특집기사

데이타베이스 시스템의 기술 동향

허 대 영† 김 창 석† 임 기 욱† 오 길 록†

❖ 목 차 ❖

1. 머리말

최근 들어 국내 데이타베이스 시스템 시장의 규모는 폭발적으로 증가하고 있다. 1992년에 약 197억원 정도의 시장을 나타내더니 93년에 들어서는 더욱 가속도가 붙어 약 340억원 정도 시장으로 확대될 것으로 예측되고 있다[13]. 국내 DBMS(DataBase Management System) 시장이 외국에 비하여 앞으로 더 급신장할 것이 예상되는 이유는 현재 국내 대부분의 정보는 데이타베이스화가 되어 있지 못한 상황에서 데이타베이스 산업이 태동기와 정보화 사회를 맞고 있다는 점이다. 세계 시장도 DBMS의 호황은 예외가 아니어서 UNIX 전문 잡지인 Unix/World 에 따르면 1992년도 UNIX 시장에서 최대 매출액을 기록한 소프트웨어 업체들 중 1위부터 4위까지는 데이타베이스시스템 전문회사 (Oracle, Informix, Sybase, Ingres)가 독차지하고 있음을 나타내고 있다. 이러한 때에 국내 DBMS 시장의 현황을 보면 Oracle을 비롯한 외국 유명사의 상품이 거의 전 시장을 점유하고 있다. 이러한 상황은 국제 수지면에서 뿐만이 아니라, 국내 데이타베이스 산업 기술의 해외 기술 의존도를 심화시키는 결과를 초래하고 있다.

DBMS 기술의 중요성은 모든 컴퓨터 시스템이 사용되는 곳의 약 80%는 컴퓨터를 사용하기 위한 것이라기 보다 데이타베이스를 사용하기 위해서 컴퓨터를 사용한다는 통계와 DBMS 가격이 시스템 가격을 상회한다는 사실에서 알 수 있다.

근래에 들어 국내 연구소 및 기업에서 이러한 중요성을 인식하고 바다를 비롯한 국산 DBMS의 개발 및 상품화에 나서고 있다. 그러나 개발력의 미숙 및 경험 부족으로 해외 유명 제품과의 어려운 경쟁을 하고있다.

본 고에서는 데이타베이스 시스템 관련 기술 중에서 가장 중요한 데이타베이스 관리 시스템에 대한 최근의 기술 동향과 데이타베이스시스템 언어(SQL)의 표준화 동향에 대하여 살펴보고자 한다.

2장에서는 데이타베이스 관리 시스템을 형태 혹은 용도에 따라 분류 한 후, 각각에 대한 기술 설명과 현안 문제를 다루고 3장에서는 데이타베이스 표준언어인 SQL의 최근동향을 객체지향 기술 관점에 대해 살펴 보도록 한다.

2. 데이타베이스 시스템의 기술 동향

최근에 컴퓨터 전분야에 광범위한 영향력을 행사하고 있는 객체지향 기술, 실시간 기술, 병렬처리 기술 등이 DBMS 분야에 어떻게 영향을 주고 있으며 이에 대한 DBMS 분야의 반응은 어떻게 나

† 정 회 원 : 한국전자통신 연구소

타나고 있는가를 살펴본다.

2.1 병렬처리 컴퓨터(MPP:Massively Parallel Processor)용 데이타베이스 시스템

2.1.1 MPP용 데이타베이스의 출현 배경

MPP용 데이타베이스 시스템의 출현 배경은 다음과 같은 3가지 관점에서 찾아볼 수 있다.

가) CPU의 가격 하락

프로세서 기술의 발전과 가격 하락, 하드웨어 기술의 발전으로 예전에는 아주 값비싼 자원이었던 CPU와 주기억 장치가 이제는 쉽게 사용할 수 있는 자원이 되었다. 기존의 컴퓨터 설계 개념은 가능한 비싼 자원인 CPU와 주기억 장치를 여러 프로세스에게 할애하는데 주안점을 두었으나 현재 추세는 저렴한 CPU, 메모리 자원을 가능한 많이 사용하여 가격 대 성능의 비율을 올리고자 하는 시도와 내고장형 시스템 및 가용성(availability) 높은 시스템 개발을 추구하게 되었다

나) 다중처리 시스템(Multiprocessors System)의 한계

현재 데이타베이스 서버로서 가장 각광을 받고 있는 시스템으로는 여러개의 CPU와 하나의 버스, 그리고 공유메모리(TICOM, Sequent, ENCORE 등)를 갖고 있는 다중처리 시스템이라고 할 수 있다. 이러한 구조를 갖는 시스템은 현재 최고 32개 정도의 CPU를 탑재할 수 있고 더 이상의 성능 향상을 기대하기는 어려운 실정이다.

이에 반하여 MPP는 일반적으로 각 CPU당 독립적인 메모리를 갖고있으며 성능 향상을 위해 수천개 CPU까지 확장이 가능하다. 이러한 점은 다른 시스템에 비하여 MPP용 DBMS에게 유리한 환경을 제공할 가능성을 갖고 있다.

다) DBMS의 병렬성

앞에서 언급한 배경들은 다른 소프트웨어에도 영향을 미치지만 특히 DBMS 분야에 미치는 영향이 크다. DBMS는 독립적인 트랜잭션의 동시 지원. 질의어 처리에 있어 각 처리 단계가 정형화되어 있다. 각 처리 단계는 전 단계의 처리 결과를 입력으로 사용하는 파이프라인 형태의 수행 과정이 많고 정렬, 병합, 조인 등 관계 연산 자체도 병렬 처리에 적합한 것이 많다. 또한, 자료의 분산 저장으로 I/O의 병렬화도 가능하다[7].

2.1.2 현황과 전망

현재 상용화되고 있는 병렬 DBMS는 대량의 트랜잭션을 고속처리하는 OLTP용 병렬 DBMS와 방대한 자료에 대한 고속 질의 처리가 주요 목적인 것이 있다.

