트리형 데이터베이스 및 언어 MUMPS 활용

임지현 · 김진덕*

동의대학교

Study on Tree-Structured Database and Language MUMPS

Ji-Hyeon Im, Jin-Doeg Kim*

Dong-Eui University

E-mail : ljh950914@naver.com, jdk@deu.ac.kr

요 약

데이터베이스(Database)란 중복이 없는 데이터의 집합을 유지하는 것을 말하며, 정보의 양이 증가하고 있는 정보화 사회에서 정보를 쉽게 이용하고 공유하기 위해서 필수적이다. 데이터베이스의 대표적인 구조는 관계형 데이터베이스와 트리구조 데이터베이스가 존재하며, 본 연구에서는 트리구조 데이터베이스이자 프로그래밍 언어인 MUMPS에 대해 연구하고자 한다. 이 언어는 동적 또는 B-Tree 형식으로 배열을 저장하여 데이터베이스를 구성한다. Java나 C#과 같은 언어와 함께 사용하여야 하는 SQL과 달리, MUMPS는 자체적으로 언어와 데이터베이스를 지원하여 데이터를 관리할 수 있다는 차이점이 있으며, 데이터 이식률이 높아 실제 미국 병원에서는 MUMPS 기반의 플랫폼이 높은 시장 점유율을 차지하고 있다.

ABSTRACT

A Database is a collection of data that does not have redundancy, and it is essential to easily use and share information in an information society where the amount of information is increasing. A typical structure of a Database is a relational database and a tree-structure Database. This research studies the programming language MUMPS, which is a tree structure database. This language constructs the database by storing arrays in a dynamic or B-Tree format. Unlike SQL, which must be used in languages such as Java and C #, MUMPS supports language and database independently and can manage data, so the data porting rate is high. In fact, in U.S. hospitals, the MUMPS-based platform has a high market share.

키워드

데이터베이스, 트리구조, MUMPS, Global Array

Ⅰ. 서 론

인터넷 발달의 특성 중 하나는 사람의 힘으로 처리할 수 없는 많은 데이터에 있다. 이를 유용하게 사용하기 위해서는 처리 및 가공이 필요하며 그 과정에서 데이터에 중복 및 무결성이 위배되는 경우가 있다. 이를 능동적으로 처리하기 위해서는 데이터베이스를 통한 데이터관리가 필수적이다[2]. 대표적으로 관계형 데이터베이스와 트리구조 데이터가 존재한다. 흔히 대중적으로 사용되는 오라클, MySQL 등은 관계형 데이터베이스이며 이는 관계형 데이터형식으로 구성되고, 독자적인 데이터베이스 구축이 되지 않아 자바, C# 등의 외부 프로그램을 사용하여야 한다.

반면 MUMPS는 인터프리터 언어로서 데이터베이스 관리 시스템을 내장하고 있다[1].

실제 미국 병원들의 진료 정보 교류시스템들이 MUMPS 기반의 VistA 플랫폼을 사용하며, 데이터 이식률이 좋아 높은 시장 점유율 가지고 있다.

본 연구에서는 트리구조인 MUMPS 언어에 대해 알아보고, MUMPS를 활용한 간단한 프로그램을 제작하여 데이터베이스와 VistA에서 어떻게 활용되는지 알아보고자 한다.

* corresponding author

Ⅱ. MUMPS

MUMPS(Massachusetts General Hospital Utility Multi-Programming System)는 1966년 Massachuset -ts 종합 병원에서 개발된 언어로 몇 년 후, DEC PDP-7 기반으로 구축되었으며 주로 의료 업무에 많이 사용하고 있다. MUMPS는 대표적으로 GT.M, Caché 플랫폼을 지원한다. 이는 기본적으로 ASCII Code로 구축되지만, 구축 시 ICU, UTF-8을 적용하면 유니코드도 사용할 수 있다. MUMPS는 유연하고 강력한 문자열 조작이 가능하며, Postconditional 기능 및 Pattern Match Operator와 주로 축약된 명령어를 사용한다. 과거 부족한 메모리로 인해 매우 간결하게 표현하도록 한 것이 지금의 축약으로 사용할 수 있게 되었다. Postconditinal과 축약의 예시는 다음과 같다.

```
VEHU6>S VAR1=2 ; 축약사용
   SET VAR1=2 로써, VAR1에 2를 저장하겠다는 의미
VEHU6>S VAR2=3

VEHU6>S:VAR1=VAR2 I=1 ; Postconditinal
   VAR1과 VAR2의 값이 일치하다면 I=1로 SET 하라는 의미
VEHU6>W I ; WRITE I, I 값 출력
%YDB-E-LVUNDEF, Undefined local variable: I
   VAR1과 VAR2의 값이 불일치 하므로 I=1 SET 불가
VEHU6>S VAR1=3

VEHU6>S:VAR1=VAR2 I=1

VEHU6>W I
1
```

