를 만들어 판매하고 있다고 하자. 현 OLAP 체제하에서는 일반적으로 이에 대해 다음의 3가지 방법으로 큐브설계를 하고 있다[2,11]. 첫 번째 방법은 변수 차원으로 이 문제를 다루는 방법이다. 모든 제품이 4 또는 8-pack로서 팔렸다면, 하나의 차원으로 구별하기보다는 패키지 형태를 (그림 10)과 같이 사실 테이블에 변수 차원으로 취급하는 것이 효율적이라고 하고 있다.

Customer_ID	Time_ID	Product_ID	4Pack	8Pack
1	1	1	2	
1	1	1	1	
2	2	2		1
3	3	1		3
3	3	2	2	

그림 10. 변수 차원으로 해결
Figure 10. Pack type distinction as variables distinction

그러나 이 방법에는 공극(sparsity)을 많이 가지게 된다는 문제점이 생기게 된다. 두 번째 방법은 (그림 11)에서와 같이 제품 차원에 두 가지 패키지 형태에 대해서 정의해 주는 방법이다.

Product_ID	Name	Pack
1	a	a.4Pack
2	a	a.8Pack
3	b	b.4Pack
4	b	b.8Pack

그림 11. 제품 차원으로 해결
Figure 11. Pack type distinction as a product dimension distinction

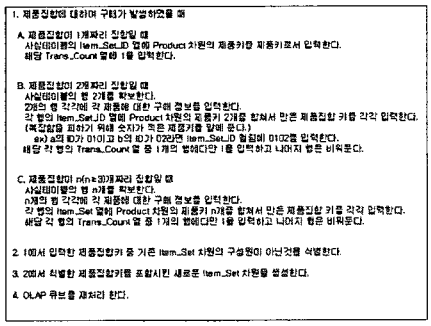


그림 12. 제품집합 생성 알고리즘
Figure 12. The itemsets generating algorithm

이 방법에는 차원의 구성원 숫자가 두 배로 된다는 문제점이 있다. 세 번째 방법은 패키지 형태에 대한 별도의 차원을 생성하는 것이다. (그림 12)는 all 수준에 4-pack과 8-pack이 연결되는 형태를 가진다. 이 방법을 적용하면 변수차원에서의 공극에 대한 문제점을 해결할 수 있고 또한 제품 차원에 패키지 형태를 정의함으로써 생기는 잎구성원의 숫자가 두 배로 늘어나는 문제점을 해결할 수 있다.

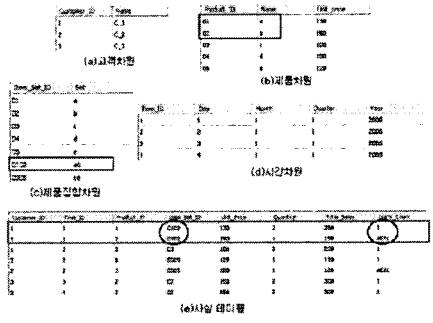
Package Dimension

Pack_ID	Pack
1	4Pack
2	8Pack

그림 13. 새로운 패키지 차원으로 해결
Figure 13. Adding a new package-type dimension

그러나 이상의 방법들은 잎구성원을 기준으로 하는 문제일 경우에만 해결 가능한 방법이다. 예를 들어, 제품 a가 2개, 제품 b가 2개로 구성된 패키지 4개들이 ab에 대한 분석은 이상의 방법으로는 해결할 수 없다. 현 OLAP 큐브 구조에서는 이러한 종류의 문제를 다룰 수 있는 방법이 결여되어 있다[11]. 다시 말하자면 잎구성원 조합에 대한 문제를 해결하는 방법이 관계형데이터베이스를 기반으로 하는 현 OLAP에서는 해결 불가능하다는 것이다. 따라서 본 장에서는 본 연구에서 제시하는 잎구성원간의 연관성을 확인할 수 있는, 잎구성원상호조합에 기반하여 OLAP 큐브를 생성하는 방법에 대하여 설명하고자 한다.

