무결성 제약에 기초한 규칙 기반 데이타베이스 검증 시스템

류 명 춘† 박 창 현††

요 약

데이타베이스 관리 측면에서 데이타의 무결성은 매우 중요한 의미를 가진다. 따라서 특정의 데이타베이스 구축시 무결성 규정들을 반드시 고려하여야 하며, 데이타베이스의 무결성 유지를 위해 데이타베이스 관리자는 지속적으로 데이타베이스의 내용을 감시하면서 데이타의 무결성을 저해하는 요소가 발생될 경우 이를 제거해 주어야만 한다. 그러나 이러한 무결성 유지 작업은 응용 영역 데이타의 양이 증가할수록 더욱 복잡해지며 또한 많은 시간을 필요로 하게 된다. 본 논문에서는 이러한 문제점을 개선하기 위한 규칙 기반 데이타베이스 검증 시스템을 제시한다. 제시하는 규칙 기반 데이타베이스 검증 시스템은 문제 영역 모델로부터 필요한 데이타베이스와 무결성 유지를 위한 규칙들을 생성하고, 추론 시스템과 데이타베이스의 결합을 통해 데이타베이스의 무결성 검증 작업을 수행한다.

A Rule-Based Database Verification System Based on the Integrity Constraints

Myung Chun Ryoo† Chang Hyeon Park††

ABSTRACT

In managing a certain database, the integrity of data is very important. The integrity constraints thus should be considered carefully when a database is designed and, after the database is created, it is required for a database manager to check continuously if some data contained in the database violate the integrity constraints considered. It is however not easy to check the violation of integrity constraints when the size and the complexity of database are increased. This paper suggests a rule-based database verification system to relax the difficulty of checking the integrity violation, in which a database is coupled with a rule-based system including the knowledge about the integrity constraints. The rule-based database verification system suggested accepts the model descriptions of an application domain, generates the knowledge base consisting of rules and facts by analyzing the model description and proceeds the verification process to check the integrity of the database.

1. 서 론

특정 데이타에 대한 사용자들의 일관된 인식속에서 실세계를 모델링하는 것은 데이타베이스 시스템의 중요한 목표중의 하나이다[5]. 따라서 데이타베이스내에 저장된 데이타들의 값은 미리 규정된 무결성 제약 조건(integrity constraints)을 만족해야 한다. 여기서 무결성이란 데이타베이스의 무결함을 의미하는 것이지만 실제로는 데이타의 최소의 중복과 값의 일관성(consistency)을 의미한다[12]. 특정 응용 영역에서 데이타베이스를 활용할 때 데이타베이스 내의 정보에 대한 무

· 이 논문은 1994년도 교육부 지원 학술진흥재단의 자유 공모과제 학술연구 조성비에 의하여 연구되었음.
† 정 회 원 : 영남대학교 대학원 전산공학과 박사과정
†† 정 회 원 : 영남대학교 전산공학과 조교수
논문접수 : 1995년8월23일, 심사완료 : 1995년12월12일

결성을 유지하기 위한 방법으로는, 데이타베이스 관리자가 지속적으로 데이타베이스의 내용을 감시하면서 데이타의 무결성을 저해하는 요소가 발생될 경우 이를 제거해주는 방법과 데이타베이스 구조에 관련된 규칙들을 표현하는 무결성 제약 조건들의 실행을 통한 방법이 있다. 그러나 데이타베이스 관리자에 의한 무결성 유지 방법은 응용 영역 데이타의 규모가 커질수록 무결성 유지 작업이 더욱 복잡해지고 또한 많은 시간을 필요로 하게 된다[3]. 무결성 제약 조건들의 실행을 통한 방법의 경우는, 불행히도 현재 대부분의 DBMS가 자신의 데이타베이스내의 데이타들의 정확도를 보장하기 위한 적당한 무결성 특징들을 제공하지 못하고 있다[5].

관계 데이타베이스의 표준 질의어인 SQL을 사용하는 Oracle 시스템에서는 데이타베이스의 무결성을 유지하기 위해서 테이블 생성 명령어에 무결성 제약 조건을 선언적으로 기술할 수 있도록 허용하여 테이블의 각 필드, 또는 필드와 필드 사이에서 지켜야 하는 조건들을 부여할 수 있도록 하고 있다[7]. 이와 같이 생성된 테이블들에 대해서 데이타의 입력 및 변경 작업이 수행되면 Oracle 시스템은 테이블 생성시에 주어진 제약 조건들과 데이타베이스의 내용을 비교하여 데이타베이스의 무결성을 유지하게 된다. 하지만 SQL을 통한 데이타베이스의 무결성 유지 기능은 응용 데이타베이스내의 구조를 부분적으로 변경해야 할 경우, 해당 테이블과 그에 관련된 무결성 제약 조건들을 모두 갱신해야 한다. 이러한 갱신 작업은 응용 데이타베이스가 대량의 데이타를 취급하는 경우에는 매우 어렵고 많은 시간을 필요로하는 문제점이 있다.

