Geo-DBMS의 3차원 Primitive를 이용한 공간정보데이터 구축 및 활용
- CityGML을 기반으로-
(Modeling Spatial Data in a geo-DBMS using 3D Primitives)

박 인 혜* 이 지 영**
(In Hye Park) (Jiyeong Lee)

요 약 최근 3차원 실내·외 공간정보데이터 모델에 대한 많은 연구가 진행되고 있다. 이러한 데이터 모델을 기반으로 구축된 3차원 공간데이터는 양이 방대하고 비교적 복잡한 구조를 갖는다. 따라서 이를 효과적으로 저장 및 관리, 응용하기 위해서는 DBMS를 활용하는 것이 유리하다. 이러한 필요에 의해 Geo-DBMS에서 데이터를 저장하고 응용하는 연구가 많이 이루어지고 있는데 Oosterom, Arens 등이 3차원 건물, 지표의 Geometry와 Topology를 DBMS에 저장하는 방법을 연구하였다. 본 논문은 GML3 기반의 3차원 도시 모델의 저장 및 교환을 위한 포맷인 CityGML 1.0을 따르는 구조로 데이터를 데이터베이스에 저장하였으며, 상용 DBMS인 Oracle Spatial 11g를 사용하였다.

키워드 : 3차원 공간정보, 데이터 모델, Geo-DBMS, CityGML

Abstract Recently, many researches have been conducted to develop 3D Indoor/Outdoor Spatial Data Models. The 3D data created based on these data models have complex data structures. In order to manage these data efficiently, it is better to use a DBMS. There have been many researches to maintain the 3D data in Geo-DBMS, such that Oosterom (2002) and Arens (2005) developed a method to store 3D Building model, geometric and topological data of coverage in DBMSa. In this study, we propose a method to store the CityGML data into the RDBMS, Oracle Spatial 11g.

Keywords: 3D Spatial Data, Data Model, Geo-DBMS, CityGML

1. 서 론

최근 3차원 실내외 공간정보데이터 모델에 대한 많은 연구가 진행되고 있다. 대부분의 데이터 모델은 스키마를 정의하기 위해 데이터 교환의 표준인 XML을 사용한다. 이에 따라 OGC은 지리정보의 저장, 전송 및 교환의 목적으로 XML기반의 표준인 GML을 제안하였고 현재 GML 3.2가 제안되어 사용되고 있다. 국내외 지리정보 및 공간정보를 표현하기 위한 데이터 모델은 CityGML, 3DF-GML 등 대부분이 GML 응용스키마의 형태로 개발되고 있는 추세이다. 이러한 데이터 모델을 기반으로 구축된 3차원 공간데이터는 양이 방대하고 비교적 복잡한 구조를 갖는다. 따라서 이를 효과적으로 저장 및 관리, 응용하기 위해서는 DBMS를 활용하는 것이 유리하다. 이러한 필요에 의해 Geo-DBMS에서 데이터를 저장하고 응용하는 연구가 많이 이루어지고 있으며 Oosterom (2002), Arens(2005)등이 3차원 건물, 지표(coverage)의 Geometry와 Topology를 DBMS에 저장하는 방법을 연구하였는데 현재 GML응용스키마를 데이터베이스에 저장하는 방법에 대한 연구는 거의 없다.

따라서 본 연구에서는 GML3(Geography Markup Language version 3) 기반의 3차원 도시 모델의 저장 및 교환을 위한 포맷인 CityGML 1.0을 따르는 구조로 데이터를 데이터베이스에 저장하여 보았다. 여기서는 상용 DBMS인 Oracle Spatial 11g를 사용하였다.

2. XML 문서 저장 방법

일반적으로 관계데이터베이스에 XML기반 문서를 저장하는 방법은 두 가지의 접근방법으로 나뉠 수 있다. 하나는 모델사상접근(Model Mapping Approach)이고 다른 하나는 구조 사상 접근(Structure Mapping Approach)이다. 모델 사상 접근 방법은 정해진 데이터베이스 스키마를 이용하여 별도의 XML 스키마 없이 XML 문서를 저장한다. 즉, 모델 사상 접근 방법은 XML문서를 데이터 모델로 변환하는 과정에서 XML의 구조적 특징과 내용을 문서 독립적으로 모델화하여 저장하지 않기 때문에 XML 문서의 구조를 갱신할 때 이에 따른 데이터베이스 구조 정보가 수정되지 않아도 된다는 장점이 있다. 모델 사상 접근 방법 에서는 간선(edge)기반과 정점(node)기반 접근방법으로 나눌 수 있다. 간선 기반 접근 방법은 기본적으로 XML 문서를 표현한 그래프 상에서 모든 간선들에 대한 정보를 하나의 테이블에 저장한다. 한편, 정점(Node)기반 접근 방법은 XML문서를 표현한 그래프 상에서의 모든 노드들에 대한 정보를 가지고, 경로, 노드,