본 연구에서는 잎구성원상호조합을 달리 제품집합이라고 부르고 있는데, 이 제품집합 생성 알고리즘을 살펴보면 (그림 13)에서와 같다. 이 제품집합 생성 알고리즘을 사용하면 잎구성원 자체에 대한 분석뿐만 아니라 잎구성원 간의 연관성 여부도 식별할 수 있게 된다.



14. 알고리즘을 적용한 구매내역 예제
Figure 14. Applying the new algorithm to an example of transaction details

4.2 제품집합 생성 알고리즘의 적용

본 난에서는 4.1에서 제시한 제품집합 알고리즘(이하 알고리즘으로 기록)을 2.1에서 기술한바 있는 구매내역 예제에 적용시켜 보기로 한다. (그림 1)의 구매내역 데이터를 사용하여 알고리즘 1을 실행시켜 보면 (그림 14)와 같은 결과를 얻게 된다.

(그림 1)에서 볼 것 같으면 1번 고객이 제품 a를 2개, 제품 b를 1개 동시에 구매했기 때문에, 알고리즘 1의 B에 의해서 2개의 행을 확보하게 된다. 1번 고객이 제품 a와 b를 동시에 구매한 사실은 (그림 14)에서 2개의 행으로 표현한 것을 확인할 수 있다. (그림 14)의 (e)의 왼쪽 동그라미 안의 Item_Set_ID 열의 제품집합 키가 0102 (제품 a의 ID 01과 제품 b의 ID 02)로 표현되어 있는 것을 확인할 수 있을 것이다. 여기서 Item_Set_ID 열은 제품집합을 식별하는 역할을 하고 있다. 또한 오른쪽 동그라미처럼 Trans_Count 열의 2개의 행중 1개의 행에만 숫자 1이 입력되어 있고 나머지 1개의 행은 널(Null)로 되어 있는 것을 보게 되는데, 이 Trans_Count 열은 구매 횟수를 식별하는 중요한 역할을 한다. 따라서 구매가 일어날 때마다 한 번씩 즉, 하나의 행에다만 1이라는 숫자를 부여하는 것이다[3,4]. 데이터 자체의 존재유무만 따지는 MDX(MultDimensional eXpressions)에서의 Count()와는 달리[9] Trans_Count 열은 데이터 자체의 내용까지 다룰 수 있게 된다.

이상의 차원과 사실테이블을 사용하여 OLAP 큐브를 생성시켜 보면 (그림 15)와 같게 된다. 2장에서의 (그림 3)의 큐브 구조와 본 알고리즘을 적용하여 생성한 (그림 15)의 큐브 구조에는 큰 차가 있슴을 확인 할 수 있을 것이다. 본 큐브 구조에는 제품집합에 대한 분석을 수행할 수 있는 Item_Set 차원이 추가되어 있고, 사실 테이블에는 Item_Set_ID 열과 Trans_Count 열이 추가되어 있는 다른 점을 발견할 수 있을 것이다.

본 알고리즘을 적용하여 SQL 2005 Analysis Service를 실행시켜 보면 (그림 16)과 같은 큐브 데이터를 얻게 된다.

(그림 16)의 큐브 데이터는 기존 방법으로 생성한 (그림 7)의 큐브 데이터에 비하여 분석적인 측면에서 많은 차이를 보여주고 있다. 본 알고리즘을 적용한 큐브 데이터는 (그림 1)의 구매내역서에서 입력되어 있는 5개의 구매 내역이 사실 그대로 5개의 트랜잭션으로 즉, 5개의 구매내역으로 표현된다. (그림 7)에서는 잎구성원간의 연관성을 알 수 없었기 때문에 고객이 동시에 구매한 제품에 대한 정보를 알 수 없었으나, (그림 16)에서는 1번 고객이 1월 1일에 제품 a와 b를 동시에 구매한 내역이 올바로 나타난 것을 확인 할 수 있다.