본 논문에서는 이러한 문제점들을 개선하기 위한 방안으로 규칙 기반 시스템(rule-based system)을 결합한 데이타베이스 검증 시스템을 제시한다. 본 논문에서 제시하는 규칙 기반 데이타베이스 검증 시스템은 문제 영역 모델로부터 필요한 데이타베이스와 무결성 유지를 위한 규칙들을 생성하고, 추론 시스템과 데이타베이스의 결합을 통해 데이타베이스의 무결성 검증 작업을 수행하도록 하였다. 문제 영역 데이타베이스의 구조가 변경될 경우에는, 문제 영역 모델에서 해당 부분만을 변경하면 시스템 스스로 데이타베이스와 무결성 규칙들을 변경할 수 있도록 하여 데이타베이스의 관리를 보다 용이 하도록 하였다.

본 논문의 구성은 2장에서 무결성 제약 규정과 지식 기반 시스템에 대해서 언급한다. 3장에서는 본 논문에서 제시하는 규칙 기반 데이타베이스 검증 시스템의 구조와 그 구성 요소들의 역할에 대해서 기술하고, 마지막으로 4장에서 결론을 내린다.

2. 관련 연구

2.1 무결성 제약 규정

데이타베이스의 무결성을 유지하기 위해서는 데이타베이스 설계시에 반드시 따르는 데이타 모델링 과정에서 부터 신중하게 고려되어져야 한다. 무결성 규정[9]은 데이타베이스내의 데이타가 일관성의 유지를 위하여 충족시켜야 하는 제약 조건이다. 기본적으로 데이타베이스의 무결성을 유지하기 위한 무결성 규정은 영역(domain), 관계(relation), 참조(reference), 명시(explicit)의 형태로 존재한다. 영역 제약은 주어진 어떤 필드 값이 데이타베이스에 있는 다른 값들과의 관계를 전혀 고려하지 않고 그 자체로 허용될 수 있는가를 규정하는 것으로 삽입이나 갱신 연산에 적용되는 것이다. 이에 반해, 관계 제약은 어느 한 레코드가 테이블에 삽입될 수 있는가 또는 한 테이블과 다른 테이블의 레코드들 간의 관계가 적절한가를 규정하는 것이다. 참조 제약은 외래키(foreign key)를 갖는 어떤 테이블과 그 외래키에 의해서 참조되는 테이블간의 관계가 적절한가 또는 외래키에 의해서 참조되는 테이블에 삽입 가능한 필드는 어떤 것인가를 규정한다. 명시 제약은 특정 데이타베이스 작업 전후에 테이블이 지켜야 하는 조건을 부여하여 조건에 만족되는 동작을 취하게 하는 규정으로서 실세계(real world)로 부터 요구되는 규정이나 정책을 반영하고 직접적으로 릴레이션 모델과의 관계는 없다[3].

특정의 데이타베이스를 구축할 경우 반드시 이러한 무결성 규정들을 고려하여, 데이타베이스

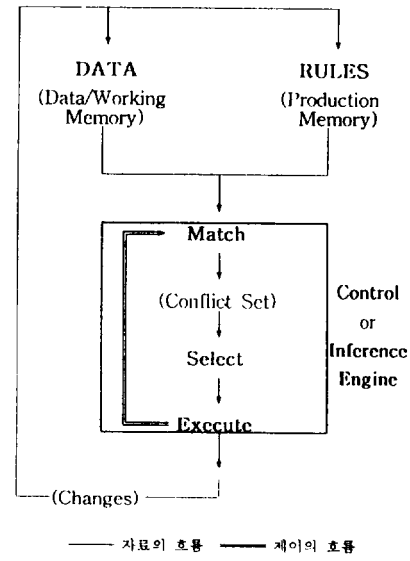
관리자는 지속적으로 데이타베이스의 내용이 이러한 규정들을 위반하지 않는지 감시하여야 한다. 본 논문에서는 데이타베이스의 무결성 검증 작업에 사용하기 위해 이들 제약 규정 이외에 열거형(enumeration)데이타를 다룰 수 있는 열거형 제약 규정과 사용자가 직접 해당 필드의 값의 범위를 지정할 경우 취급되는 데이타가 해당 범위를 위반 하는지의 여부를 조사하는 등 보다 일반적인 사항들을 추가하여 광범위한 무결성 검증 시스템을 제공한다.

2.2 지식 기반 시스템

지식 기반 시스템(knowledge-based system)은 특정 영역에 대한 지식을 축적하고 있는 지식베이스의 지식을 이용하여 추론 기관에 의한 추론 과정에 의해 결론을 도출함으로서, 사용자로부터 요구되는 문제를 해결하는 컴퓨터 시스템이다[8, 10]. 지식 기반 시스템의 성능은 특정 영역의 지식을 표현할 수 있는 지식 표현 방법과 그 지식을 효과적으로 이용하기 위한 추론 기관의 성능에 달려 있다.