고속 질의 처리용 병렬 DBMS 분야는 기존의 데이타베이스 Machine 연구분야와 밀접한 연관성이 있으며, OLTP용 병렬 DBMS 이전에 앞서 연구가 되어 왔던 분야이다. 이 시스템은 주로 통계 및 의사결정 지원 도구등 방대한 자료에 대한 고속 질의 처리가 요구되는데 이용된다. 이러한 시스템으로는 병렬처리 컴퓨터를 이용한 최초의 상용 데이타베이스시스템인 Teradata(Teradata Corp.), 80386 프로세서를 1024개까지 채용할 수 있는 DBC/1012 (DataBase Computer 10**12 Data Storage), Gamma,Super Database Computer, Bubba 등이 있다. 이에 반하여 OLTP용 병렬 DBMS 분야의 연구는 고속 질의 처리 분야 보다 늦게 연구되기 시작한 분야로 OLTP 시장을 대상으로 하고 있다. 이러한 시스템으로는 Oracle Ver. 7.0, Sybase, Tandem NonStop SQL 등을 꼽을 수 있다[1][10].

현재는 MPP의 사용환경은 일반 시스템 개발자가 프로그래밍하기에 난해하다. 그러므로 MPP용 DBMS를 개발하기 위하여는 DBMS 개발자와 MPP 개발자들간의 공동연구가 요구되고 있다. 즉, 2가지 기술을 자체 확보하지 않으면 현실적으로 MPP용 DBMS 개발은 어렵다.

그러나, MPP가 범용 시스템으로 자리를 잡아가기 위해서는 필연적으로 일반 사용자가 쉽게 사용할 수 있는 병렬 프로그래밍 환경을 제공하게 될 것이고, 더욱 많은 MPP용 DBMS의 개발이 촉진

될 것이다. 또한 용도별로 다른 발전 역사를 갖고 있는 병렬 DBMS들도 통합되어 한 병렬 DBMS로 고속질의 및 고속 OLTP 지원이 가능해질 것으로 예상된다.

2.2 실시간 데이타베이스 시스템

2.21 실시간 시스템의 개념과 특성

최근 모 피자 회사에서는 주문한 피자를 30분이내 배달하면 피자 값을 모두 받고, 50분이내에 배달하면 2,000원을 할인해 주며, 50분 이상이 걸리면 무료로 제공해 주는 제도를 만들어 화제가 된 일이 있다. 이 피자 회사의 배달 방법은 마감 시간(deadline)이 30분인 소프트(soft) 실시간 시스템이라고 할 수 있다. 즉, 마감 시간내에 일을 끝내야만 의미가 있다. 통신 분야와 교통의 발달, 시간 관념에 대한 새로운 인식으로 컴퓨터 시스템을 이용하는 특정 분야 뿐만 아니라 우리의 실생활에서도 실시간에 대한 필요성이 현실화 되고 있다.

실시간 시스템이란 시스템의 연산 정확성이 계산의 논리적 결과에만 의존하지 않고 결과가 도출된 시간에도 의존하는 시스템을 말한다. 특히 실시간 시스템이 사용하는 데이타의 양이 증가함에 따라 데이타의 효율적인 저장과 조작이 필요하여 기존 데이타베이스 시스템과 결합하게 되었다. 이를 실시간 데이타베이스 시스템(Real-Time Data-Base System)이라 한다.

다음 표는 데이타베이스 시스템, 실시간 데이타베이스 시스템의 특성을 비교한 것이다.

<표 1>

기준 / 시스템	데이타 무결성 보장방법	트랜잭션	성능 목표
데이타베이스 시스템	시스템 의존	요구자원의 불예측성 시스템정책에 따른 수행순서	최대의 TPS
실시간 데이타베이스시스템	사용자와 시스템 의존	요구자원의 예측성 우선순위에 따른 수행순서	마감시간을 준수한 최대의 TPS

실시간 시스템은 넓은 영역에 걸쳐서 매우 중요하게 응용되고 있다. 응용 분야로는 우주 항공 산업, 국방 시스템, 원자력 플랜트, 교통제어 시스템, 로보트 제어, 주식 시장 등 광범위하다. 최근들어 하드웨어와 인공지능 및 멀티미디어 기법의 발달로 이들과 직접적인 결합이 필요한 실시간 시스템에 관한 관심이 고조되고 있다. 특히 시간적 제약을 다루고 있기 때문에 시스템 고장이 경제적으로나 사회적으로 또는 생태학적으로 큰 재앙을 초래할 수 있어 이에 대한 지속적인 연구와 투자가 요구되고 있다. 그 동안 실시간 시스템이 학계나 산업계에서 관심을 받지 못한 이유중의 하나가 잘못된 인식에 기인한 것도 있다. 다음은 실시간 시스템의 잘못된 인식을 살펴보고 특성을 정리한다[11].

첫째, 실시간 처리는 빠른 계산(fast computing)을 의미하는 것은 아니다. 빠른 계산이란 주어진 작업을 수행하는데 평균 응답시간을 단축하려는 노력이지만, 실시간 처리란 각각의 작업에 연관된 시간 제약(timing constraints)을 만족하려는 노력이다. 실시간 시스템은 빠른 계산도 필요하지만 예견성(predictability)이 더 중요하므로 작업의 요구 사항이 시스템의 기능적 및 시간적 행위에 만족하는가를 예견할 수 있어야 한다.

둘째, 슈퍼 컴퓨터, 고성능 컴퓨터 시스템의 발달이 실시간 처리 요구사항을 만족한다는 것은 잘못된 인식이다. 슈퍼 컴퓨터, 고성능 컴퓨터는 다수개의 프로세서를 채용하여 병렬 처리 기법의 사용으로 생산성은 향상시키지만 자동적으로 시간제약을 만족시키지는 못한다.

셋째, 실시간 시스템 설계는 체계적인 방법은 없고 특별한(ad hoc) 방법으로만 가능하다는 것은 잘못된 인식이다. 현재까지는 주로 이런 방법으로 시스템을 구성했지만 과학적인 접근 방법과 체계적인 설계가 요구되는 분야가 많아지고 있다.