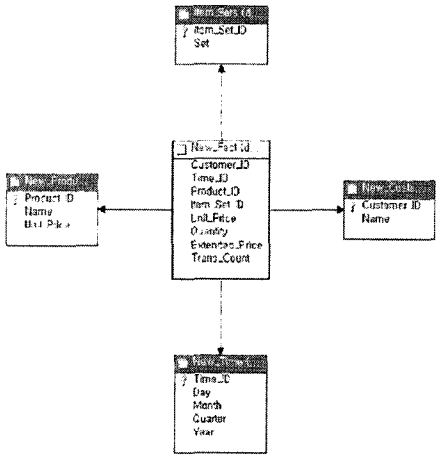


그림 15. 새로운 OLAP 큐브 구조
Figure 15. A new OLAP cube structures

Set	Extended Price	Trans Count
ab	410	1
b	600	2
c	200	1
ce	220	1
총합계	1430	5

그림 16. 새로운 알고리즘을 적용한 큐브 데이터
Figure 16. Generated cube data by applying the new algorithm

제품 b에 대해서 예를 들어보자. 1번 고객이 1월 1일에 제품 b를 1개를 구매였고, 3번 고객이 1월 3일에 제품 b를 2개, 그리고 역시 3번 고객이 1월 4일에 제품 b를 2개 구매한 것으로 되어 있다. 그러나 이들 구매내역에 대해 MS 연관규칙을 사용하여 장바구니분석을 수행해 보면 (그림 4)와 (그림 6)의 왼쪽 두 개의 그림에서와 같은 결과를 얻게 된다. 실제로 b를 3번 구매 하였으나, (그림 6)에서는 b가 2로 나타난 것을 확인할 수 있을 것이다(2.2 참조). 이러한 문제 역시 본 알고리즘을 적용하면 올바른 값을 가져오는 것을 확인할 수 있을 것이다. 즉, 제품 b가 ab의 제품집합으로서 1개, 제품 b만 구매한 횟수 두 번으로 올바르게 표현된다.

또한 1번 고객이 1월 1일에 제품 a와 b를 동시에 구매하였고, 1월 2일 제품 c를 구매한 것으로 되어 있는데, 분

석기간을 고려하지 못하는 MS 연관규칙을 사용하여 구매 제품 집합을 생성해 보면, a, b, c, ab, ac, bc, abc의 7개 제품집합이 생기게 된다. 실제로는 ab와 c의 2개 제품 집합 밖에 없었으나 (그림 4)에서와 같이 허구의 5개 제품 집합(a, b, ac, bc, abc)이 생기게 되는 것이다. 그러나 본 연구에서 제시하는 알고리즘을 적용하면 이러한 허구의 제품집합 생성을 방지할 수 있다.

OLAP과 데이터마이닝에서는 추적분석이 아주 중요한 부분을 차지하고 있다. 그러나 현재 관계형데이터베이스를 기반으로 하는 OLAP 큐브를 마이닝 하고자 할 때에는 3장에서 언급한바 같이 추적분석이 불가능하게 된다. 그러나 본 알고리즘 사용시에는 고객별, 시간별 분석이 가능하기 때문에 기존 방법에서의 추적분석에 관한 문제 역시 해결 할 수 있게 된다.

또 하나의 특징은 각 제품집합의 판매금액에 대한 분석 결과도 얻을 수 있다는 점이다. 1번 고객이 1월 1일 구매한 제품집합 ab의 구매금액이 410이라는 값을 확인할 수 있는데, 이 410이라는 값은 제품 a의 단가 130에 구매수량 2개를 곱한 값에 제품 b의 단가 150에 구매수량 1을 곱한 값을 합한 값이 된다. 즉, (130×2)+(150×1)=410 이라는 결과 값을 산출하게 되는 것이다. 장바구니분석에서는 판매금액에 대한 점은 전혀 고려하지 않고 있는데 비해서 본 알고리즘을 사용하면 현실적으로 중요한 인자가 되는 금액까지도 살펴볼 수 있다는 이점을 얻게 된다.