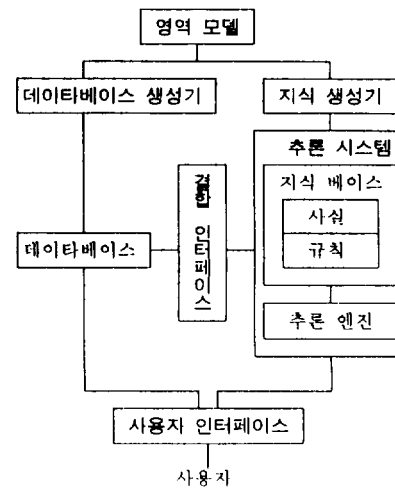
지식 표현 방법으로는 가정(premise 또는 if-part)과 결론(conclusion 또는 then-part)의 문장으로 표현되는 규칙(rules), 실세계의 객체들의 관계를 망 구조로 표현하는 의미망(semantic network), 프레임이라고 하는 객체의 구조와 슬롯(slot)이라고 하는 객체의 애트리뷰트의 묘사에 중점을 둔 프레임(frame), 수학이나 논리학에서 사용되었던 논리(logic)를 이용하는 방법 등이 있다. 이러한 지식 표현 방법 중에서 규칙을 사용하는 지식 기반 시스템을 규칙 기반 시스템이라고 한다. 일반적으로 규칙 기반 시스템은 현재의 상태를 나타내는 데이타들을 저장하는 작업 메모리(working memory)와 if-then형태의 규칙들, 그리고 추론 기관으로 이루어진다[2]. 규칙 기반 시스템의 구조는 (그림 1)과 같다.

추론 기관은 지식베이스의 사실과 규칙을 이용하여 추론하기 위한 모듈로서, 추론 전략과 제어를 포함하고 있다. 추론 과정 중 생성하는 사실은 작업 메모리에 저장되어 사용된다. 추론 방법에는 주어진 사실로 부터 결론을 도출하는 전향



(그림 1) 규칙 기반 시스템의 구조
(Fig. 1) Architecture of a rule-based system

추론(forward inferencing), 주어진 목표를 만족하기 위하여 부목표(subgoal)를 설정하여 추론하는 후향 추론(backward inferencing), 두 방향이 혼합된 혼합 추론(hybrid inferencing)이 있다[6]. 본 논문에서는 지식베이스의 규모가 클 때 효율적인 패턴 매칭을 수행하는 Rete알고리즘[1]을 기반으로 한 전향 추론 엔진을 사용한다.



(그림 2) 규칙 기반 데이타베이스 검증 시스템의 구조
(Fig. 2) Architecture of the rule-based database verification system

3. 규칙 기반 데이타베이스 검증 시스템의 구조

본 논문에서 제시하는 규칙 기반 데이타베이스 검증 시스템의 구조는 (그림 2)와 같이 영역 모델, 데이타베이스 생성기, 지식 생성기, 추론 시스템, 결합 인터페이스와 사용자 인터페이스로 구성된다.

3.1 영역 모델

특정 응용 영역의 데이타베이스를 구축하는데 필요한 데이타 모델은 구조(structure)적 요소와 제약 조건으로 구성된다. 구조적 요소는 데이타베이스내에 존재하는 개체(entity)와 개체들간의 관계, 그리고 각 개체의 애트리뷰트와 애트리뷰트간의 관계를 정의한 것으로서 데이타베이스의 정적 성질을 나타낸다. 이에 반해 제약 조건은 데이타베이스에 허용되는 개체 어커런스(occurrence)에 대한 논리적 제한을 명세한 것으로서 데이타베이스 작업시 데이타 조작을 제어하는 규칙들이 된다.

본 논문에서 제시하는 규칙 기반 데이타베이스 검증 시스템은 데이타베이스를 구축하고 관리하는데 필요한 모든 정보들을 데이타베이스의 구조 및 제약 조건들을 포함하는 영역 모델(domain model)로 기술하여, 이를 무결성 유지 작업에서 이용한다. 영역 모델의 기술 방법은 데이타베이스와 지식베이스가 동일한 문제 영역에서 공통적으로 정보를 사용할 수 있도록 고려하였다. 사용자는 영역 모델만 기술해 주면 시스템이 자동으로 데이타베이스와 지식베이스를 생성해 준다. 다음은 본 논문의 규칙 기반 데이타베이스 검증 시스템이 지원하는 영역 모델 기술 방법의 예이다.

```
table t1 (
  name c 20 primary_key,
  no i 10 ref_table: t2,
  age i 5
)

table t2 (
  no i 10 primary_key,
  address c 30
)
  ...

table t10 (
  name c 20 enum: ("tom","john","sam",
                   "jorge" ) primary_key,
  telephone c 20
)

table t20 (
  name c 20 primary_key,
  property i 10 valid: (100, 100000)
)
```

