데이터 모델링 기법을 이용한 EPCIS 시스템의 모델링에 관한 연구

이 종 석*
*남서울대학교 산업경영공학과

A Study on EPCIS System Modeling by Data Modeling Method

Zhong-Shi Li*
*Department of Industrial & Management Engineering, NAMSEOUL University

Abstract

Obtaining and applying information is considered as a critical task in the modern informationized society. Finding the one's necessary information and processing it into a detailed knowledge are becoming more priortized in the enormous amount of information.

Data modelling is the process that does not only reflect the demands of the user but the one that also facilitates the user's comprehension of the model itself. Ultimately, data modelling fully supports the processes that are requisite for the implementation of a data base and minimizes the alternations of the model during the development of applications.

Keywords: Data Modeling, EPCIS, EPCglobal Network.

1. 서 론

기업, 은행, 관공서에서는 방대한 양의 데이터를 처리하는 것이 주 업무가 되고 있다. 데이터의 효율적인 관리, 처리 및 유지가 기업 및 기관의 사활을 좌우하며 이러한 기업에서의 데이터 처리는 데이터베이스 시스템을 이용해서 이루어지고 있으며, 현재 상용화된 데이터베이스 관리 시스템(DBMS) 중에서 오라클, SQL 서버 등이 사용되어 지고 있다.[1]

관계형 데이터베이스 관리 시스템은 쓸데없이 낭비되는 공간 없이 효율적으로 데이터를 저장하기 위해서 최소한의 의미를 가지는 테이블로 나뉘어져야 하며 관계를 맺고 있는 두 테이블 중에 반드시 하나는 부모 테이블이 되고 나머지 하나는 자식 테이블이 되며 부모 테이블의 기본키(Primary Key, PK)는 자식 테이블의 포린키(Foreign Key, FK)로 전이되어진다.[11]

본 연구에서는 EPCglobal Network에서 발생하는 네 가지 정보 데이터를 파악하고, 데이터 모델링 기법을 이용하여 저장 용량과 속도를 모두 고려한 고성능 처리와 대용량 처리를 동시에 지원할 수 있게 하기 위한 방안을 제시한다.

2. 이론적 배경

2.1 EPCIS

EPCIS는 EPCglobal Network의 구성요소로서 EPCglobal Network의 EPC 데이터의 정보교환을 목표로 제품정보의 흐름을 제공하고, RFID Tag에 기록된 EPC 데이터, 입고일, 입고장소, 출고일 등의 정보를 제공한다[3].

† 이 논문은 2011년도 남서울대학교 연구지원비에 의해 연구되었음.
† 교신저자: 이종석, 남서울대학교 산업경영공학과
M・P : 010-2631-9780, E-mail: leezs0423@nsu.ac.kr
2012년 4월 9일 접수; 2012년 6월 7일 수정본 접수; 2012년 6월 7일 게재확정

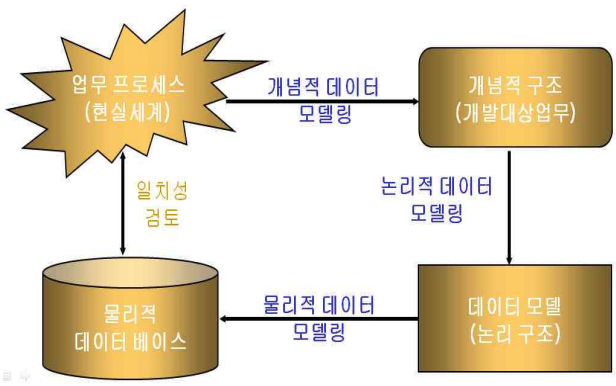
물리적으로 객체가 이동됨에 따라 서로 다른 이동 거점에 설치된 RFID 시스템으로부터 인식된 RFID 태그 정보는 기업의 비즈니스 로직에 따라 Event 데이터의 형태가 결정이 되는데, 크게 네 가지 정보 즉 Object Event Data, Aggregation Event Data, Quantity Event Data, Transaction Event Data로 구분된다. EPCIS 서버는 이상의 데이터를 기반으로 거래정보를 포함하여 EPCIS Repository에 저장·관리 한다.

2.2 데이터 모델링

데이터 모델링은 정보시스템을 구축하기 위해 해당 업무에 어떤 데이터가 존재하는지 또는 업무가 필요로 하는 정보는 무엇인지를 분석하는 방법이다. 이것을 좀 더 실무적으로 해석해 보면 업무에서 필요로 하는 데이터를 시스템 구축 방법론에 의해 분석하고 설계하여 정보시스템을 구축하는 과정으로 정의할 수 있다.[5]

데이터 모델링을 하는 주요한 이유는 업무 정보를 구성하는 기초가 되는 정보들을 일정한 표기법에 의해 표현함으로써 정보시스템 구축의 대상이 되는 업무 내용을 정확하게 분석하는 것이 첫 번째 목적이다. 두 번째는 분석된 모델을 가지고 실제 데이터베이스를 생성하여 개발 및 데이터 관리에 사용하기 위한 것이다. 즉, 데이터 모델링이라는 것은 단지 데이터베이스만을 구축하기 위한 용도로만 쓰이는 것이 아니라 데이터 모델링 자체로서 업무를 설명하고 분석하는 부분에도 매우 중요한 의미를 가지고 있다고 할 수 있다.