(그림 14-(e))의 사실테이블은 (그림 9)의 릴레이션이 되기 위한 모든 조건을 만족시킴을 알 수 있을 것이다. (그림 2)의 사실 테이블은 장바구니분석을 할 경우 (그림 9)의 모든 조건을 다 만족시키지 못하는데 비해서, (그림 14-(e))는 이를 다 만족시키기 때문에 전술한 모든 문제점을 해결할 수 있게 된다고 본다.

마이크로소프트 연관알고리즘을 사용하는 장바구니 분석은 두 단계로 크게 양분된다. 첫 번째 단계에서는 빈번하게 발생하는 제품집합을 발견하고, 두 번째 단계에서는 이러한 제품집합을 기반으로 하여 연관규칙을 생성하고 규칙의 중요도를 계산한다. 여기서 빈번하게 발생하는 제품집합을 찾는 첫 번째 단계가 장바구니 분석에 있어서의 핵심 부분이 된다. (그림 4)에서와 같은 제품집합(항목집합)과 해당 제품집합 발생빈도(지원)를 찾아내는 것이 첫 번째 단계에서 할 일이다. (그림 5)에서와 같은 연관규칙을 생성하고, 그 중요도를 계산하는 두 번째 단계는 오로지 첫 번째 단계의 결과에만 의존하는 기계적인 계산 작업만 수행하는 단계가 되기 때문에 첫 번째 단계가 장바구니분석의 핵심부분이라고 할 수 있는 것이다. 기존방식으로 핵심작업인 첫 번째 단계를 수행할 때 여러 가지 문제점을 야기했던 것을 본 알고리즘을 사용하면 첫 번째 단계의 작업을 OLAP 큐브 하나만으로 완벽하게 수행할 수 있게 되는 것이다.

V. 결론

오늘날 우리가 채택하고 있는 OLAP 큐브에는 셀 수 없을 만큼 많은 데이터가 들어 있으며, 또한 이 데이터에는 여러 가지 고객의 구매 패턴이 숨어들어 있다. 이 거대한 OLAP 큐브로부터 CRM적인 정보를 찾는 데에는 데이터마이닝 기술이 절대적으로 필요하게 된다. 현재의 OLAP 큐브는 관계형데이터베이스를 기반으로 하고 있다. 잎구성원 기반으로 설계되어 있는 현재의 OLAP 큐브구조에 마이크로소프트 연관 알고리즘을 적용할 경우, 구매 시기에 대한 기본키가 설정되지 않기 때문에 분석 기간 동안 고객이 여러 번 구매활동을 하더라도 한번밖에 구매활동을 하지 않았다고 간주하는 문제가 발생하며, 더 나아가서는 고객이 구매하지도 않은 제품집합을 만들게 된다. 이 때문에 기존 방법에서는 CRM에서 중요시 되는 추적분석이 불가능해진다.

본 연구의 알고리즘을 사용하면 관계형데이터베이스에서 요구하고 있는 릴레이션이 되기 위한 모든 조건을 만족시킴과 동시에 잎구성원 간의 연관성을 표현할 수 있게 되어 올바른 장바구니분석을 할 수 있게 된다. 이는 사실 테이블에서의 행간의 관련을 맺어줄 수 없었던 현 OLAP의 큐브구조를 행간의 관련을 맺어줄 수 있는 큐브구조로 변환시켰기 때문이다.