실시간 처리 시스템의 특성을 정리하면 다음과 같다.

- 시스템이 관리할 자원으로 시간제약이 추가된다.
- 고성능 계산 능력이 요구된다.
- 복잡하고 대규모이며 복잡성이 많다.
- 예견성이 있어야 한다.
- 신뢰성과 fault tolerant 해야한다.
- 동적환경에서 작동해야 한다.

2.2.1 실시간 데이타베이스 시스템

실시간 데이타베이스 시스템은 트랜잭션이 마감시간이라는 시간제약을 가져 계산된 결과가 마감시간내에 도출되어야 의미가 있다. 기존의 데이타베이스 시스템의 주요 쟁점은 데이타의 무결성 유지, 데이타 공유, 비순차적 질의어 사용, 편리한 사용자 인터페이스 제공 등이었다. 기존의 데이타베이스 시스템은 트랜잭션들의 수행 순서는 데이타의 일치성이 보장되고 최대의 TPS(Transactions Per Second)를 달성하고자 하는 관점에서 데이타베이스 시스템이 자의적으로 배정한다. 데이타의 대부분이 디스크에 상주하므로 트랜잭션 시간의 대부분을 디스크 접근에 할애하여 비교적 낮은 성능을 보인다. 이런 낮은 성능과 예견성의 부재로 기존의 데이타베이스 시스템은 높은 성능과 예견성을 요구하는 실시간 응용에 부적합하다.

실시간 데이타베이스 시스템은 데이타의 일치성을 유지하면서 마감 시간을 만족하도록 트랜잭션 스케쥴링을 하는데 있다. 그러므로 실시간 트랜잭션 처리에서는 마감시간 및 중요성(criticality)에 기반한 우선순위가 스케쥴링에 반영되어야 한다. 즉, 트랜잭션 상호간에는 상대적인 우선순위가 부여되어야 하고, 낮은 우선순위의 트랜잭션은 높은 우선순위의 트랜잭션을 기다리게 하여 높은 우선순위의 트랜잭션이 마감 시간을 지키도록 한다. 실시간 데이타베이스 시스템에서는 데이타 일치성, 트랜잭션 정확성, 마감시간에 의거하여 동시성과 자원효율을 극대화하는 실시간 스케쥴링 및 동시성 제어 기법을 개발하는 것이 관건이다.

이러한 실시간 데이타베이스 시스템의 트랜잭션은 크게 소프트 트랜잭션과 하드 트랜잭션으로 나뉜다. 소프트 트랜잭션은 마감 시간을 못지켜도 수행 결과가 어느 정도 가치가 있는 것을 말하며, 하드 트랜잭션은 반드시 시간 제약을 지켜야 하며 못지킬 경우 중대한 손실을 가져오는 것을 말한다. 예를 들어 대공 미사일을 발사하는 국방 시스템이나 원자력 플랜트 등에 응용되는 것이 여기에 속한다.

2.2.3 실시간 데이타베이스 접근 방법

실시간 데이타베이스를 구현하는 방법은 두 가지가 있다. 첫째는 데이타베이스의 낮은 성능의 주된 원인인 디스크를 빠른 매체인 주기억 장치로 대체하는 것으로 주기억 상주 데이타베이스 시스템(Main-Memory Database System)이 여기에 속한다. 둘째는 기존의 데이타베이스 시스템의 기능을 실시간 시스템에 적합하도록 변경하는 것이다.

여기서는 두번째 방법에 대하여만 논의하고 첫번째 방법에 대하여는 나중에 별도로 다루기로 한다.

기존 데이타베이스 시스템의 실시간화는 일반적으로 실시간 데이타베이스 시스템은 전통적인 데이타베이스 시스템에 비하여 빠른 트랜잭션 처리를 요구하며 트랜잭션 처리시간에 대한 예측성을 요구한다. 현재의 기술로는 기존 데이타베이스 시스템과 같은 범용성을 갖는 실시간 데이타베이스 시스템을 개발하는 것은 비현실적이다. 그러므로 실시간 데이타베이스 시스템에서는 기존 데이타베이스 시스템은 자신이 처리할 트랜잭션의 성격에 대하여 아무런 지식이 없다는 점을 가정하고 있는데 반하여 특정 데이타베이스에 요구되는 모든 트랜잭션의 성격을 미리 파악할 수 있다는 가정과 사용자가 어느정도의 불편성을 감수할 수 있다는 가정에서 출발한다.

이러한 가정은 기존 데이타베이스 시스템에서 편리한 사용자 인터페이스 기능을 축소하여 직접 실시간 응용시스템과 연결하여 성능을 향상할 수 있는 여지, 트랜잭션의 직렬성(serializability)의 자동적이고 엄격한 보장에 대한 완화를 통한 성능향

상과 동시성제어 기법에 의하여 자의적인 기다림에 의한 트랜잭션 마감시간에 대한 불확실성을 감소시킬 수 있는 여지를 제공한다. 또한 모든 트랜잭션의 성격을 미리 알 수 있으므로 이들 간의 스케쥴링을 좀더 정확히 수행할 수 있다.

기존 데이타베이스 시스템을 실시간 환경에 맞게 변경하기 위하여 우선 트랜잭션 관리 기법에 대한 보강과 트랜잭션 직렬성의 완화로 인하여 발생될 수 있는 무결성 데이타베이스의 손상에 대한 대처를 위한 회복 기법 기존 동시성 제어방법에 대한 변경들이 주요 연구대상으로 떠오르고 있다.

2.2.4 향후 발전 추세와 전망

실시간 데이타베이스 시스템은 공장 자동화, 군장비 시스템, 항공 우주 산업, 교통 제어 시스템 등에 사용되었으나 앞으로는 주식 거래, 국제 금융 거래 등의 인문 사회 분야에서도 활발히 이용될 것으로 보인다. 그러나 실세계에서 널리 사용되는 실시간 데이타베이스 시스템 설계를 위해서는 실시간 처리에 대한 과학적인 연구와 운영체제, 분산 시스템, 고장 감뢰 시스템 등 연관 있는 분야와도 공동 노력이 필요하다.