현실세계에서 데이터베이스까지 만들어지는 과정은 시간에 따라 진행되는 과정으로서 추상화 수준에 따라 개념적 데이터 모델, 논리적 데이터 모델, 물리적 데이터 모델로 정리할 수 있다.[11][17]



[그림 1] 데이터 모델링

2.3 개념적 데이터 모델링

개념적 데이터 모델링은 조직, 사용자의 데이터 요구사항을 찾고 분석하는데서 시작한다. 이 과정은 어떠한 자료가 중요하며 또 어떠한 자료가 유지되어야 하는지를 결정하는 것도 포함한다.

이 단계에 있어서의 주요한 활동은 핵심 엔티티와 그들 간의 관계를 발견하고, 그것을 표현하기 위해서 엔티티-관계 다이어그램(ERD)을 생성하는 것이다. ERD는 조직과 다양한 데이터베이스 사용자에게 어떠한 데이터가 중요한지 나타내기 위해서 사용된다.[17][18]

개념적 데이터 모델링 과정을 거치게 되면 관련된 엔티티들이 추출되고 이들 엔티티들의 속성과 엔티티들 간의 관계가 정의되며 최종적으로 ERD가 생성된다.

2.4 논리적 데이터 모델링

논리적 데이터 모델링은 데이터베이스 설계 프로세스의 Input으로써 비즈니스 정보의 논리적인 구조와 규칙을 명확하게 표현하는 기법 또는 과정이라 할 수 있다.

논리 데이터 모델링의 핵심은 어떻게 데이터에 액세스하고, 누가 데이터에 액세스하며, 그러한 액세스의 전산화와는 독립적으로 다시 말해서 누가, 어떻게 그리고 전산화와는 별개로 비즈니스 데이터에 존재하는 사실들을 인식하여 기록하는 것이다.[17][18]

데이터 모델링 과정에서 가장 핵심이 되는 부분이 논리 데이터 모델링이라고 할 수 있다. 이는 시스템 구축을 위해서 가장 먼저 시작할 기초적인 업무 조사를 하는 초기단계에서부터 인간이 결정해야 할 대부분의 사항을 모두 정의하는 시스템 설계의 전 과정을 지원하는 '과정의 도구'이다.

이 단계에서 수행하는 또 한 가지 중요한 활동은 정규화이다. 정규화는 논리 데이터 모델 상세화 과정의 대표적인 활동으로, 논리 데이터 모델의 일관성을 확보하고 중복을 제거하여 속성들이 가장 적절한 엔티티에 배치되도록 함으로써 보다 신뢰성 있는 데이터 구조를 얻는데 목적이 있다. 논리 데이터 모델의 상세화는 식별자 확정, 정규화, M:M 관계 해소, 참조 무결성 규칙 정의 등을 들 수 있으며, 추가적으로 이력 관리에 대한 전략을 정의하여 이를 논리 데이터 모델에 반영함으로써 데이터 모델링을 완료하게 된다.

따라서 논리적 데이터 모델링은 개념적 데이터 모델링에서 정의된 ERD를 맵핑 룰을 통해 관계형 데이터베이스 이론에 입각한 스키마를 생성하고 완벽한 정규화 과정을 수행하는 과정이다.

2.5 물리적 데이터 모델링

데이터베이스 설계 과정의 세 번째 단계인 물리적 데이터 모델링은 논리 데이터 모델이 데이터 저장소로서 어떻게 컴퓨터 하드웨어에 표현될 것인가를 다룬다. 데이터가 물리적으로 컴퓨터에 어떻게 저장될 것인가에 대한 정의를 물리적 스키마라고 한다. 이 단계에서 결정되는 것은 테이블, 컬럼 등으로 표현되는 물리적인 저장 구조와 사용될 저장 장치, 자료를 추출하기 위해 사용될 접근 방법 등이 있다.[17][18]

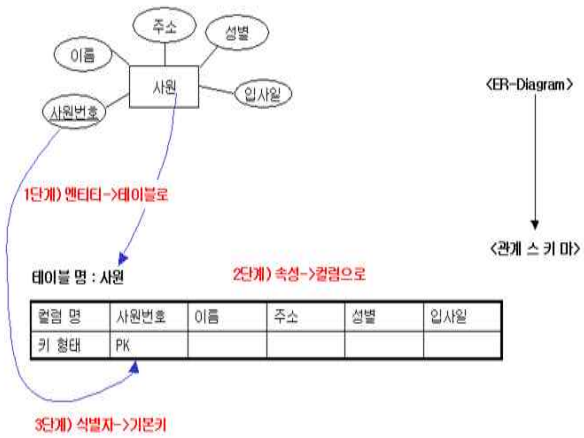
즉 물리적 데이터 모델링에서는 사용되어질 DBMS를 선정하고 데이터 타입과 사이즈를 정의하며 데이터 사용량 분석과 업무 프로세스 분석을 통하여 역정규화, 제약조건, 오브젝트 등을 정의하고 최종 데이터베이스를 생성하게 된다.