---

†이 논문은 2009 GIS 공동추계학술대회에서 'Geo-DBMS의 3차원 Primitive를 이용한 공간정보데이터 구축 및 활용'의 제목으로 발표된 논문을 확장한 것임

†본 연구는 국토해양부 첨단도시기술개발사업 - 지능형국토공간정보기술혁신 사업과제의 연구비지원(07국토정보C04)에 의해 수행되었습니다.

* 서울시립대학교 공간정보공학과 박사과정, ihpsm@uos.ac.kr
** 서울시립대학교 공간정보공학과 교수, jlee@uos.ac.kr(교신저자)

논문접수 : 2009.08.07
수 정 일 : 2009.09.10
심사완료 : 2009.09.25

값 등의 정보들을 따로 분할하여 테이블로 저장하는 방식이다.

관계 데이터베이스 기술을 기반으로 한 XML 문서를 저장의 또 다른 접근 방법인 구조사상 접근 방법은 XML 문서의 구조를 표현하는 DTD또는 XML 스키마를 기반으로 생성된 관계데이터베이스의 스키마에 따라 저장되므로, 관계데이터베이스 스키마들을 만족하는 XML 스키마 문서들이 많이 존재하거나 스키마 생성에 사용된 스키마의 수정이 빈번하게 발생하지 않는 경우에 적합하다.

본 연구에서는 후자인 구조 사상 접근 방법을 적용하여 관계데이터베이스에 CityGML 데이터를 저장하는 방법을 연구하였다.

3. Methodology

3.1 CityGML 1.0

CityGML은 개방형 데이터 모델로 XML을 기반으로 하는 가상 3차원 도시 모델의 저장과 교환을 위한 포맷이다. 또한 GML3기반 응용스키마(application schema)이다. CityGML의 개발 목적은 다른 응용분야들 간에 공유할 수 있는 3차원 도시 모델의 공통 기본 항목(entity), 속성(attribute), 관계(relation)들을 정의하는 것이다. CityGML은 GML응용스키마로 구현되기 때문에 ISO 19100 시리즈를 기본으로 참조하고 있으며, W3C의 XML 표준에 관한 여러 문서들을 참조하고 있다.

CityGML은 thematic level과 Geometric-topological level을 중점적으로 데이터 모델을 정의하고 있는데 그림 1은 core module을 나타내는 UML 다이어그램이다. 다이어그램에 표현된 CityObject를 중심으로 한 주요 클래스와 이로부터 확장된 지형, 빌딩, 토지이용, 수역, 식생, generic city, furniture등 다른 thematic 모델들이 포함되는데 이로부터 CityGML이 어떤 도시정보를 표현하고 있는지를 파악 할 수 있다.

이번 프로젝트에서는 geometry model과 core module 그리고 11개의 thematic extension module 중 Appearance, Building, Transportation을 적용하여 수행하였다.

3.2 데이터베이스 설계

앞에서 언급했듯이 본 연구에서는 CityGML 1.0을 따르는 데이터를 저장하고 공간적 특성을 이용한 분석을 하기 위해 상용 DBMS인 Oracle 11g를 사용하였다. Oracle 11g는 기본적인 젯장하고Geometry와 Composite Surface, Simple Solid, Composite Solid, Collection 타입으로 젯장하고Geometry의 표현이 가능하다.

CityGML을 Relational Database에 저장하기 위해서는 필요한 테이블과 그들의 관계를 정의해야 한다. CityGML1.0 의 기본 feature들과 그들의 관계를 정의하기 위해 추가적으로 필요한 테이블을 정의하고 확정된 테이블간의 관계를 정의하여야 한다. 이를 위해서는 우선 관계형 데이터모델로 표현하여야 하는데 그림 2는 그림 1의 CityGML Core module을 기준으로 ER 다이어그램을 표현한 것이다. 관계형 데이터베이스에 데이터를 저장하기 위해서는 그 구조가 자료의 수정, 삭제에 따른 예기치 않은 오류를 최소화하여 데이터구조의 안정성을 최소화하는 정규화가 이루어 져야 한다. 따라서 주요클래스간의 관계성과 데이터 구조를 고려하여 3단계의 정규화 과정으로 구조의 안정성을 위해 추가적으로 필요한 클래스를 정의하고 이들간의 관계를 설정하였다. 그림 1과 그