그림 1. Postcondital 축약 사용

사진과 같이 MUMPS는 대부분 대문자로 쓰여지며, S VAR와 S var는 다른 변수로 인식한다. MUMPS 내에선 Global, Local 2가지의 변수가 존재하며, Global 변수는 디스크나, 외장 장치에 저장되어 프로그램 종료 후에도 계속 존재하는 반면 Local 변수는 프로그램 내에서 생성되어 프로그램 종료 후에는 사라진다. Block 내에서만 사용하기 위한 변수로는 NEW를 선언하여 사용 할 수 있다. 또한 언어의 특이점으로 연산은 엄격히 왼쪽에서 오른쪽 순서로 수행되며 사칙이나 논리연산의 우선순위를 따르지 않는다. 그의 예시는 다음과 같다.

```
VEHU6>s a=2,b=30

VEHU6>w a<10&b>20
0
VEHU6>w (a<10)&(b>20)
1
VEHU6>w 2+3*5
25
```

그림 2. 연산 우선순위

Ⅲ. Global Array

^를 사용하여 표현하고, 디스크에 저장되어 모든 프로세스에서 사용할 수 있으며 프로세스가 종료되더라도 영구적으로 유지된다. Index 자체가 데이터 자체인 경우가 대부분이고, B Trees를 사용하고 있으며 Pointer Blocks과 데이터 Blocks으로 구성되어 있다[1]. Global Array의 트리구조 예시는 다음과 같다.

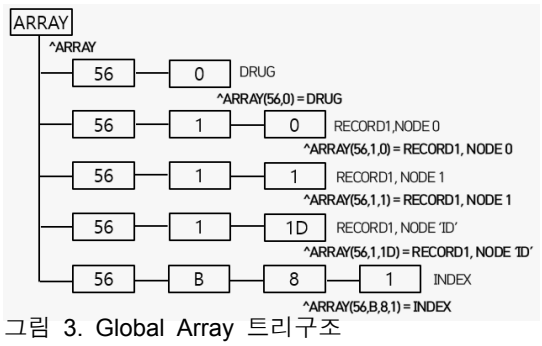

그림 3. Global Array 트리구조

Ⅳ. Database

다음은 Global Array를 이용하여 데이터를 저장하고 출력하는 간단한 프로그램이다. 추가, 수정, 삭제, 검색, 출력, 나가기 옵션을 구현하였으며, 프로그램 내부에서 데이터를 추가하고 출력한 후, 프로그램 외부에서 Global Array인 ^INFO에 저장된 정보를 출력해보고자 한다. 추가기능을 사용하여 정보를 입력하는 것은 다음과 같다.

```
1. ADD 2. MODIFY 3. REMOVE 4. SEARCH 5. PRINT 6. QUIT
ENTER TO OPTION: 1

NAME: JIHYEON
BIRTH: 950914
PHONE:
```

그림 4. 정보 저장 프로그램 추가 기능

그림과 같이 JIHYEON의 정보를 입력하였다. 이에 해당하는 기능의 루틴은 다음과 같다.

```
EXAM01ADD

 S ^INDEX=^INDEX+1
 R !,"NAME: ",NAME,!
 R "BIRTH: ",BIRTH,!
 R "PHONE: ",PHONE,!
 S STR=NAME_"^"_BIRTH_"^"_PHONE_"^1"
 S ^INFO(^INDEX,"NAME","BIRTH","PHONE")=STR
 W !,"NAME: "_$P(STR,"^",1)_" BIRTH: "_$P(STR,"^",2)_" PHONE: "_$P(STR,"^",3),!
 K STR
 Q
```

그림 5. 정보 저장 루틴

R은 READ를 뜻하며, C언어의 scanf와 같은 의미이다. 이름, 생년월일, 번호를 입력받아 ^로 구분하여 변수에 저장하고, SET ^INFO를 통해 데이터를 저장한다. 이것이 프로그래밍 언어와 데이터베이스를 연결하고, INSERT INTO를 통해