실질적인 현실 프로젝트에서는 개념적 데이터 모델링 => 논리적 데이터 모델링 => 물리적 데이터 모델링으로 수행하는 경우는 드물며 개념적 데이터 모델링과 논리적 데이터 모델링을 한꺼번에 수행하여 논리적인 데이터 모델링으로 수행하는 경우가 대부분이다.

3. 데이터 모델링 기법

3.1 매핑 룰(Mapping Rule)

매핑 룰이란 개념적 데이터베이스 모델링에서 얻어진 ERD를 관계형 데이터베이스 이론에 입각하여 데이터베이스 스키마로 변환하는 과정이다. 즉 개념적 모델링 단계에서 지정한 단순 엔티티는 테이블로, 속성은 컬럼으로, 식별자는 기본키로, 관계는 포린키로 변환시키는 과정이다.



[그림 2] 매핑 룰

3.2 정규화(Normalization)

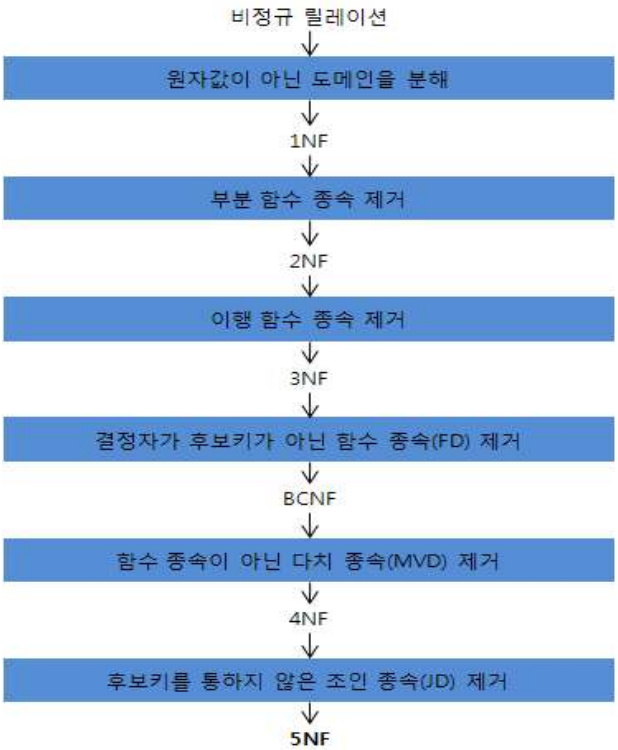
데이터베이스를 구축할 때 선행되어야 할 작업이 데이터 설계이며, 여기에서 중복된 데이터가 삽입되는 것을 방지하는 작업이 필요하다. 따라서 이러한 데이터의 중복성 문제를 해결하기 위해 테이블을 분해하는 과정이 정규화이다.[1]

정규화가 모든 중복을 완전하게 제거하는 것은 불가능하다. 하지만 최소화시킬 수 있을 만큼 최소화해야 하는 것이 데이터 중복이다. 이상 현상이 발생하지 않을수록 데이터 무결성은 높아지며 데이터 품질은 높아진다. 또한 데이터와 데이터베이스의 안정성과 신뢰성도 높아진다.[4]

관계형 데이터 모델이 되려면 속성의 종속성과 의존성을 분석해 더는 분해될 수 없는 엔티티로 만드는 정규화 과정을 반드시 거쳐야 한다.[4]

정규화를 수행하면 결과적으로 데이터 무결성은 높아진다. 또한 정규형을 사용하면 데이터 저장 공간의 사용을 최소화할 수 있으며 데이터 모델을 단순화할 수 있다. 일반적으로 데이터의 저장 공간이 최소화되면 성능에 도움을 준다. 그리고 데이터 모델이 단순해지면 모델을 관리하기 수월해진다.[4]

현재 정규화 이론은 6단계까지 있다. 그러나 일반적으로 3단계까지만 사용한다.[11]



[그림 3] 정규화

3.3 역정규화(Denormalization)

정규화된 스키마는 데이터를 입력, 수정, 삭제할 때 관계를 맺고 있는 테이블을 참조해야 하며 가장 작은 단위로 테이블에 나뉘어져 있기 때문에 연관된 정보를 보기 위해서 조인을 수행해야 한다. 그러므로 정규화된 스키마는 시스템의 부하를 유발하게 된다. 역정규화는 시스템의 성능 향상을 위해서 정규화에 위배되는 행위를 하는 것을 말한다. 즉 성능 향상을 위해 정규화를 무시하는 것이다.