위의 기술 방법에서 "table"은 데이타베이스내에 생성되는 테이블의 시작을 의미하며 "table" 다음에 나오는 스트링은 실제 생성되는 테이블명이 된다. "("와 ")" 사이에 기술되는 내용들은 생성되는 테이블의 필드명과 필드의 속성 그리고 해당 테이블의 무결성 제약 조건이 된다. 테이블 t1을 기술하는 영역 모델의 경우, 먼저 table t1은 생성하는 테이블명이 t1임을 의미하며, 필드를 기술하는 "name c 20"은 필드명이 name이고 필드의 타입은 문자형이며 필드의 길이는 20을 의미한다. 다음의 "primary_key"는 name 필드가 t1테이블의 기본 키임을 가리키게 되는데 이것은 t1테이블의 무결성 제약 조건중의 하나로서 t1테이블의 무결성 검증을 위한 규칙으로서 지식베이스 내에 저장된다.

3.2 데이타베이스 생성기

규칙 기반 데이타베이스 검증 시스템의 데이타베이스 생성기는 주기억 장치에 적재된 영역 모델에 대해 DBMS에서 테이블을 생성하는데 필요한 정보, 즉 영역 모델의 구조적 요소들을 추출하여 재구성함으로서 영역 모델의 데이타베이스를 구성하는 테이블들을 생성하기 위한 SQL 명령어를 생성하게 된다. 생성된 SQL명령어를 DBMS가 실행함으로서 영역 모델의 데이타베이스 생성이 이루어지며, 사용자는 생성된 데이타베이스에 대해 원하는 데이타의 검색, 삽입, 삭제, 갱신 등의 작업을 수행하게 된다.

다음은 주어진 영역 모델을 분석하여 데이타베

이스 생성기가 출력한, 테이블 생성을 위한 SQL 명령어이다.

```
create table t1 (
        name        char(20),
        no          number(10),
        age         number(5)
);

create table t2 (
        no          number(10),
        address     char(30)
);
        ...

create table t10 (
        name        char(20),
        telephone   char(20)
);

create table t20 (
        name        char(20),
        property    number(10)
);
```

3.3 지식 생성기

지식 생성기는 데이타베이스 변경시 추론을 통한 무결성 검증을 위해 필요한 지식베이스를 구축하기 위하여 영역 모델의 제약 조건들을 분석하여 규칙들을 생성한다. 일반적으로 규칙의 형태는 조건부(condition)와 실행부(action)로 표현되며, 조건부가 만족되는 경우에만 해당 규칙이 실행될 수 있다. 지식 생성기는 제약 조건들을 분석하여 이를 규칙 형태로 변환함으로서 지식베이스를 형성하고, 데이타베이스 무결성 검증을 위해 이를 추론 시스템에서 이용할 수 있도록 한다. 본 논문의 지식 생성기는 무결성 제약 조건을 관계, 영역, 참조, 명시 등으로 분류하여 영역 모델로 부터 규칙을 생성한다.

다음은 앞에서 주어진 영역 모델로 부터 지식 생성기가 생성한 규칙들의 예이다.

```
ruleclass RB_integrity()
{
    comment: "This ruleclass is for diagnosing
              DBASE."
    direction: forward
}
        ...
rule Relation_integrity17865(RB_integrity)
{
 for: (?a in t1)
 if:  (?a name = NULL)
 do: (print "Error in t1:: The primary key of t1")
     (print "is null value.\n")
}
        ...
rule Reference_integrity11221(RB_integrity)
{
 for: (?a in t1)
      (?b in t2)
 if:  (?a no = ?r0)
     -(?b no = ?r0)
 do: (print "Error in t1:: The reference key,")
     (print "(no=", ?a.no, " ")
     (print ") does not exist in t2\n")
}
        ...
```

위에서 기술하고 있는 내용을 간단히 설명하면 다음과 같다. 먼저 "ruleclass RB_integrity() { ··· }" 는 RB_integrity라는 규칙클래스를 선언한 형태이다. 다음으로 "rule Relation_integrity17865(RB_integrity) { for: ··· if: ··· do: ··· }"는 규칙클래스 RB_integrity의 인스턴스인 규칙 중의 하나이다. 여기에서 for 부분(변수 범위 선언부)의 "(?a in t1)"이 의미하는 것은 ?a라는 변수는 사실클래스 t1에 속하는 사실, 즉 사실클래스 t1에 속하는 인스턴스라는 선언적 의미이다. if 부분(조건부)의 "(?a name = NULL)"은 ?a의 name 애트리뷰트 값이 NULL인지를 확인하는 부분으로 만약 NULL일 경우 기본 키가 가지는 관계 무결성 제약 조건에 위배되며, 규칙 기반 데이타베이스 검증 시스템은 do 부분(결론부)을 실행하여 기본 키 값이 NULL임을 알리는 에러 메세지를 출력하게 된다. "rule Reference_integrity11221 (RB_integrity) { ··· }"는 사실클래스 t1에 속하는 변수 ?a의 no애트리뷰트 값이 사실클래스 t2에 속하는 변수 ?b의 no애트리뷰트에 존재하지 않을 경우 참조 무결성 제약 조건에 위배됨을 알린다.