2.3 주기억 상주 데이타베이스 시스템

2.3.1 출현 배경과 연구 동향

일반적인 데이타베이스 시스템에서는 데이타를 디스크에 저장해야 하므로 데이타 입출력시 디스크 접근을 해야 한다. 디스크 접근은 대략 18-25ms의 접근 시간이 소요되므로 빈번한 디스크 접근은 고성능 데이타베이스 시스템을 구현할 때 큰 장애요인이다. 또한 온라인 트랜잭션 처리나 실시간 데이타베이스 처리와 같이 고속의 자료 처리가 필요한 응용 분야가 생겼다. 디스크 접근 시간을 감소시켜 데이타베이스 연산을 빠르게 하는 가장 보편적인 방법으로는 전체 데이타 혹은 대부분의 데이타를 주기억 장치에 상주시켜 수행하는 방법이 있다. 고가의 자원인 주기억 장치에 데이타를 상주시키는 방법은 과거에는 경제적인 타당성 문제로 현실성이 없었다. 그러나 반도체 기술의 급속한 발달은 컴퓨터 메인 메모리 소자인 RAM의 가격을 하락시켰으며 칩의 집적도도 해마다 두 배씩 높아가고 있다. 현재 개인용 컴퓨터도 수 백 메가바이트 메모리를 가지며, 수 년내에 수 기가 바이트 메인 메모리를 가질 것으로 전망된다. 이러한 메모리 소자의 기술적 추세와 고성능 자료 처리의 필요성 대두로 모든 자료를 메모리에 상주시켜 디스크와의 입출력 병목 현상을 없앤 메인 메모리 데이타베이스 시스템의 경제적 타당성이 생기게 되었다.

지난 수 년간 메인 메모리 데이타베이스(MMDB : Main Memory Database)에 관한 연구가 진행되어 왔다. MARS, MMM, MM-DBMS, PRISMA와 같은 여러 연구 프로젝트가 진행되어 왔으며, SiDBM, MACH와 같은 시제품도 제작되었다. 또한 IMS/VS Fast Path, OBE와 같이 상용으로 사용되는 것도 있다. 다음 (그림 1)은 주기억 상주 시스템들을 특성별로 분류하여 비교한 것이다[3].

MMDB에 대한 개념은 크게 다음 두 가지로 분류할 수 있다[4].

① 모든 데이타베이스가 메인 메모리에 저장되어 있어서 데이타에 접근하기 위한 I/O가 필요없다. 이 경우는 메인 메모리 용량이 모든 데이타를 저장할 만큼 충분해야 한다.

② 데이타베이스는 디스크에 상주하나 큰 버퍼나 캐쉬가 존재하여 I/O 횟수는 감소한다.

MMDB에 대한 초창기의 연구방향은 대부분 ② 번의 큰 버퍼 풀을 어떻게 효율적으로 사용하느냐에 촛점을 맞추었다. 이 방법은 기존의 DBMS를 재 디자인할 필요없이 약간의 변형으로 디스크 I/O 횟수를 줄여 성능 향상을 기대할 수 있으나, 데이타베이스에 접근할 때 마다 매번 버퍼 관리기(buffer manager)를 통해 원하는 페이지가 메모리에 존재하는지 조사하여, 존재하면 주소 계산 작업을 한 후 해당 페이지를 고정하여 swapping 되지 않게 한다.

대부분의 MMDB 연구 프로젝트는 ①번의 정의를 가정하고 진행한 것이 많다. 이 접근 방법은 데

	Concurrency	Committ Processing	Data Representation	Access Methods	Query Processing	Recovery
MM-DBMS	two-phase locking of relations	stable log tail by segment	self-contained segment heap per segment extensive pointer use	hashing, T-trees, pointers to values	merge, nested-loop, joins	
MARS	two-phase locking of relations	stable shadow memory log tail				
HALO		in hardware nearly transparent				
OBE			extensive use of pointers	inverted indexs	nested loop-join, on-the-fly-index creation, optimization focused on processor costs	
TPK	serial transaction execution	group committ precommitt	arrays			two memory resident databases, fuzzy checkpoints
System M	two-phase locking minimize concurrency	several alternatives	self-contained segment, heap per segment			various checkpointing, logging options
Fast Path	VERIFY/CHANGE for hot spols	group committ				

(그림 1) 주기억 상주 시스템

이타에 접근하기 위한 I/O는 필요없어 성능은 우수하나, 기존 데이타베이스 관리 시스템과는 자료구조, 동시성 제어, 색인 구조, 회복 기법등이 다름으로 기존 DBMS를 대폭 수정하거나 재 설계해야 하는 단점이 있다. 여기서는 ①의 접근 방법을 위주로 설명한다.

2.3.2 연구 분야

MMDB는 기존의 데이타베이스 시스템과 비교할 때 몇가지 문제점을 가진다. MMDB의 가장 큰 문제점으로는 휘발성인 주기억 장치는 시스템이 파손되었을 때 기억된 내용을 모두 잃어버리므로 전체 데이타베이스가 파괴되는 결과를 초래한다. 그래서 MMDB에서도 안정된 기억장치로 디스크가 필요하다.

일반적으로 MMDB의 회복 방법은 시스템이 파손되었을 때 파손된 전체 데이타베이스는 디스크에서, 현재 갱신 중인 데이타는 로그 버퍼에서 가져와서 둘을 합하여 회복한다. 일반적인 데이타베이스 시스템에서는 데이타의 구성 접근 방법 질의어 처리를 위한 알고리즘이 CPU 사용시간보다 디스크 접근 시간을 단축하는 방향으로 설계되어 왔다. 이는 일반적인 데이타베이스 시스템은 평소 소량의 데이타를 주기억 장치 버퍼에 유지하기 때문이다. 기존 데이타베이스 시스템에 적합하게 고안된 알고리즘은 이런 환경적인 특성 때문에 주기억 상주 데이타베이스 시스템에는 비효율적이라는 시험 결과가 나와 있다. 또한 주기억 상주 데이타베이스 시스템은 한정된 주기억 장치에 대량의 데이타를 유지해야 하므로 기억 공간 사용 효율성이 문제가 된다. 즉, 주기억 상주 데이타베이스에서는 처리 시간의 단축과 기억 공간의 효율성을 모두 고려하여 알고리즘이 고안되어야 한다.