역정규화를 수행하기 위해서는 우선 정확한 업무 분석과 사용자들의 업무 프로세스를 분석해야만 한다. 역정규화는 데이터 사용량이 많은 테이블을 기준으로 해서 우선적으로 고려되어야 한다.

역정규화 유형에는 데이터 중복, 파생 컬럼의 생성, 테이블 분리, 요약 테이블 생성, 관계 제거, 테이블 통합 등이 있다.

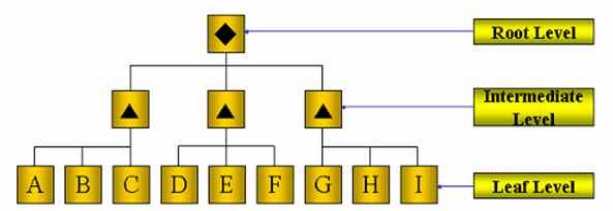
3.4 인덱스(Index)

데이터베이스를 사용하는 이유는 방대한 데이터를 효율적으로 저장하여 사용자가 원하는 데이터를 신속, 정확하게 찾기 위함이다.[1] 이때 필수적으로 사용되어지는 것이 인덱스이다.

인덱스는 데이터베이스 내의 테이블에서 원하는 데이터를 좀 더 빨리 찾아줄 수 있게끔 데이터의 위치 정보를 모아 놓은 데이터베이스 내의 개체이다. 인덱스는 항상 정렬되어 있는 상태로 구성되며 이러한 정렬된 내용에 데이터의 위치 정보가 포함되어 있음으로 해서 보다 빠른 데이터의 검색이 가능하게 된다.

3.4.1 인덱스 구조와 유형

인덱스는 계층적인 구조로 정의되어 있고 상위 레벨의 페이지들은 하위 레벨의 페이지들에 대한 정보를 가지고 있기 때문에 찾고자 하는 데이터를 인덱스의 루트부터 시작해서 단지 몇 단계만 거치면 원하는 항목을 찾을 수 있으므로 효율적인 검색이 가능해 진다.



[그림 4] 인덱스의 구조

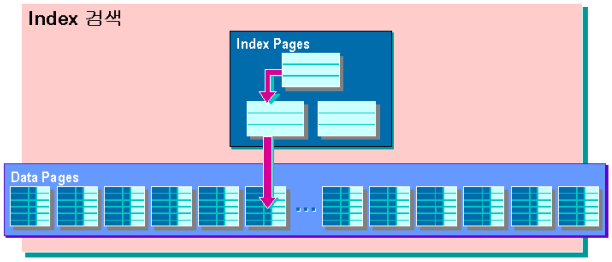
인덱스에는 클러스터드 인덱스와 넌 클러스터드 인덱스 두 가지 유형이 있다.

인덱스를 만들기 원하는 컬럼에 클러스터드 인덱스를 만들게 되면 기본적으로 그 열을 기준으로 물리적으로 데이터를 정렬시킨다. 클러스터드 인덱스의 리프 레벨은 바로 데이터 페이지가 된다.

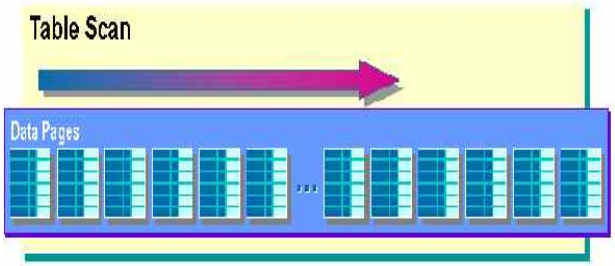
인덱스를 만들기 원하는 컬럼에 넌 클러스터드 인덱스를 만들게 되면 물리적으로 데이터 페이지를 정렬시키지 않고 데이터 페이지들에 있는 데이터들의 위치 정보를 인덱스로 구성하게 된다. 따라서 데이터 페이지 위에 인덱스의 리프 레벨이 만들어지게 되며 넌 클러스터드 인덱스의 리프 레벨은 정렬된 상태로 관리되며 하위 데이터 페이지에 대한 포인터 정보를 갖는다.

3.4.2 데이터 검색 방법과 검색 유형

데이터베이스에서 데이터를 검색하는 경우 데이터베이스 관리 시스템은 크게 두 가지 방법으로 데이터를 검색하는데 하나는 인덱스를 이용한 검색으로 이를 Index Seek라 하며 다른 하나는 인덱스가 없거나 인덱스가 없는 컬럼을 기준으로 데이터를 검색하는 경우 데이터를 검색하기 위해 인덱스를 이용하지 않고 데이터가 저장되는 기본 단위인 Page를 모두 뒤지는 방법으로 이를 Table Scan이라 한다.