데이터베이스에 데이터를 입력하는 SQL문과의 차이점이다. 이 후에 3명의 정보를 입력한 후의 출력 기능은 다음과 같다.

```
1. ADD 2. MODIFY 3. REMOVE 4. SEARCH 5. PRINT 6. QUIT
ENTER TO OPTION: 5
^INFO(1,"NAME","BIRTH","PHONE")="JIHYEON^950914^010123156"
^INFO(2,"NAME","BIRTH","PHONE")="KANGHYEON^9507123^010123456789"
^INFO(3,"NAME","BIRTH","PHONE")="DONGHYEOP^920101^010123456"
^INFO(4,"NAME","BIRTH","PHONE")="DONGGYU^930505^0101234856"
```

그림 6. 정보 저장 프로그램 내부 출력 기능

마지막으로, 프로그램 종료 후 프로그램 외부에서 Global array에 저장된 내용은 다음과 같다.

```
GTM>ZWRITE ^INFO
^INFO(1,"NAME","BIRTH","PHONE")="JIHYEON^950914^010123156^0"
^INFO(2,"NAME","BIRTH","PHONE")="KANGHYEON^9507123^010123456789^1"
^INFO(3,"NAME","BIRTH","PHONE")="DONGHYEOP^920101^010123456^1"
^INFO(4,"NAME","BIRTH","PHONE")="DONGGYU^930505^0101234856^0"
```

그림 7. 정보 저장 프로그램 외부 출력

값의 마지막인 1은 후에 데이터를 삭제했을 시, 복구를 위한 값이며 프로그램 내부에서는 출력 하지 않는다. ^는 보통 값을 구분하기 위한 구분자로 쓰이고 다른 문자로도 구분할 수 있다. Global Array의 다른 저장 방법은, ^INFO(1,"JIHYEON","950914","01063294360")=1 과 같이 사용 가능하며, 이러한 저장 방법이 'Index 자체가 데이터 자체인 경우'이다.

Ⅴ. 활용

오늘날 MUMPS는 금융 및 의료계에 광범위하게 사용되고 있다. 미국의 가장 큰 온라인 거래 서비스인 Bank of England, Barclays Bank에서 사용된다. 미국의 대부분 모든 VA 병원 시스템은 데이터 추적을 위해 MUMPS 데이터베이스를 기반으로 사용하며, 의료계에서 가장 대표적으로 미국 VA의 전산 의료 기록 시스템인 VistA가 있다. 이는 미국에서 가장 익숙한 EHR이다. 이 외에 Epic, MEDITECH, GE Healthcare등 많은 의료 소프트웨어 IT 회사가 존재한다. VistA는 MUMPS와 MUMPS 데이터베이스로 개발되었으며, VistA에 내장되어 있는 메뉴를 통해 환자 정보 및 기록을 저장하고, 트리 구조로 데이터를 출력할 수 있다. 현재 실제 업무에서 VistA를 서버로 사용하고 있고, 이를 활용하여 환자의 정보와 기록을 저장한다. 다음은 VEHU VistA 메뉴에 환자 정보를 저장하는 그림이다.

```
Select OPTION: 4  MODIFY FILE ATTRIBUTES
Do you want to use the screen-mode version? YES//

Modify what File: 서강현
  Are you adding '서강현' as a new FILE? No// y  (Yes)
   FILE NUMBER: 999001// 8714
```

그림 8. 데이터 저장 과정

그림과 같이, 정보를 MUMPS로 이루어진 데이터베이스에 저장하는 데 사용되며 다음 그림과 같이 PATTERN MATCH를 이용하여 데이터 저장 시 자동으로 입력 데이터의 범위를 지정한다.

```
                    Field #.01 in File #8714
FIELD LABEL: NAME                          DATA TYPE... FREE TEXT

            MINIMUM LENGTH: 3
            MAXIMUM LENGTH: 30
AUDIT C   PATTERN MATCH (IN 'X'): X'?1P.E
```