주기억 상주 데이타베이스 시스템을 위한 다양한

자료 구조들이 제안되었다. Ammann은 중복되는 데이타를 한 곳에 저장하고 튜플을 포인터로 연결하여 기억 공간을 절약한 이중 연결 리스트(doubly linked list)를 제안했고, Lehman은 AVL 트리와 B+ 트리 구조를 결합한 T 트리를 제안하여 순서화된(ordered) 데이타베이스에서는 T 트리가 우수한 성능을 발휘하고 비순서화된(unordered) 데이타베이스에서는 modified linear hashing이 우수한 성능을 발휘함을 보였다[3].

주기억 상주 데이타베이스 시스템의 응용 분야는 트랜잭션이 빈번한 고성능의 컴퓨팅이 요구되는 분야이므로 동시성 제어에 관한 연구는 대단히 중요하다. 주기억 상주 데이타베이스 시스템에서는 디스크 접근 시간이 거의 없으므로 트랜잭션이 동시성 제어를 위한 록킹 시간이 비교적 짧으며, 결과적으로 블록킹에 의한 지연 시간이 짧게 된다. 그러므로 주기억 상주 데이타베이스 시스템에서는 록킹 단위를 크게할 수도 있다. 큰 록킹 단위는 충돌을 빈번히 발생시킬 수 있으나 전체적으로 보면 부대 비용이 감소된다. Lehman은 보통때는 테이블 수준의 록킹을 하다가 충돌이 증가하여 기준치를 초과하면 레코드 수준의 록킹으로 변환되는 2 단계 계층 록킹(two level hierarchical locking)을 제안했다. SiDBM과 OBE에서는 낙관적 동시성 제어 기법을 사용하고, PRISMA에서는 2 단계 록킹(two-phase locking)을 사용한다. 그러나 아직까지 어떤 동시성 제어 기법이 주기억 상주 데이타베이스 시스템에 가장 적합한지 체계적으로 분석이 부족한 상태이다.

2.4 객체지향 데이타베이스 시스템

본 절에서는 요즘 컴퓨터 분야 전반에 걸쳐 영향을 미치고 있는 객체지향 기술과 데이타베이스 시스템의 결합인 객체지향 데이타베이스 시스템의 등장 배경과 기존 관계형 데이타베이스와의 통합 혹은 상호 운영 기법에 대하여 설명한다. 그리고 객체지향 데이타베이스 시스템의 향후 전망에 대해서도 기술한다.

2.4.1 객체지향 데이타베이스의 등장 배경

객체지향 데이타베이스의 등장은 두가지 방향에서 이루어졌다고 할 수 있다. 첫째는 객체지향 언어(Simula, Smalltalk, C++)에다 지속성을 갖는 데이타(persistent data) 사용 능력을 추가하려는 시도로 부터의 접근이다. 여기서 지속성 데이타는 기존의 데이타베이스와 유사한 속성을 가지므로 기존의 데이타베이스 관리 시스템과 같은 기능이 필연적으로 요구된다. 둘째 접근방법은 첫번째 방법보다 후에 이루어진 것으로 기존의 DBMS 분야에서 객체지향 기술을 흡수하는 방향에서 출발하였다. 이러한 객체지향 데이타베이스 시스템은 객체지향 기술의 장점을 기존 DBMS 기술에 보강하므로서 현재 새로운 응용분야로 부상하고 있는 CAD/CAM, 다중매체 처리(multimedia), 탁상 출판, CASE, 인공 지능 응용 등 새로운 응용분야에 기존의 데이타베이스 관리 시스템보다 효율적으로 적응할 수 있는 능력 배양에 중점을 두고 있다.

2.4.2 객체지향 데이타베이스 시스템의 주요 특성

객체지향 데이타베이스 시스템은 데이타베이스 시스템과 객체지향 기술이 집목된 시스템이다. 그러므로 이러한 시스템들이 제공해야하는 기능은 DBMS 고유 기능인 동시성 제어, 회복 기능, 질의 기능, 접근 방법의 최적화 기능, 와부 기억 장치 관리 기능 등은 물론 다음과 같은 객체지향 기술의 다섯가지의 기본 개념을 지원해야 한다[8].

① 실세계의 각 실체는 한 객체로 표현되어야 하며 각 객체는 고유의 식별자를 가져야 한다.

② 각 객체는 애트리뷰트들의 인스탄스들의 집합과 메소드로 구성된다. 또한 각 애트리뷰트의 값은 한 객체 또는 객체들의 집합이다.

③ 애트리뷰트의 값은 객체의 상태를 표현하며 이 값은 그 객체가 메세지를 받아 수행되는 메소드를 통하여 변화되어 질 수 있다.

④ 각 객체는 클래스의 인스턴스이다.

⑤ 클래스는 하나 이상의 클래스로부터 유도되어질 수 있다.

이러한 객체지향 개념이 어떻게 객체지향 데이타베이스 관리 시스템에 의하여 지원 되는가를 UniSQL을 통하여 살펴보면 다음과 같다.

UniSQL은 릴레이션을 클래스로, 튜플 또는 레코드를 클래스의 인스탄스로, 열을 애트리뷰트로, 프로시듀어를 메소드로, 릴레이션 간의 계층을 클레스의 계층으로 확장된 개념으로 제공한다. 물론 이러한 개념 확장은 UniSQL 이라는 객체지향 데이타베이스 관리 시스템을 통하여 인식될 수 있는 것이다[12].

이러한 객체지향 개념을 지원하는 객체지향 데이타베이스 시스템의 장점은 데이타 모델링이 용이하고, 객체지향 언어의 자료형 기능을 지원할 수 있고, 재사용성 및 질의 성능의 향상을 도모할 수 있다.