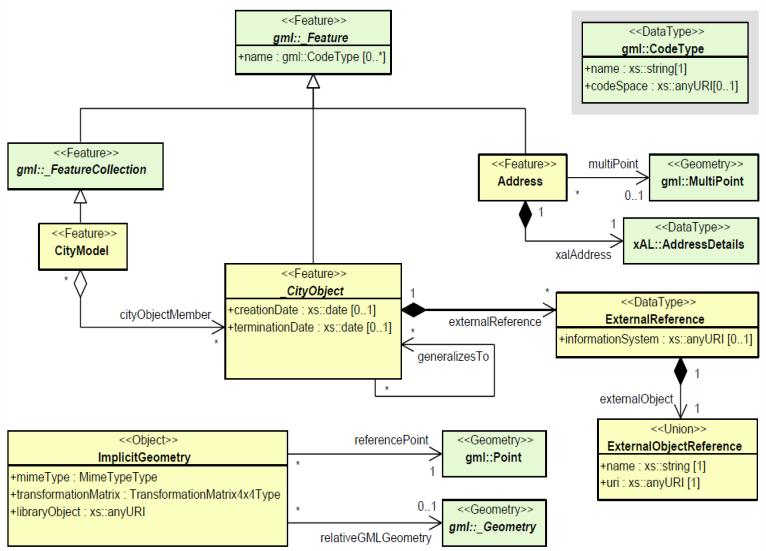


그림 1. CityGML Core Module

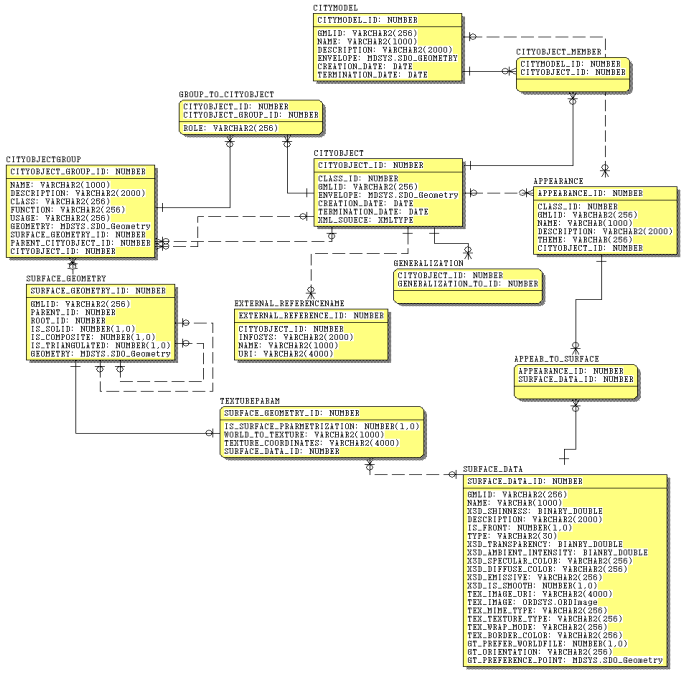


그림 2. CityGML의 Core module을 따르는 ER 다이어그램

림 2를 비교하여보면 주로 aggregation 관계와 association 관계에 추가적인 테이블이 요구된다. 예를 들어 그림 1에서 CityObject feature는 CityModel feature의 menber가 되는데 이때 정의되는 CityModel에 따라 CityObject가 여러 번 사용된 수 있거나 한 번도 사용되지 않을 수 있다. 이때 두 feature에 대해 정의된 클래스에 다수의 반복적 값을 갖게 될 수 있으며 이는 제 1 정규화에 위배되고 이를 해결하기 위해 그림 2의 CITYMODEL과 CITYOBJECT 테이블의 관계를 표현하는 추가적인 테이블인 CITYOBJECT_MEMBER를 생성하여 준다. 이러한 방법으로 데이터베이스를 설계하고 저장하여 보았다. 그림 3은 데이터베이스를 설계하는 과정을 보여주는 흐름도 이다.