그림 9. 저장된 데이터 수정

마지막으로 저장되어 있는 MUMPS Global Array로 이루어진 데이터이다.

```
^VA(20,"C","ZZZRETTWOTHIRTYSEVEN,PATIENT",6273)=""
^VA(20,"C","ZZZRETTWOTHIRTYSIX,PATIENT",3716)=""
^VA(20,"C","ZZZRETTWOTHIRTYSIX,PATIENT",6364)=""
^VA(20,"C","ZZZRETTWOTHIRTYTHREE,PATIENT",3645)=""
^VA(20,"C","ZZZRETTWOTHIRTYTHREE,PATIENT",6299)=""
^VA(20,"C","ZZZRETTWOTHIRTYTWO,PATIENT",3833)=""
^VA(20,"C","ZZZRETTWOTHIRTYTWO,PATIENT",6471)=""
^VA(20,"C","ZZZRETTWOTHREE,PATIENT",3589)=""
^VA(20,"C","ZZZRETTWOTHREE,PATIENT",6245)=""
^VA(20,"C","ZZZRETTWOTWELVE,PATIENT",3617)=""
^VA(20,"C","ZZZRETTWOTWELVE,PATIENT",6272)=""
^VA(20,"C","ZZZRETTWOTWENTY,PATIENT",3788)=""
^VA(20,"C","ZZZRETTWOTWENTY,PATIENT",6429)=""
^VA(20,"C","ZZZRETTWOTWENTYEIGHT,PATIENT",3770)=""
^VA(20,"C","ZZZRETTWOTWENTYEIGHT,PATIENT",6411)=""
^VA(20,"C","ZZZRETTWOTWENTYFIVE,PATIENT",3605)=""
^VA(20,"C","ZZZRETTWOTWENTYFIVE,PATIENT",6261)=""
^VA(20,"C","ZZZRETTWOTWENTYFOUR,PATIENT",3697)=""
^VA(20,"C","ZZZRETTWOTWENTYFOUR,PATIENT",6348)=""
^VA(20,"C","ZZZRETTWOTWENTYNINE,PATIENT",3793)=""
^VA(20,"C","ZZZRETTWOTWENTYNINE,PATIENT",6434)=""
^VA(20,"C","ZZZRETTWOTWENTYONE,PATIENT",3679)=""
^VA(20,"C","ZZZRETTWOTWENTYONE,PATIENT",6330)=""
^VA(20,"C","ZZZRETTWOTWENTYSEVEN,PATIENT",3583)=""
^VA(20,"C","ZZZRETTWOTWENTYSEVEN,PATIENT",6239)=""
^VA(20,"C","ZZZRETTWOTWENTYSIX,PATIENT",3646)=""
^VA(20,"C","ZZZRETTWOTWENTYSIX,PATIENT",6300)=""
^VA(20,"C","ZZZRETTWOTWENTYTHREE,PATIENT",3602)=""
^VA(20,"C","ZZZRETTWOTWENTYTHREE,PATIENT",6258)=""
^VA(20,"C","ZZZRETTWOTWENTYTWO,PATIENT",3661)=""
^VA(20,"C","ZZZRETTWOTWENTYTWO,PATIENT",6312)=""
^VA(20,"C","ZZZRETTWOTWO,PATIENT",3733)=""
^VA(20,"C","ZZZRETTWOTWO,PATIENT",6381)=""
^VA(20,"C","서,강현",8714)=""
```

그림 10. VistA 데이터

Ⅵ. 결론

MUMPS는 일반적인 관계형 데이터베이스처럼 SQL과 다른 프로그래밍 언어가 함께 쓰이는 것과 달리 그 자체로 프로그래밍 언어와 데이터베이스를 지원한다. MUMPS는 과거 부족한 저장용량으로 인해 축약된 명령어를 사용하고, 사칙/논리 연산 우선순위를 따르지 않는다는 특징이 있으며, 오라클, MySQL과 같은 관계형 데이터베이스와 달리 SQL문을 사용하지 않고 데이터를 조작할 수 있다는 큰 차이점이 있다. NoSQL은 데이터 처리량이 높고, 지연시간이 낮다. SQL을 통해 데이터 저장 및 검색을 해야 하는 관계형 데이터베이스와 달리, 객체기반 API를 통해 데이터 저장 및 검색이 가능하다. 끝으로, 실제 미국에서 쓰이고 있는 MUMPS 기반의 플랫폼과 활용방안에 대해 연구가 필요할 것으로 본다.

References

[1] H. E. Shim. "Library Automation Using MUMPS", Taegu, Korea, pp. 1-3, 10-11, 1991.