2.4.3 객체지향 데이타베이스 시스템과 관계형 데이타베이스 시스템

객체 지향 데이타베이스의 가장 큰 현안 문제는 이미 광범위하게 사용되고 있는 기존의 관계형 데이타베이스와 어떻게 협조하느냐는 문제가 될 것이다. 1980년대 중반 처음으로 객체 지향 데이타베이스 시스템이 발표되었을 때는 기존의 관계형 데이타베이스를 대체할 차세대 데이타베이스 기술로 기대 되었으나 이미 널리 사용되고 있는 관계형 데이타베이스 시스템에 대한 높은 현실감(데이타 관리 표준 언어인 SQL을 지원 안함)과 객체 지향 기술의 한계(관계형 데이타베이스가 지원하는 기능을 완전히 지원 못하고 있음)로 인해 현재는 관계형 데이타베이스과의 통합 또는 상호 운영 방법을 모색하고 있다. 이에 반하여 관계형 데이타베이스 분야에서는 기존의 SQL에 Abstract 자료형을 지원할 수 있게 SQL(ISO SQL III)을 확장하여 객체지향 기술을 흡수하여 자신의 기득권을 유지하려 하고 있다[5].

객체지향 기법이 데이타베이스 기술에 획기적인 도약을 가져올 수 있으나 기존의 관계형 데이타베이스와의 호환성을 유지해야 한다는 전문가들도 있다. 이러한 관점에서 객체지향 기술과 관계형 데이타베이스 시스템과의 통합시도는 통합된 정도에 따라 3가지로 분류될 수 있다. 첫째는 객체지향 개념에 따른 질의를 SQL로 변환하여 이를 관계형 데이타베이스 시스템에 전달하는 형태이고 둘째는, RDB 상층에 객체지향 계층을 두고 사용자는 객체지향 데이타베이스 언어를 사용할 수 있게 하지만, 이를 관계형 데이타베이스 시스템을 이용하여 지원한다. 셋째로 관계형 및 객체지향 자료 모델을 전 시스템에서 통일적으로 지원하는 방식을 말한다[9].

2.4.4 향후 발전 추세와 전망

1987년초부터 판매되기 시작한 객체지향 데이타베이스 시스템은 현재 GemStone (Servio Corp.), ONTOS(ONTOS Inc.), ObjectStore(Object Design), Objectivity/DB(Objectivity), Versant(Versant Object Tech.), Orion(Xidak), OpenODB(HP)등이 있다. 그러나 이들 대부분은 아직 시제품 수준이어서 중요한 응용에 사용된 적이 없다.

지난 5년 동안은 객체지향 데이타베이스의 태동기라고 할 수 있다. 앞으로 관계형 데이타베이스가 처리하기 힘든 CAD/CAM, 멀티미디어 처리 분야 등을 발판으로 시장을 구축해 나갈 것으로 예상되며 앞으로는 순수 객체지향 데이타베이스 보다 관계형과 객체지향을 혼합한 확장형 데이타베이스 시스템이 우세할 것이 전망된다.

3. 데이타베이스 언어(SQL) 표준화 동향

데이타베이스 시스템 기술 발전에 있어 가장 중요한 부분중에 하나가 데이타베이스 언어의 발전 동향이다. 그 이유로는 데이타베이스 언어의 변천이 결국 데이타베이스 시스템 발전 방향에 커다란 영향력을 행사하기 때문이다. 여기서는 SQL 표준화 과정과 현재 표준화가 진행중인 SQL3을 객체지향 기술 관점에서 살펴보기로 한다[5][6].

3.1 SQL 표준화

Codd에 의해 관계 데이타 모델이 제안된 후,

IBM사에서 System R이라는 관계 데이타베이스 관리 시스템(RDBMS: Relational Database Management System)을 만들었다. System R의 데이타베이스 언어는 SEQUEL이었으며, 이것이 지난 몇 년 사이 관계형 데이타 모델(Relational Database Model)에서 데이타베이스 언어인 SQL (Structured Query Language)로 국제 표준이 되었다. SQL에 대한 최초의 표준화 작업은 ANSI (American National Standard Institute)에 의해 수행되었고, ANSI에 의해 제안된 SQL이 ISO에 의해 1987년 국제 표준 ISO 9075-1987으로 채택되었다. 그러나 DBMS 제품이 점차 많아지고 응용 분야가 넓어짐에 따라 데이타베이스 언어의 기능을 확장하고자 하는 기운이 또한 커졌다. ISO와 그 밖의 SQL 표준화 작업에 관심이 있는 그룹에서는 ISO 9075-1987 표준에 기본적인 참조 무결성(referential integrity)에 대한 정의를 추가하여 ISO/IEC 9075-1989을 만들었고, 다른 한편으로는 SQL 표준의 후속판(즉, SQL2)을 제안하기 위한 작업을 해왔다. 현재 SQL2의 표준화 작업은 1991년 4월에 ISO/IEC 9075-1992의 DIS (Draft International Standard)판이 1992년에 확정되었다. 현재는 다음 세대의 데이타베이스 언어로 SQL3에 대한 작업이 진행되고 있다. SQL3는 SQL2의 기존 기능을 보안(참조 무결성 기능, 널값 관리, 커서와 뷰의 관리 기능 등을 개선), 객체 관리 기능 지원, 저장함수 지원, 분산 데이타 지원및 타 표준안과의 인터페이스 지원 방향으로 확장되고 있다.

3.2 객체지향 관점의 SQL3 기능

본 절에서는 SQL3의 표준화 진행 방향에서 핵심 분야인 객체 관리 부분의 확장에 대하여 논의하고자 한다.