3.4 추론 시스템

추론 시스템은 사실베이스(factbase), 규칙베이스(rulebase), 추론 기관(inference engine)으로 구성된다.

사실베이스는 응용 영역 문제를 해결하기 위해서 규칙에 이용되는 객체들에 대한 정보를 저장한다. 본 논문에서는 객체 지향적 방법을 이용하여 관련 사실들의 집합을 사실클래스로 선언하고 그의 인스턴스인 사실은 사실클래스의 속성을 표현하는 각 애트리뷰트에 대해 실제값을 저장함으로서 사실베이스를 구축한다. 규칙 기반 데이타베이스 검증 시스템의 사실베이스 객체들은 이름에 의하여 구분되므로 사실클래스와 사실의 이름은 작업 메모리 내에서 중복될 수 없다. 또한 사실클래스는 속성 패시트(facet)를 가질 수 있고, 사실은 자신이 속한 사실클래스의 속성을 계승받는다. 본 논문의 규칙 기반 데이타베이스 검증 시스템은 영역 모델이 입력되면 시스템이 자동적으로 영역 모델의 객체와 사실클래스를 일대일로 대응시켜 사실클래스를 생성해 낸다. 생성된 사실클래스들은 데이타베이스의 데이타들을 다운로드(download) 받게 되면 그의 인스턴스인 사실들이 생성되어 무결성 규칙 기반의 추론시에 데이타베이스에 오류가 있는지를 탐지해 낸다.

규칙베이스는 추론에 필요한 규칙들의 모임이며, 규칙클래스와 그의 인스턴스인 규칙들로 이루어진다. 규칙클래스는 관련된 유사한 작업을 수행하는 규칙들의 집합이다. 규칙 기반 데이타베이스 검증 시스템에서 영역 모델로 부터 자동 생성되는 무결성 규칙들은 모두 데이타베이스의 무결성을 검증하는 규칙들로서, 규칙클래스 RB_integrity에 속하게 되어 추론시에 특정 영역 데이타베이스의 무결성 검증에 사용된다.

추론 방법에는 주어진 사실로부터 결론을 도출하는 전향 추론, 주어진 목표를 만족하기 위하여 부목표를 설정하여 추론하는 후향 추론, 두 방향이 혼합된 혼합 추론이 있다. 본 논문에서는 Rete 기반의 전향 추론 엔진을 사용한다[1].

3.5 결합 인터페이스

추론 시스템과 데이타베이스 사이의 결합 인터페이스는 두 시스템 사이에 필요한 데이타를 전송하고, 각 시스템의 표현 방법에 적절하도록 데이타 변환 작업을 수행한다. 추론 시스템이 데이타베이스에 저장되어 있는 데이타를 추론에 이용하기 위해서는 사실베이스의 구조와 데이타베이스의 구조를 대응시켜야 한다. 두 구조의 대응 관계는 다음과 같다.

관계형 데이타베이스		사실베이스
Table	⟷	사실클래스
Record	⟷	Fact(Instance)
Field	⟷	Attribute

문제 영역의 데이타베이스로 부터 필요한 정보를 가져오기 위해서는 테이블과 사실클래스를 대응시키는 Class_map과 테이블의 각 필드를 사실클래스의 애트리뷰트에 대응시키는 Slot_map을 만들어야 한다. 이들 Class_map과 Slot_map 사상 정보를 시스템 지식(system knowledge)이라 하며, 이들은 최상위 클래스인 DB—SYS의 하위 클래스가 된다. DB_SYS는 모든 클래스들이 계승할 수 있는 애트리뷰트들을 포함하고 있어서 하위의 모든 객체들은 이 애트리뷰트를 정의 없이 사용할 수 있다. DB_SYS의 구조는 다음과 같다.

```
Factclass DB_SYS()
{
    PRIMARY0:
    PRIMARY1:
    PRIMARY2:
    PRIMARY3:
    PRIMARY4:
    REFERENCE0:
    REFERENCE1:
    REFERENCE2:
    REFERENCE3:
    REFERENCE4:
    ...
}
```

Class_map의 DBname 애트리뷰트는 작업중인 데이타베이스의 이름을 기술하고 TBname과 CLname은 대응되는 각각의 테이블 이름과 사실클래스의 이름을 기술한다. condition 애트리뷰트는 시스템간의 통신에 드는 비용을 줄이기 위해 필요한 데이타만을 로드하도록 테이블에 부여하는 조건을 기술한다. 데이타베이스와 추론 시