[그림 5] Index Seek



[그림 6] Table Scan

데이터를 검색하는 유형에는 다음과 같은 것들이 있다. 단일한 데이터 즉 검색한 결과가 하나 혹은 없는 경우의 검색 유형을 포인트 쿼리라 하며 검색한 결과가 여러 개의 데이터 즉 많거나 하나이거나 혹은 조건

에 맞는 데이터가 없는 경우 아무런 결과가 없을 수 있는 질의의 유형을 범위 조회라고 한다. 그리고 검색 유형이 포인트 쿼리일 수도 있고 혹은 범위 조회일 수도 있지만 조회의 조건과 조회의 대상이 되는 컬럼이 모두 인덱스로 구성된 경우의 질의를 커버드 쿼리라고 한다.

3.5 저장 프로시저(Stored Procedure)

저장 프로시저는 SQL 문을 서버에서 미리 컴파일해서 저장해 놓은 것을 말한다. 일반적으로 우리가 쿼리 분석기에서 특정 테이블에 대한 검색, 삭제, 갱신 등의 작업을 하는 문을 작성하게 되면 이 문장은 서버에 전달되어 최적화 과정을 거쳐서 데이터를 가져오는 아주 복잡한 과정을 거치게 된다. 저장 프로시저는 빈번히 발생하는 SQL 문이나 필요한 SQL 문을 미리 서버에서 컴파일 해 두었다가 클라이언트에서 필요할 때 저장 프로시저의 이름을 호출하면 결과를 보여주는 형태이다.[1]

3.6 뷰(View)

데이터베이스를 설계할 때 정규화 과정을 거쳐서 테이블 내에 종속성을 제거하여 테이블을 분해하여 여러 테이블로 만드는 것이 일반적이다. 그런데 실제 업무에서는 JOIN 등의 방법으로 여러 테이블에 흩어져 있는 정보를 묶어서 가져오는 경우가 많다. 이런 경우 매번 SQL 문장을 작성하여 필요한 정보를 가져오는 것보다 실제 데이터가 들어있는 테이블 간에 자주 질의되는 것을 모아둔 가상의 테이블인 뷰를 사용한다.[1]

뷰는 실제 데이터 값이 저장되지는 않지만, 실제 테이블에 대한 창이므로 뷰를 통해 원하는 작업(SELECT, DELETE, UPDATE 등)을 할 수 있다.[5]

3.7 트리거(Trigger)

트리거는 자동으로 실행되는 프로시저의 한 형태로서 테이블에 데이터가 입력, 수정, 삭제되어질 때 다른 테이블에 연관된 작업을 정의하기 위한 목적으로 사용되어지며 데이터 무결성 등을 확인하는 용도로도 사용되어 질 수 있다.

트리거는 테이블이나 뷰를 통해 데이터가 입력, 수정, 삭제될 경우 자동으로 실행되어지기 때문에 연관된 작업을 처리하는데 있어서 여러 번 프로시저를 호출해서 실행하거나 여러 번 SQL 명령을 실행할 필요가 없기 때문에 사용하는 입장에서의 복잡성을 줄일 수 있다.

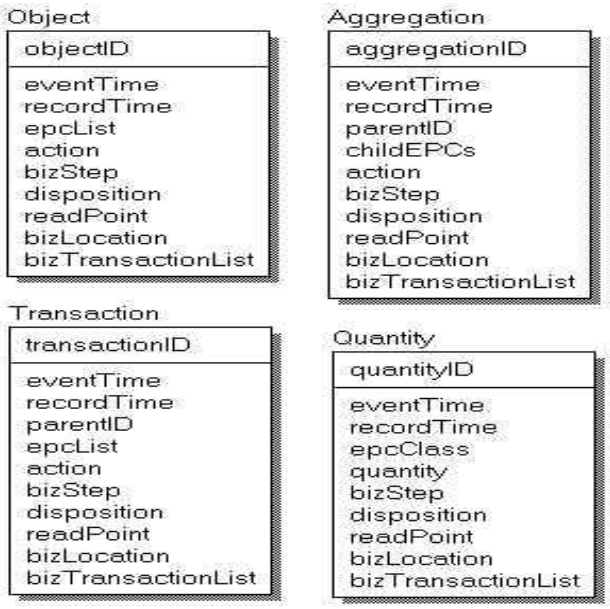
따라서 프로젝트 수행 시 개발자나 프로그래머들이 복잡한 업무를 숙지하지 않아도 되기 때문에 프로젝트를 안정적으로 수행할 수 있는 장점이 있다.

3.8 커서(Cursor)

커서는 SELECT 문장의 결과 집합이다. 커서를 사용하는 이유는 한번 커서를 정의해서 만들면 추가로 SELECT할 필요 없이 바로 커서를 이용하여 각각의 레코드에 대한 데이터 처리 작업이 가능하기 때문이다.

4. 데이터 모델링

EPCglobal Network에서 발생하는 데이터들은 크게 네 가지 정보 즉 Object Event Data, Aggregation Event Data, Quantity Event Data, Transaction Event Data로 구분된다.[12][14]



[그림 7] Event Data

다음은 Event Data 중 Object Event 테이블에 저장되는 가상의 데이터이다.