3.2.1 ADT(Abstract Data Type)

SQL3에서는 객체지향 기술의 클래스에 대응되는 것으로 ADT를 제공한다. 새로 작성되는 ADT는 기존의 자료형태 또는 먼저 정의된 ADT로 부터 유도되어질 수 있다. 이것은 다음과 같은 형태를 갖는다.

```
CREATE TYPE <ADT name>
  [ < OID options > ]
  [ < subtype clause > ]
  [ < member list > ]
```

여기서 < member list > 는 ADT의 애트리뷰트, 연산, 메소드를 정의한다. 애트리뷰트 정의시에는 PUBLIC, PRIVARE, PROTECTED와 같은 encapsulation수준을 정의할 수 있다. PUBLIC인 경우는 해당 ADT에 접근 권한을 갖고있는 모든 사용자가 볼 수 있으며, PRIVATE 인 경우는 완전히 encapsulate 되며, PROTECTED 이면 정의된 ADT나 또는 그 ADT의 모든 subtypes들에게 보여진다. 연산은 메소드의 일종으로 인스탄스들의 순서 및 동일성을 정의하여 추후 SQL 비교 연산시 판단 기준으로 이용된다. 그외의 메소드는 FUNCTION으로 정의된다. 이 FUNCTION은 SQL문으로 정의된 SQL 함수와 일반적인 프로그램 언어(Ada, C, C++)로 정의된 외부 함수가 있다. 이러한 함수들은 연산과 달리 스키마의 한 요소로서 ADT 정의의 한부분으로서, 또는 모듈 정의시 부분으로 선언될 수 있다.

```
예) CREATE TYPE person
    WITH OID VISIBLE
    (name          char(30),
     address       char(100),
     PRIVATE birthday DATE,
     FUNCTION get—age(:p person)
     RETURNS REAL
     < age 계산 프로시듀어 >;
     END FUNCTION
    );
```

3.2.2 객체 식별자(Object Identifier)

객체 식별자(OID)는 객체에 대한 고유 식별자로 특정 객체를 객체값에 의존하지 않고 식별할 수

있게 한다. 기존의 관계형에서는 특정 레코드의 분별을 레코드 값만으로 인식할 수 있으나 객체 식별자는 객체 값과는 무관하다. 즉 특정 인스탄스(레코드)의 값이 변하여도 OID는 고유값을 유지된다.

OID는 다음과 같이 표현된다.

WITH OID VISIBLE,

WITH OID NOT VISIBLE, or

WITHOUT OID

WITH OID 가 정의되면 해당 ADT의 모든 인스탄스에게 OID가 부여되고 VISIBLE 이 정의되면 인스탄스를 OID를 이용하여 참조할수 있다. 또한 NOT VISIBLE이 정의되어 있으면 아는 OID를 함수의 파라미터나 또는 주언어 변수로 사용할 수 없다. *WITHOUT OID*가 정의된 경우는 인스탄스에 OID가 부여되지 않으며 관계형에서와 같이 인스탄스 값으로만 참조가 가능하다.

3.2.3 Subtypes과 계승(Inheritance)

subtype은 ADT를 정의할 때 <subtype clause> 인 "UNDER <ADT name>"을 사용하여 새로 정의한 ADT가 <ADT neme> (Supertype)으로 정의된 기존 ADT의 subtype임을 정의한다. 그러므로 한 subtype은 한개 이상의 supertype 의 상세화 (specialization) 이고 supertype 은 subtype 들의 일반화 (generalization) 관계가 성립된다. 이렇게 성립된 ADT 간에는 다중 계승(mutiple inheritance)이 발생한다.

예) CREATE TYPE student UNDER person(studentID integer);

3.3 SQL3 기타 확장기능

SQL3는 3.2에서 설명한 객체지향 관점의 확장뿐 아니라 제어구조를 위하여 SQL 언어에 ASSIGNMENT, CALL, RETURN 과 같은 제어문, 복합문을 지원하기 위한 블록(block), 예외 처리(exception handling)를 위한 문 및 제어흐름문(CASE, IF-THEN-ELSE, WHILE, LEAVE) 등이 첨가 되었다. 그밖에 parameterized type, 저장 프로시듀어 기능들이 확장되었다.

4. 맺음말

본 고에서는 데이타베이스 시스템 관련 기술을 DBMS 기술 동향과 데이타베이스 언어의 표준화 관점에서 살펴보았다. 이러한 동향을 살펴보면 데이타베이스 시스템이 객체지향 기술과 하드웨어 요소 기술(CPU, 메모리)에 대한 대응으로 생각된다.

실시간과 주기억 상주 데이타베이스 시스템에서 관심을 갖고 주시해야 할 하드웨어 관련 기술은 flash 메모리 분야일 것이다. 이 메모리는 현재 write 횟수(약 백만번 허용)에 대한 제약으로 일반 시스템의 메인 메모리로서 또는 가격이 너무 비싸서 주요 보조기억 장치로 활용되지 못하는 대신 기존 ROM 분야의 응용에 치우친 감이 있으나, 앞으로 write 횟수에 대한 제약이 없어지고 가격이 저렴해 지면 일반 시스템의 안전한 주기억 장치(safe memory)로 사용이 가능해 질 것으로 예상된다.

그러면 현재 실시간, 주기억 상주 데이타베이스 시스템의 회복 기능에 대한 큰 장애가 없어지게 될 것이고 현재 안전한 기억 장치로 디스크를 이용하고 있는 전통적인 데이타베이스 시스템에도 커다란 변혁이 요구될 것이다.

또다른 변화 요인은 대용량 주기억 장치를 갖는 시스템들의 등장이다. 현재와 같은 메모리 기술 발전 속도를 유지하면 멀지않아 퍼스컴 레벨의 시스템들도 기가 단위의 메모리를 갖게 될 것이다. 이러한 상황에서는 기존시스템이 현재와 같이 메모리를 단지 디스크 캐쉬용으로 사용하는 데는 한계가 있을것이고 주기억 상주 데이타베이스 시스템 기술의 상당 부분을 흡수하게 될 것으로 판단된다.