스템의 데이타 통신은 download와 upload함수에 의해 실제로 데이타가 교환된다. download는 데이타베이스로 부터 사실베이스로 데이타를 전송하는데, Class_map의 download 필드가 "yes"의 값을 가질 때, 데이타베이스의 해당 테이블의 레코드들이 사실베이스의 사실들로 전송된다. upload는 이와 반대 방향으로 데이타가 전송된다. Class_map의 구조는 다음과 같다.

```
Factclass Class_map(DB_SYS)
{
  DBname:
      [valuetype: string]
      [inheritance: instance]
  TBname:
      [valuetype: string]
      [inheritance: instance]
  CLname:
      [valuetype: string]
      [inheritance: instance]
  condition:
      [valuetype: string]
      [inheritance: instance]
  upload:
      [valuetype: string{"yes","no"}]
      [inheritance: instance]
  download:
      [valuetype: string{"yes","no"}]
      [inheritance: instance]
}
```

Slot_map 클래스에는 하나의 애트리뷰트가 하나의 필드에 어떻게 대응하는지에 대한 정보가 하나의 인스턴스로서 표현된다. 이때 map 애트리뷰트는 이 대응 관계와 관련된 클래스와 테이블 사이의 사상 이름을 나타낸다. field_name과 attribute_name은 각각 대응되는 테이블의 필드와 사실클래스의 애트리뷰트를 기술하며, condition 애트리뷰트는 해당 필드가 사상되는 조건을 표시한다. key 애트리뷰트는 필드의 기본 키(primary key) 여부를 나타낸다. 이때 애트리뷰트가 기본 키이면 최상위 DB_SYS의 PRIMARYn: 애트리뷰트를 계승받아 그 값을 복사할 수 있다. Slot_map의 구조는 다음과 같다.

```
Factclass Slot_map(DB_SYS)
{
  map:
      [valuetype: object]
      [inheritance: instance]
  field—name:
      [valuetype: string]
      [inheritance: instance]
  attribute—name:
      [valuetype: string]
      [inheritance: instance]
  condition:
      [valuetype: string]
      [inheritance: instance]
  key:
      [valuetype: string]
      [inheritance: instance]
}
```

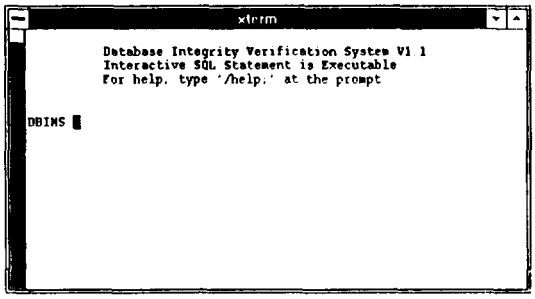
3.6 사용자 인터페이스

사용자 인터페이스는 데이타베이스 조작 및 추론 시스템의 조작을 지원할 수 있는 명령어 해석 및 실행 모듈이다. 규칙 기반 데이타베이스 검증 시스템의 사용자 인터페이스는 Oracle 시스템에서 응용 프로그래머용으로 지원하는 프리컴파일러 인터페이스를 사용하여 C 언어에 SQL언어로 기술해서 짜넣은 문(statement)을 사용해서 C와 SQL의 양쪽 기능을 통합하여 구현하였다. 또한 보다 다양한 환경에서도 잘 동작하도록 하기 위해 실행시에 입력되는 모든 SQL문을 받아들일 수 있도록 동적문(dynamic statement)[11]을 사용하여 프로그래밍 하였으므로, 사용자는 사용자 인터페이스의 프롬프트 상에서 Oracle 시스템의 SQL 언어를 구사하여 데이타베이스를 조작할 수 있도록 하였다. 본 논문에서는 사용자 인터페이스를 데이타베이스와 추론 시스템을 동시에 운용하면서 항상 데이타베이스의 무결성을 유지할 수 있도록 하였다. 또한 보다 융통성있는 시스템 운용을 위해 두 가지의 처리 모드를 마련하고 있다. 먼저 즉시처리 모드(mode)는 프롬프트상에서 의사 명령어 "/dr;"를 이용하여 특정 테이블에 대해 데이타들이 조작(입력,삭제,갱신)되는 즉시 조작 데이타와 현존하는 데이타베이스를 비교하여 무결성 제약 규정에 적합할 경우에만 데이타베이스에 조작 데이타를 반영하게 된다. 일괄처리 모드는 의사 명령어 "/indr;"를 사

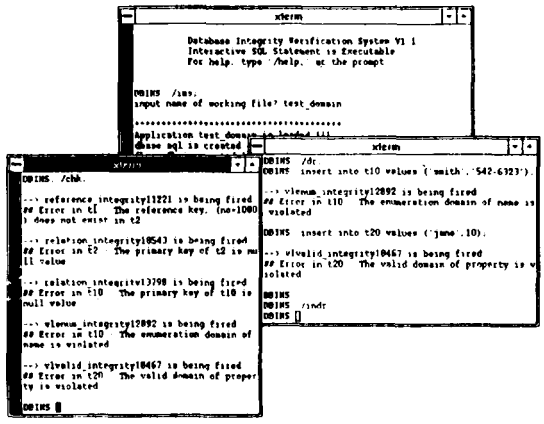
용하여 필요로 하는 모든 데이타를 조작한 다음, 의사 명령어 "/chk;"를 사용하여 주기적으로 데이타베이스의 무결성을 검증하는 방법으로 만약 무결성 제약 규정을 위반하는 데이타들이 발생하게 되면 시스템이 자동적으로 이를 처리하게 된다.