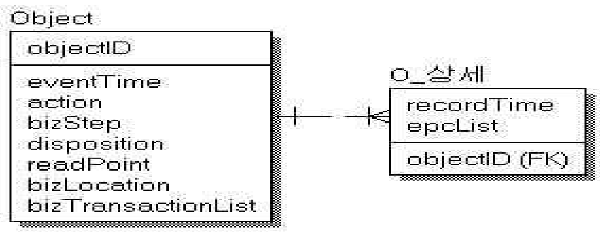
이처럼 EPCIS 규격에 따라 정의된 각 EPCIS Event 데이터 구조를 기반으로 하여 데이터를 저장하면 많은 중복이 발생하며 이로 인하여 성능에 큰 영향을 미친다. 따라서 이러한 데이터의 중복성 문제를 해결하기 위해 테이블을 분해하는 과정 즉 정규화 과정이 필요하다.

num	dsEventTime	dsRecordTime	dsEPCList	dsAction	dsBizStep
1	2012-04-04 13:48:18	2012-04-04 13:48:18	urn:epc:id:sgtin:0037001,030241,0000001	OBSERVE	urn:epcglobal:epcis:bizstep:fmcg:shipped
2	2012-04-04 13:48:18	2012-04-04 13:48:18	urn:epc:id:sgtin:0037001,030241,0000002	OBSERVE	urn:epcglobal:epcis:bizstep:fmcg:shipped
3	2012-04-04 13:48:18	2012-04-04 13:48:18	urn:epc:id:sgtin:0037001,030241,0000003	OBSERVE	urn:epcglobal:epcis:bizstep:fmcg:shipped
4	2012-04-04 13:48:18	2012-04-04 13:48:18	urn:epc:id:sgtin:0037001,030241,0000004	OBSERVE	urn:epcglobal:epcis:bizstep:fmcg:shipped
5	2012-04-04 13:48:18	2012-04-04 13:48:18	urn:epc:id:sgtin:0037001,030241,0000005	OBSERVE	urn:epcglobal:epcis:bizstep:fmcg:shipped
6	2012-04-04 13:48:18	2012-04-04 13:48:18	urn:epc:id:sgtin:0037001,030241,0000006	OBSERVE	urn:epcglobal:epcis:bizstep:fmcg:shipped
7	2012-04-04 13:48:18	2012-04-04 13:48:18	urn:epc:id:sgtin:0037001,030241,0000007	OBSERVE	urn:epcglobal:epcis:bizstep:fmcg:shipped
8	2012-04-04 13:48:18	2012-04-04 13:48:18	urn:epc:id:sgtin:0037001,030241,0000008	OBSERVE	urn:epcglobal:epcis:bizstep:fmcg:shipped
9	2012-04-04 13:48:18	2012-04-04 13:48:18	urn:epc:id:sgtin:0037001,030241,0000009	OBSERVE	urn:epcglobal:epcis:bizstep:fmcg:shipped
10	2012-04-04 13:48:18	2012-04-04 13:48:18	urn:epc:id:sgtin:0037001,030241,0000010	OBSERVE	urn:epcglobal:epcis:bizstep:fmcg:shipped

dsDisPosition	dsReadPoint	dsBizLocation	dsBizTransactionList
urn:epcglobal:epcis:disp:fmcg:unknown	urn:epc:id:sgln:0037001,073418,33554431	urn:epc:id:sgln:0037001,073417,33554431	http://transaction,inha,ac,kr/po/1
urn:epcglobal:epcis:disp:fmcg:unknown	urn:epc:id:sgln:0037001,073418,33554431	urn:epc:id:sgln:0037001,073417,33554431	http://transaction,inha,ac,kr/po/1
urn:epcglobal:epcis:disp:fmcg:unknown	urn:epc:id:sgln:0037001,073418,33554431	urn:epc:id:sgln:0037001,073417,33554431	http://transaction,inha,ac,kr/po/1
urn:epcglobal:epcis:disp:fmcg:unknown	urn:epc:id:sgln:0037001,073418,33554431	urn:epc:id:sgln:0037001,073417,33554431	http://transaction,inha,ac,kr/po/1
urn:epcglobal:epcis:disp:fmcg:unknown	urn:epc:id:sgln:0037001,073418,33554431	urn:epc:id:sgln:0037001,073417,33554431	http://transaction,inha,ac,kr/po/1
urn:epcglobal:epcis:disp:fmcg:unknown	urn:epc:id:sgln:0037001,073418,33554431	urn:epc:id:sgln:0037001,073417,33554431	http://transaction,inha,ac,kr/po/1
urn:epcglobal:epcis:disp:fmcg:unknown	urn:epc:id:sgln:0037001,073418,33554431	urn:epc:id:sgln:0037001,073417,33554431	http://transaction,inha,ac,kr/po/1
urn:epcglobal:epcis:disp:fmcg:unknown	urn:epc:id:sgln:0037001,073418,33554431	urn:epc:id:sgln:0037001,073417,33554431	http://transaction,inha,ac,kr/po/1
urn:epcglobal:epcis:disp:fmcg:unknown	urn:epc:id:sgln:0037001,073418,33554431	urn:epc:id:sgln:0037001,073417,33554431	http://transaction,inha,ac,kr/po/1
urn:epcglobal:epcis:disp:fmcg:unknown	urn:epc:id:sgln:0037001,073418,33554431	urn:epc:id:sgln:0037001,073417,33554431	http://transaction,inha,ac,kr/po/1