상호보완적인 관계에 있는 데이터마이닝과 OLAP이지만 올바른 장바구니분석을 이끌어 내는 데에는 OLAP측의 책임이 더 막중하다고 생각한다. 이는 기존 OLAP 큐브구조의 잘못된 설계로 인하여 장바구니분석이라는 데이터마이닝 기법이 올바로 적용되지 못하기 때문에서이다. 이런 의미에서 올바른 큐브구조를 생성하는 본 연구의 알고리즘이 의의가 있다고 할 수 있겠다.

본 연구에서 제시하는 방법으로 OLAP 큐브를 생성하면, 데이터마이닝의 중요 기술인 장바구니분석기법을 별도로 채택하지 않고서도 OLAP 큐브 자체만으로 장바구니분석의 핵심부분을 수행할 수 있게 된다. 이러한 점에서 본 연구에서 제시한 방법은 현대 사회에서처럼 고객에 대한

수많은 분석 작업이 이루어지는 환경에서 기업의 시간과 비용에 대한 부담을 보다 줄여 줄 수 있을 것이라고 생각하며 아울러 OLAP의 적용분야를 더욱 견실하게 만들 수 있을 것으로 예상한다.

참고문헌

[1] 권오주, 「OLAP Solutions +α SQL Server Analysis Services」, pp.40-54, 대림, 2001.

[2] 조재희, 박성진, 「OLAP 테크놀로지」, Sigma Insight, pp.292-295, 2003.

[3] 최인수 등, "DW에서의 질의어처리 성능향상을 위한 데이터 구조화 방법", 「한국컴퓨터 정보학회논문지」, 제10권, 제1호, pp.7-13, 2005.

[4] 유한주 등, "비유일 외래키 조합 기반의 사실테이블 모델링과 MDX 쿼리문 작성법", 「한국컴퓨터 정보학회논문지」,제12권, 제1호, pp.176-188, 2007.

[5] Berson, A., et al., Building Data Mining Applications for CRM, pp.89-92, McGraw Hill, 2000.

[6] Craig, R.S., et al., Microsoft Data Warehousing, pp.63-66, Wiley, 1999.

[7] Inman, W.H., Building the Data Warehouse, 2nd Edition, pp.127-128, Wiley, 1996.

[8] Kroenke, D.M., Database Processing, 10th Edition, pp.547-549, Prentice Hall, 2006.

[9] Spofford, G., et al., MDX Solutions, 2nd Edition, pp.295-305, Wiley, 2006.

[10] Tang, Z. and MacLennan, J., Data Mining with SQL Server 2005, pp.265-272, Wiley, 2005.

[11] Thomsen, E., OLAP Solutions, 2nd edition, pp.375-386, Wiley, 2002.

[12] Turban, E., et al., Introduction To Information Technology, 5th Edition, pp.424-427, Wiley, 2006.

[13] Chen, W.K., "Data structures for selective association mining", UMI Microform 3167946, 2005.

[14] Giuffrida, G., "Data mining of large relational databases", UMI Microform 3040197, 2002.

[15] Yang, Y. and Padmanabhan, B., "New data mining and marketing approaches for customer segmentation and promotion planning on the Internet", UMI Microform 3125924, 2004.

[16] Jensen, M.R., et al., "Discovering Multidimensional Structure in Relational Data", LNCS3181, pp.138-148, 2004.

[17] Kaya, M. and Alhajj, R., "Integrating Fuzziness with OLAP Association Rules Mining", LNAI2734, pp.353-368, 2003.

저 자 소 개

유 한 주

2001년 남서울대학교 산업공학과 졸업(학사)

2004년 숭실대학교 산업·정보시스템 공학과 졸업 (공학석사)

2005년 ~ 현재 숭실대학교 산업·정보시스템공학과 박사과정 재학 중

〈관심분야〉 MIS, DW, OLAP, MDX, CRM

최 인 수

1985년 서울대학교 산업공학과 졸업(공학박사)

1980년 ~ 현재 숭실대학교 산업·정보시스템공학과 교수

〈관심분야〉 MIS, DW, OLAP, MDX, CRM