(그림 3)은 본 논문의 규칙 기반 데이타베이스 검증 시스템의 실행시 초기 화면이다.

사용자 인터페이스 프롬프트(DBIMS:) 상에서 의사 명령어 "/ims;"를 입력하여 영역 모델의 입력 모드로 들어갈 수 있다. 3.1절에서 기술한 영역 모델을 입력으로 하였을 경우 데이타베이스 무결성 검증 시스템의 실행과정을 살펴보면 (그림 4)와 같다.



(그림 3) 규칙 기반 데이타베이스 검증 시스템의 초기 상태
(Fig. 3) Initial state of the rule-based database verification system



(그림 4) 규칙 기반 데이타베이스 검증 시스템의 실행 예
(Fig. 4) Execution of the rule-based database verification system

(그림 4)에서 즉시처리 모드(/dr;)인 경우, insert 명령에 의해 테이블 t10의 name 필드의 값을 "smith"로 입력하게 되면 규칙 기반 데이타베이스 검증 시스템은 테이블 t10의 제약 조건을 만족하는지 검사함으로서 name 필드에서 에러가 발생했음을 사용자에게 통보한다. 이와 같이 즉시처리 모드인 경우는 데이타베이스 갱신 명령이 발생하면 그 즉시 무결성 여부를 검사하게 된다. 일괄처리 모드(/indr;)인 경우에는, 사용자가 필요로하는 모든 데이타를 조작한 다음 (그림 4)의 왼쪽과 같이 의사 명령어 "/chk;"에 의해서 일괄적인 무결성 검증 작업이 실행된다. 〈표 1〉은 (그림 4)의 작업에서 사용된 각 테이블의 구조와 요소들에 대한 리스트를 표시하고 있으며 각 테이블에 대한 제약 조건은 3.1절에서 기술하고 있다.

〈표 1〉 영역 데이타베이스의 리스트
〈Table 1〉 List of the Domain Database

테이블 t1			테이블 t2	
name	no	age	no	address
John	1001	20	1001	Florida
Anna	1080	18	1100	Hawaii
Smith	1100	22	1120	Nevada
Henry	1120	21	1150	Oregon
Michael	1150	23	1200	Arizona
Brown	1200	21	NULL	Utah
...	...	...	...	...
...			...	
테이블 t10			테이블 t20	
name	telephone		name	property
Sam	750 - 1776		Brian	10000
NULL	234 - 5678		Andrew	15000
Tom	523 - 8162		Jane	50
John	634 - 3344		Gilbert	75000
Jorge	425 - 1909		Ivan	60000
			Alice	90000
			...	...

4. 결론 및 고찰

데이타베이스 관리 측면에서 데이타의 무결성은 매우 중요한 의미를 가진다. 특정의 데이타베이스 구축시에는 무결성 규정들을 반드시 고려하여야 하며, 데이타베이스의 무결성 유지를 위한 지속적인 노력이 요구된다. 하지만 데이타베이스

의 응용 분야가 넓어지고 저장되는 데이타가 대규모화됨에 따라 데이타베이스 관리 시스템은 데이타베이스의 무결성 유지 작업에 많은 경비와 시간을 투자해야 하는 어려움을 갖게 되었다. 본 논문에서는 이러한 문제점을 개선하기 위해 무결성 제약에 기초한 규칙 기반 데이타베이스 검증 시스템을 제시하였다.

관계 데이타베이스의 표준 질의어인 SQL을 사용하는 Oracle 시스템에서는 데이타베이스의 무결성을 유지하기 위해서 테이블 생성 명령어에 무결성 제약 조건을 선언적으로 기술할 수 있도록 하여 테이블의 각 필드, 또는 필드와 필드 사이에서 지켜야 하는 조건들을 부여할 수 있도록 하고 있다. 생성된 테이블에 대해서 데이타의 입력 및 갱신 작업이 수행되면 Oracle시스템은 테이블 생성시에 주어진 제약 조건들과 데이타베이스의 내용을 비교하여 데이타베이스의 무결성을 유지한다. Oracle 시스템의 선언 무결성은 기본(primary)키/유일성 제약(unique constraints)과 참조 무결성 제약, 검사 제약(check constraints)의 시행에만 제한되어 있는 단점이 있다[7]. 또한 각 필드에 관한 도메인 무결성 유지시에는 단순한 타입(type) 검사만을 하는데 비해, 본 논문에서 제시하는 규칙 기반 데이타베이스 검증 시스템은 각 필드의 타입 뿐만 아니라 값의 범위와 참조 집합등 영역의 일반적인 성질까지도 조사하게 된다. 예를 들어 3.6절에서 설명한 바와 같은 열거형(enumeration) 타입의 경우에도 본 논문에서 제시하는 시스템에서는 해당 필드의 제약 조건에 위배되는 요소들을 정확히 파악하여 사용자에게 통보해 줄 수 있다. 또한 사용자 인터페이스를 데이타베이스와 추론 시스템을 동시에 운용하면서 항상 데이타베이스의 무결성을 유지할 수 있도록 하였으며, 시스템을 보다 융통성있게 운용하기 위해서 두 가지의 처리 모드를 마련하고 있다. 먼저 즉시처리 모드는 사용자 인터페이스 프롬프트상에서 데이타가 조작(입력,갱신,삭제)되는 즉시 해당 데이타와 현존하는 데이타베이스를 비교하여 무결성 제약 규정에 적합할 경우에만 데이타베이스에 조작 데이타를 반영하게 된다. 일괄처리 모드인 경우에는 필요로 하는 모든 데이타를 조작한 다음 주기적으로 데이타베이스의 무결성을 검증하는 방법으로 만약 무결성 제약규정을 위반하는 데이타들이 발생하게 되면 시스템이 자동적으로 이를 처리하게 된다.