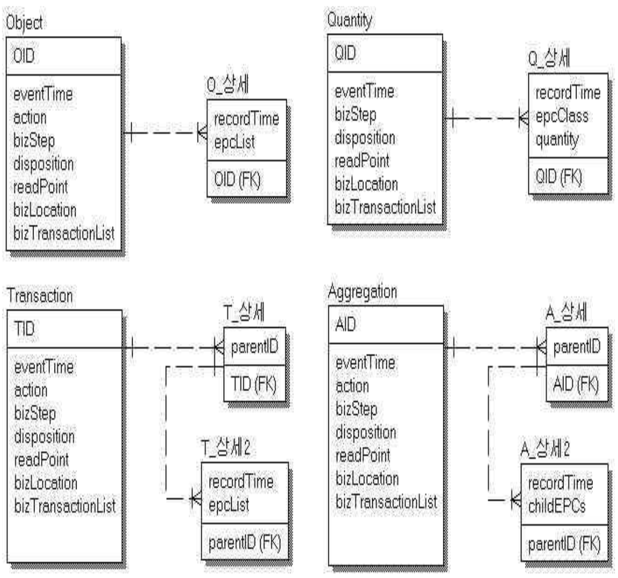
[그림 8] Object Event Data

다음은 Object Event 테이블에 대하여 정규화를 한 후의 테이블 구조이다.



[그림 9] Object Event 테이블 정규화

마찬가지 방법으로 나머지 이벤트 테이블에 대해서도 정규화 과정을 거치면 다음과 같다.



[그림 11] Event 테이블 정규화

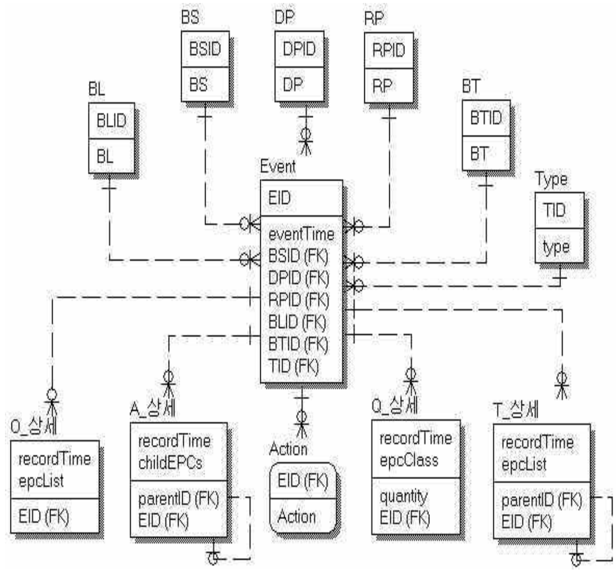
정규화 과정을 마치면 비슷한 성격의 테이블이 생긴 것을 확인할 수 있다. 또한 이벤트 테이블들이 서로 관계가 형성되지 않은 상태다.

따라서 역정규화 과정을 거쳐서 테이블을 통합하여 새로운 테이블을 생성하여 각 이벤트 테이블을 연결시키고, 각 이벤트 별로 구분하기 위하여 파생 컬럼을 추가한다.

또한 이벤트 테이블의 인스턴스의 길이가 너무 길어질 것이 예상되므로 bizStep, disposition, readPoint, bizLocation, bizTransactionList 컬럼에 대해서 각기 부모 테이블을 만들고 부모 테이블의 기본키로 관계를 맺는다.

Aggregation 이벤트와 Transaction 이벤트의 자식 테이블들은 하나의 테이블에서 스스로 관계를 가질 수 있게 하기 위하여 재귀적 관계로 구성한다.

Event 테이블에서 특정 필드에 대한 조회가 많이 발생하면 데이터 중복을 고려할 수도 있다.



[그림 11] Event 테이블 역정규화

EPCglobal Network에서 발생하는 데이터는 대용량이며 이는 빈번한 데이터 삽입 프로세스를 유발한다. 따라서 빈번히 발생하는 SQL 문이나 필요한 SQL 문을 미리 서버에서 컴파일 해 두었다가 클라이언트에서 필요할 때 호출해서 사용하는 저장 프로시저를 만들어 두는 것이 좋다.

또한 Event 테이블의 삽입에 따른 추가 삽입 작업이 많기 때문에 트리거를 이용하면 SQL 명령을 실행하는 횟수를 줄일 수 있다.