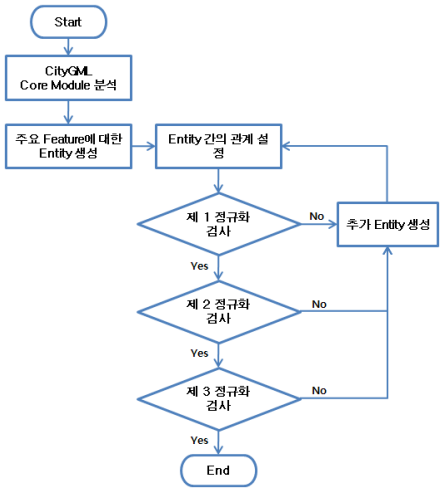


그림 3. 데이터베이스 설계 흐름도

4. Implementation

4.1. 데이터베이스 설계 및 저장

CityGML을 Relational Database에 저장하기 위해서는 필요한 테이블과 그들의 관계를 정의해야 한다. CityGML1.0 의 기본 feature들과 그들의 관계를 정의하기 위해 추가적으로 필요한 테이블을 정의하고 확정된 테이블간의 관계를 정의하여야 한다. Oracle 11g에서 데이터를 다루기 위해서는 SQL 구문을 사용해야 한다. 다음은 CITYMODEL 과 CITYOBJECT 테이블, 관계테이블인 CITYOBJECT_MEMBER를 생성하고 필요한 column들을 만들고 Primary Key를 지정하는 DDL 구문이다.

표 1과 같은 방법으로 테이블을 정의하고 테이블과 테이블 관계를 설정하여 주어 데이터베이스를 설계하였다. 설계된 데이터베이스에 SQL 문을 이용하여 데이터를 저장한다. 그림 4는 샘플데이터와 입력된 테이블을 보여준다. 테이블에서 GEOMETRY column이 모델의 세밀도(Level of Detail)에 따라 표현되는 surface를 설명한다.

표 1. DDL 구문

```
CREATE TABLE CITYMODEL
(
      CITYMODEL_IDNUMBER   NOT NULL,
      GMLID                VARCHAR2(256),
      NAME                 VARCHAR2(1000),
      DESCRIPTION          VARCHAR2(2000),
      ENVELOPE             MDSYS.SDO_GEOMETRY,
      CREATION_DATE        DATE,
      TERMINATION_DATE     DATE,
);
ALTER TABLE CITYMODEL ADD CONSTRAINT CITYMODEL_PK
PRIMARY KEY (CITYMODEL_ID);

CREATE TABLE CITYOBJECT
(
            CITYOBJECT_ID          NUMBER    NOT NULL,
            CLASS_ID               NUMBER    NOT NULL,
            GMLID                  VARCHAR2(256),
            GMLID_CODESPACE        VARCHAR2(1000),
            ENVELOPE               MDSYS.SDO_GEOMETRY,
            CREATION_DATE          DATE NOT NULL,
            TERMINATION_DATE       DATE,
            XML_SOURCE             XMLTYPE
);

ALTER TABLE CITYOBJECT ADD CONSTRAINT CITYOBJECT_PK
PRIMARY KEY (CITYOBJECT_ID);

CREATE TABLE CITYOBJECT_MEMBER
(
            CITYMODEL_ID        NUMBER    NOT NULL,
            CITYOBJECT_ID       NUMBER    NOT NULL
);

ALTER TABLE CITYOBJECT_MEMBER ADD CONSTRAINT
CITYOBJECT_MEMBER_PK PRIMARY KEY (CITYMODEL_ID,
CITYOBJECT_ID);
```

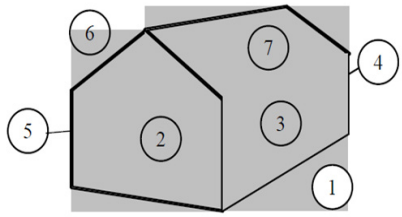
저장된 데이터에 surface를 추가하여 보았다. 그림 5의 짙은 색 두 개의 surface가 추가된 것이고 테이블을 보면 LoD2일 경우 surface '6'과 '9'가 합쳐진 형태 그리고 '7'과 '8'이 합쳐진 형태로 roof 가 표현되는 것을 알 수 있다.

5. 결 론

CityGML 1.0을 따르는 데이터의 저장방법에 대해 연구하여 보았다. 상용DBMS 중 Oracle 11g를 사용하였으며 이는 몇 가지의 3차원 기본 Geometry를 제공한다. CityGML은 LoD의 개념을 지니고 있어 같은 객체라도 level에 따라 표현되는 surface가 달라 이상적인 저장장법을 찾는 것 또한 중요한 연구가 된다.

이번 논문에서는 데이터베이스를 설계하고 저장하는 단계에서 그쳤으나 앞으로 저장된 데이터를 가시화하고 데이터를 응용하여 객체들 간의 topology 관계를 찾는 연구가 진행되어야 할 것이다.