MPP용 데이타베이스 시스템 개발에 있어 가장 큰 장애는 병렬 프로그래밍 환경에 있다고 할 수 있다. 현재와 같이 MPP가 사용자에게 편리한 병렬 프로그래밍 환경을 제공하지 못 한다면 결국은

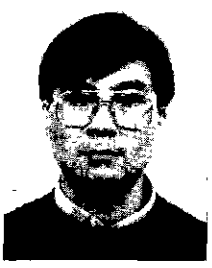
MPP 시스템에 대한 노하우에 접근할 수 있는 소수만이 MPP용 데이타베이스 시스템을 개발하게 될 것이고 개발된 MPP용 데이타베이스 시스템 또한 다른 시스템에 설치가 불가능하게 될 것이다. 이러한 상황은 일반 MPP 시스템이 나오기 전까지의 잠정적인 상태일 것으로 생각된다. 왜냐하면 오픈 시스템 기술 추세가 대세를 이루고 있으며, 병렬 프로그래밍을 환경을 제공할 수 있는 멀티쓰레드 프로그래밍 환경이 숙성되 가고 있기 때문이다 (Solaris 2.0, UNIX ES/MP.). 만일 MPP가 일반화 되지 못한다면 가격 상승을 초래해 현재 가격대 성능비에 대한 장점을 상실하고 특수용으로 전락될 것으로 생각된다.

객체지향 데이타베이스 시스템은 현재는 앞에서 설명한 2가지 접근방식의 상품이 혼재하고 있지만 결국은 데이타베이스 관리 시스템에 객체지향 기술을 확장하는 쪽이 승리할 것으로 예상된다. 왜냐하면, 기존의 데이타베이스 시스템의 거의 모든 시장을 기존 데이타베이스 시스템이 갖고 있으며 SQL3에서 보듯이 기존 관계형 데이타베이스 시스템에서 객체 지향 기술을 흡수하는 것이 어렵지가 않을것으로 판단되기 때문이다.

끝으로 이러한 기술들의 등장과 발전으로 현재보다 더욱 저렴한 비용으로 양질의 데이타베이스를 이용할 수 있기를 기대하고 특히, 국내 DBMS 개발자들의 분발을 기대한다.

참 고 문 헌

1. Dwight B. Dams, "Oracle's Parallel Punch for OLTP", Datamation, pp.67-74, August 1992.
2. A. Trew and G. Wilson, "Past, Present, Parallel: A Survey of Available Parallel Computer Systems", Springer-Verlag, pp. 225-231, 1991.
3. H. Garcia-Molina and K. Salem, "Main Memory Database Systems: An Overview ", IEEE Transactions on Knowledge and Data Engineering, Vol. 4, No. 6, Dec. 1992.
4. M. Eich, "Main Memory Database Research Direction", 6th International Workshop on Database Machine, pp. 251-268, 1989.
5. Leonard J. Gallagher, "Object SQL: Language Extentions for Object Data Management"
6. ISO/IEC 9075. Database Language SQL, International Standard ISO/IEC 9075: 1992
7. David Dewitt and Jim Gray, "Parallel Database Systems: The Future of High Performance Database Systems", COMMUNICATIONS of the ACM, June 1992,Vol 35, No. 6.
8. Elisa Bertino and Lorezo Martino, "Object-Oriented Database Management Systems: Concepts and Issues", IEEE Computer, April 1991, pp.33-47.
9. Rafiul Ahad and Tu-Ting Cheng, "HP OpenODB: An Object-Oriented Database Management System for Commercial Applications", Hewlett-Packard Journal, June 1993, pp.20-30.
10. Steve Hele, "Massively Parallel: Reaching Critical Mass", DATABASE & PROGRAMMING, June 1993, pp.46-50.
11. 이 상호, "실시간 데이타베이스의 개론," 정보과학회지, 제 11권, 제 1호, pp.7-12, 1993년 2월호.
12. 김 원, "객체지향 데이타베이스 기술," 정보과학회지, 제 11권, 제 2호, pp.42-58, 1993년 4월호.
13. 컴퓨터 월드, "지난해 국내시장 197억원 규모", 컴퓨터 월드, pp.168-175, 1993년 6월호.



허 대 영

1982년 숭실대학교 전자계산학과 졸업
1991 전자계산조직응용 기술사
1982년 - 현재 한국전자통신연구소 데이타베이스연구실장, 책임연구원 데이타베이스 서비스 시스템 개발 사업 책임자
관심분야 : 멀티미디어 데이타베이스, 실시간 데이타베이스

김 창 석

1983년 경북대학교 전자공학과(전산 전공) 졸업
1990년 경북대학교 대학원 전자공학과(전산 전공) 졸업(석사)
1993년 경북대학교 대학원 컴퓨터공학과 박사과정 수료
1992년 전자계산조직응용 기술사
1983년 ~ 현재 한국전자통신연구소 데이타베이스 연구실 선임연구원
관심 분야 : 멀티미디어 데이타베이스, 퍼지 및 자능형 데이타베이스

임 기 욱

1977년 인하대학교 전자공학과 졸업
1987년 2월 한양대학교 대학원 전자계산학(석사)
1994년 2월 인하대학교 대학원 박사과정 수료(전자계산전공)
1977년 2월 ~ 1985년 5월 한국전자기술연구소 선임연구원
1985년 6월 ~ 1988년 8월 한국전자통신연구소 시스템 S/W 연구실장
1988년 8월 ~ 1989년 8월 Univ.of California,Irvine 방문연구원
1989년 10월 ~ 현재 한국전자통신연구소 시스템연구부장, 책임연구원
관심분야 : 운영체제, DBMS, 시스템 S/W분야, 컴퓨터구조 연구

오 길 록

1968년 서울대학교 문리대 졸업(학사)
1975년 한국과학기술원 산업공학과 졸업(석사)
1981년 프랑스 INSA 전산학과 졸업(박사)
1969년 7월 ~ 1978년 7월 한국과학기술연구소 선임연구원
1978년 8월 ~ 1985년 5월 한국전자기술연구소 컴퓨터연구부장
1985년 6월 ~ 현재 한국전자통신연구소 컴퓨터연구단장
관심분야 : S/W공학, 컴퓨터통신, 데이타베이스, 인공지능