데이터를 조회할 때 Table Scan을 가급적 피해야 한다. 따라서 인덱스를 적절히 사용해야 한다. 넌 클러스

터드 인덱스는 클러스터드 인덱스보다 한 단계를 더 내려가야만 원하는 데이터를 찾을 수 있다. 업무의 성격상 데이터를 조회할 때 대부분의 경우가 포인트 쿼리이다. 즉 특정 epcList에 대한 이벤트별 정보가 조회의 대상이다. 따라서 각 테이블의 기본키에는 넌 클러스터드 인덱스를 우선적으로 적용하도록 한다.

물리적인 테이블에 FK를 사용하지 않아도 데이터 모델 관계에 의해 상속받은 FK 속성들은 SQL WHERE 절에서 조인으로 이용되는 경우가 많이 있다. FK 인덱스를 적절하게 설계하여 구축하지 않더라도 초기에는 데이터양이 얼마 되지 않기 때문에 성능 저하가 나타나지 않는다. 그러나 데이터양이 누적되면 SQL 성능이 나빠져 데이터베이스 서버에 심각한 장애를 일으킨다. 그러므로 물리적인 테이블에 FK 제약을 걸었을 때는 반드시 FK 인덱스를 생성하도록 해야 한다.

5. 결 론

데이터를 정확히 관리하고 사용자에게 빠르게 제공하기 위해서 모델링은 필요하다. 데이터가 잘 관리될 수 있도록 지원해 주는 것이 데이터 모델링이며 오늘날 기업은 데이터를 효율적으로 관리하는 것에 성공이 달렸다.[4]

본 연구에서는 EPCglobal Network에서 발생하는 네 가지 정보 데이터를 파악하여 데이터 모델링 기법을 이용하여 분리하고 재설계하였다. 저장 용량과 속도를 모두 고려하여 고성능 처리와 대용량 처리를 동시에 지원할 수 있게 하기 위하여 업무 분석과 사용자들의 업무 프로세스를 분석하여 역정규화 과정을 수행하였다.

추후 연구과제로는 기존 시스템과의 비교 분석 및 다양한 물리적 오브젝트를 적용했을 때의 효과 분석 그리고 현업 프로세스를 기반으로 하는 시뮬레이션에 대한 연구가 필요할 것이다.

6. 참 고 문 헌

[1] 그림으로 배우는 MS-SQL 2000 서버, 홍릉과학출판사, 박정용.
[2] 김진승, “스트림 데이터 관리자를 이용한 EPCIS에서의 효율적인 대용량 데이터 처리”, 경희대학교, 석사학위논문, 2007.
[3] 김현주, “대용량 데이터베이스의 스키마구조 분석”, 동국대학교 석사학위논문, 2003.
[4] 관계형 데이터 모델링, 오픈메이드, 김기창.
[5] 데이터베이스 설계와 구축: 성능까지 고려한 데이터 모델링, 한빛미디어, 이춘식.
[6] 데이터베이스 튜닝, 브레인코리아, 최용락.
[7] 실무 사례로 다지는 고성능 데이터베이스 튜닝, 비팬북스, 권순용.
[8] 실행 계획으로 배우는 고성능 데이터베이스 튜닝, 비팬북스, 권순용.
[9] 새로 쓴 대용량 데이터베이스 솔루션 I, ㈜엔코아컨설팅, 이화식.
[10] 아는 만큼 보이는 데이터베이스 설계와 구축, 한빛미디어, 이춘식.
[11] 알기 쉽게 해설한 데이터베이스 모델링, 프리렉, 김연홍.
[12] 이종석, 이창호, “시뮬레이션을 이용한 EPCIS의 효율화 방안에 관한 연구”, 대한안전경영과학회지, 제12권 제4호, 2010.
[13] 이창호, “EPC Network의 효율적 운영을 위한 EPCIS Repository와 Hybrid DBMS 최적화 연구”, 인하대학교 산학협력단 연구보고서, 2010.
[14] 이창호, 조용철, “EPCIS Event 데이터 크기의 정량적 모델링에 관한 연구”, 대한안전경영과학회지, 제11권 제4호, 2009.
[15] 정동규, 박재관, 황봉희, “EPCglobal Network에서 분산 연속질의 처리”, 한국정보과학회 2008 가을 학술발표논문집 제35권 제2호(C), 2008.
[16] 조용철, “RFID기반의 통합물류센터를 위한 효율적인 EPCIS Repository 구축에 관한 연구”, 인하대학교 박사학위논문, 2009.
[17] SQL 전문가 가이드, 한국데이터베이스진흥원.
[18] http://www.dbguide.net
[19] http://www.infomaster.co.kr
[20] http://itmore.tistory.com

저 자 소 개

이 종 석

인하대학교 산업공학과 공학박사 취득. 현재 남서울대학교 산업경영공학과 교수로 재직 중.
관심분야: EPCglobal Network, 시뮬레이션, RFID를 활용한 응용시스템, SCM, DB 등.

주 소 : 충남 천안시 성환읍 매주리 21 남서울대학교 산업경영공학과