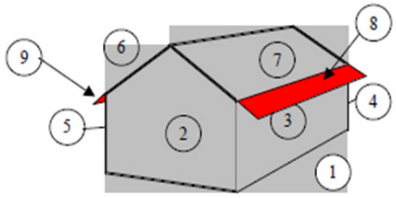
본 논문에서도 추가적인 실험으로 Topology 구조를 이용하여 연결된 객체들을 찾는 분석을 하였다. Geometry로부터 계산하는 방법을 사용하였으나 이 방법은



SURFACE_GEOMETRY

ID	GMLID	PARENT_ID	ROOT_ID	IS_SOLID	IS_COMPOSITE	GEOMETRY
1			1	1	0	
2	LoD1	1	1	0	1	
3			3	1	0	
4	LoD2	3	3	0	1	
5	Left2	4	3	0	1	Geometry of surface 5
6	Left1	2	1	0	0	LoD1 geometry surf. 5
7	Front1	2	1	0	0	LoD1 surface 2
8	Right1	2	1	0	0	LoD1 surface 3
9	Back1	2	1	0	0	LoD1 surface 4
10	Roof1	2	1	0	0	LoD1 surface 6-7 (plain)
11	Left21	5	3	0	0	LoD2 comp surface 5-1
12	Left22	5	3	0	0	LoD2 comp surface 5-2
13	Front2	4	3	0	0	LoD2 surface 2
14	Right2	4	3	0	0	LoD2 surface 3
15	Back2	4	3	0	0	LoD2 surface 4
16	Roof21	4	3	0	0	LoD2 surface 6
17	Roof22	4	3	0	0	LoD2 surface 7
....	....			.....		.....

그림 4. 샘플데이터 및
SURFACE_GEOMETRY 테이블



SURFACE_GEOMETRY

ID	GMLID	PARENT_ID	ROOT_ID	IS_SOLID	IS_COMPOSITE	GEOMETRY
3			...	...	...	
4	LOD2		3	0	1	
5	Left2	3	3	0	1	Geometry of surface 5
11	Left21	4	3	0	0	LoD2 comp surface 5-1
12	Left22	4	3	0	0	LoD2 comp surface 5-2
13	Front2	3	3	0	0	LoD2 surface 2
14	Right2	3	3	0	0	LoD2 surface 3
15	Back2	3	3	0	0	LoD2 surface 4
16	Roof2_1	20	3	0	0	LoD2 surface 6
17	Roof2_r	21	3	0	0	LoD2 surface 7
18	Over_1	20	3	0	0	Geometry of overhang 8
19	Over_r	21	3	0	0	Geometry of overhang 9
20	Roof_1	3	3	0	1	LoD2 comp roof 6 + 9
21	Roof_r	3	3	0	1	LoD2 comp roof 7+8
....		....		.....		.....

BUILDING

ID	....	BUILDING_ROOT_ID	...	LOD1_GEOMETRY_ID	LOD2_GEOMETRY_ID
...	.....	...	.....		
1	.....	1	.....	1	3
....	.....	...	.....		

그림 5. 추가된 데이터 및
SURFACE_GEOMETRY 테이블

상당히 긴 시간이 걸렸다. 차후 Oracle 11g 에서 제공하는 네트워크 데이터를 활용한 방법을 적용하는 방법에 대한 실험을 수행 할 계획이고 네트워크를 사용하거나 데이터베이스에 저장하지 않고 응용 단계에서 적용하는 것이 프로세스 효율을 높이는 하나의 방법이 될 수 있을 것이라고 판단된다.

참 고 문 헌

[1] C. Arens, J. Stoter and P. van Oosterom, “Modelling 3D spatial objects in a geo-DBMS using a 3D primitive”, Calin Arens, Jaintien Stoter, Peter van Oosterom, Computer & Geosciences 31 (2005) pp.165-177.

[2] P. van Oosterom, J. Stoter, W. Quak and S. Zlatanova, “The Balance Between Geometry and Topology”.

[3] G. Groger, T. H. Kolbe, A. Czerwinski and C. Nagel, “OpenGIS CityGML Encoding Standard”, 2008.

[4] 김훈, 홍의경, “객체관계형 데이터베이스를 이용한 XML 문서 저장 모델 설계”. 한국정보과학회 가을 학술발표 논문집(Ⅰ), 제 27권 2호, 2000.

[5] http://www.citygml.org/1539