관계 데이타베이스 관리 시스템은 응용 데이타베이스의 구조를 부분적으로 변경해야 할 경우, 해당 테이블과 그에 관련된 무결성 제약 조건들을 모두 갱신해야 하는데, 이러한 갱신 작업은 응용 데이타베이스가 대량의 데이타를 취급하는 경우에는 매우 어렵고 많은 시간을 필요로 한다. 하지만 본 논문의 규칙 기반 데이타베이스 검증 시스템은 문제 영역의 모델을 데이타베이스와 지식베이스 생성 작업에 대한 지식으로서 이용할 뿐만 아니라 데이타베이스와 지식베이스 갱신 작업에 대한 지식으로도 활용할 수 있으므로, 응용 데이타베이스의 구조 변경시 응용 영역 모델의 해당 부분만을 수정하면 시스템 스스로 문제 영역 모델과 데이타베이스 및 지식베이스를 비교하면서 데이타베이스와 지식베이스를 갱신할 수 있다. 이러한 데이타베이스 자동 갱신 능력은 처리해야 할 데이타가 증가할수록 그 효율이 높아지기 때문에 규칙 기반 데이타베이스 검증 시스템은 대규모 데이타베이스의 무결성 유지에 적합하며 이를 효과적으로 지원할 수 있을 것으로 기대된다.

참 고 문 헌

[1] Charles L. Forgy, "Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Problem," Jounal of Artificial Intelligence, Vol. 19, No. 1, pp. 17-37, 1982.

[2] Lee Brownston, Robert Farrell, Elaine Kant, Nancy Martin, 'Programming Expert Systems in OPS5', Addison-Wesley, 1985.

[3] David Bell, Jane Grimson, 'Distributed Database Systems', Addison-Wesley, 1992.

[4] Michael Stonebraker, "The Integration of Rule Systems and Database Systems",

IEEE Transactions on Knowledge and Data Engineering, Vol. 4, No. 5, pp. 415-423, 1992.

[5] Magdi N. Kamel, "A Prototype Rule-Based Front End System for Integrity Enforcement in Relational Database: An Application to the Naval Aircraft Flight Records Database", Proceedings of the Database and Expert Systems Applications, pp. 713-723, 1994.

[6] Adrian A. Hopgood, 'KNOWLEDGE-BASED SYSTEMS for Engineers and Scientists', CRC Press, Inc., pp. 57-85, 1993.

[7] Oracle Systems Korea Technical Support Group, 'ORACLE Tips & Techique', pp. 71-79 , 1993.

[8] Michael L. Brodie, John Mylopoulos, 'On Knowledge Base Management Systems: Integrating Arificial Intellignece and Database Technologies', Springer-Verlag, 1986.

[9] Subrata Kumar Das, 'Deductive Database and Logic Programming', pp. 275-287, Addison-Wesely, 1992.

[10] Jeffrey D. Ullman, 'Principles of Database and Knowledge-Based Systems', Vol. 1, Computer Science Press, 1988.

[11] 한국 오라클, 'Pro*C 유저 가이드', 버젼 1.1, 1991.

[12] 오해석, '인공지능 데이타베이스', 희중당, 1991.

류 명 춘

1985년~89년 영남대학교 공과대학 전산공학과 졸업.
1989년~91년 영남대학교 대학원 전산공학과 전산기시스템 전공 석사.
1993년~현재 영남대학교 대학원 전산공학과 박사과정.
관심분야: 지식기반시스템, 지식획득, 데이타베이스

박 창 현

1982년~86년 경북대학교 공과대학 전자공학과 졸업.
1986년~88년 서울대학교 자연과학대학 계산통계학과 전산학전공 석사.
1988년~92년 서울대학교 자연과학대학 계산통계학과 전산학전공 박사.
1992년~93년 서울대학교 컴퓨터신기술공동연구소 특별연구원.
1993년~현재 영남대학교 전산공학과 조교수.
관심분야: 인공지능(Knowledge-Based System, Machine Learning, Planning)