나에게 맞는 제품은 어느 것일까?

현재 기업은 수많은 정보의 효율적 관리를 위해 데이터베이스관리시스템(DBMS)을 널리 사용하고 있으며, 기업 고유의 데이터 처리에 적합한 DBMS을 도입하려고 많은 노력을 기울이고 있다. 업체별 DBMS제품의 특성 및 장·단점을 사용자와 개발자 측면에서 비교·검토하여 DBMS 도입시에 필요한 가이드라인을 제시한다. 〈편집자〉

김상하/ 으뜸정보기술 사장

주요 DBMS 종류별 특성 비교

정보시스템을 구축하는데 있어서 가장 핵심이 되는 기초 분야는 정보 통신 분야와 데이터베이스 분야를 꼽을 수 있다. 그 중에서도 데이터베이스는 매우 중요한 정보기술 분야에 속한다. 그런데 이렇게 중요한 데이터베이스는 불행하게도 그것을 도입하고 개발하고 사용하는 개발자나 사용자들이 잘못 도입하여 개발하고 사용함으로써 심대한 문제점을 야기시킨다.

그 이유는 수많은 종류의 데이터베이스와 관련 제품들이 시중에 많이 나와 있고 각각의 제품에 대한 정확한 이해와 경험이 부족한 경우가 많기 때문이다.

〈표 1〉은 파일 시스템(File System)과 데이터베이스 관리시스템(DBMS)의 종류별 특성을 비교한 것이다. 표에서 보다시피, DBMS는 파일 시스템의 데이터의 중복성과 최신 정보기술에 부적합한 3세대 언어의 문제점으로 등장하게 되었다.

DBMS는 원칙적으로 데이터베이스를 구성하는 3가지 구성요소의 측면에서 살펴야 한다. 그 3가지 구성요소는 데이터 구조, 데이터 조작, 데이터 무결성이다. 데이터 구조는 DB에 저장된 데이터가 담기는 곳으로서 파일(파일시스템), 세그먼트(HDB), 테이블(RDB), 레코드타입(NDB), 클래스(OO/ORDB)로 불린다.

데이터 조작은 DB에 저장된 데이터 구조에서 데이터를 조작(등록, 수정, 삭제, 조회)하는 방법을 기술한 것으로서 데이터 검색 방법(Data Access Method)을 뜻한다.

데이터 검색 방법은 포인터 액세스(Pointer Access)와 속성 조인(Attribute Join)의 두 가지 방법이 있는데, 관계형 데이터베이스(RDB)가 속성 조인에 의하여 데이터를 조작하는 데에 비하여 파일 시스템에서 객체지향형 DBMS까지는 포인터 추적에 의하여 데이터를 조작한다.

데이터를 액세스 시에 파일 시스템은 애플리케이션 프로그램을 통하여 억세스하고, HDB는 트리 구조, NDB는 네트워크 구조, RDB는 외부키(Foreign Key)로서 두 테이블을 조인하게 되며, OODB는 데이터와 관련 조작 프로그램을 함께 가지고 있는 캡슐화(Encapsulation)와 유전(Inheritance) 등의 구조로서 데이터를 액세스하며, ORDB는 OODB와 RDB의 특성인 조인과 캡슐화와 유전 등을 함께 가지고 데이터를 액세스한다.

데이터 무결성은 크게 실체 무결성(Entity Integrity Rule; Primary Key Integrity Rule), 참조 무결성(Referential Integrity Rule), 도메인 무결성(Domain Integrity Rule), 속성 무결성(Trigger) 등으로 나뉘어 진다.

DBMS와 파일 시스템의 가장 큰 차이점인 무결성 규칙은 DBMS가 자동으로 제공하는 것으로서 잘 이용하면 품질이 좋고 유지보수가 간편한 DB 시스템을 구축하는데 커다란 공헌을 하게 된다.

최근에 가장 많이 사용되고 있는 RDB는 다른 DB와 데이터 모델링, 데이터 조작 방법, 데이터 무결성 등에서 커다란 차이를 보이고 있고, DB 시스템을 구축하는데 있어서 개발자에게 매우 곤혹스러움을 느끼게 하는 여러 측면들이 존재한다.

특히 RDB는 SQL(구조적 질의어; Structured Query Language)을 통하여만 DB를 액세스할 수있는 유일한 수단인 것에 비하여 SQL 그 자체는 집합이론을 기반으로 데이터 연산을 하기에 이해와 사용이 매우 어려운 것으로 알려져 있다.

또한 RDB는 물리적 디스크에 있는 데이터들이 메모리내로 불려와 조인 작업을 통하여 연산을 하기에 필연적으로 성능(Performance) 문제가 발생한다. 그리고 파일 시스템 등에서 데이터를 빨리 검색하는데 사용되었던 인덱스(Index)가 RDB에서는 데이터를 보유하고 있는 또 하나의 테이블이 되어 조인시의 성능 문제를 유발한다.

그러나, RDB는 매우 뛰어난 장점들을 많이 가지고 있으며, 최근에는 RDB의 개발 경험과 분석, 설계 경험이 풍부한 개발자가 많아지고 있어서 그 위력을 한층 더해가고 있다.

OO/ORDB는 시기적으로 그 활용이 임박한 DB이다. 복잡한 데이터 타입과 사건들의 저장을 위하여 객체 지향 개념을 가진 DB와 분석, 설계 및 관련 정보기술의 도입은 필수적이다.

최근에는 데이터 모델링이 간편한 RDB와 멀티미디어 데이터 타입의 저장이 가능한 OODB와의통합 및 확장을 이룬 ORDB가 크게 그 시장 점유율을 높여가고 있는 중이다.

관계형 DBMS의 종류별 특성 비교

관계형 DBMS(RDBMS)는 최근 들어 시장의 성장이 정체를 보이고는 있지만, 여전히 정보시스템을 구축하는데 있어서 중요한 변수가 된다. 예년에는 개발자 대부분이 RDBMS의 데이터 모델링과 SQL언어에 대한 연구에 상당한 관심을 보였으나, 요즈음은 DB 성능 튜닝 및 DB 관리로 초점이 옮겨가고 있다.

그러나 RDBMS 자체의 도입 및 판매에 대하여는 아직도 벤더의 마케팅 우위에 좌우되고 있다. RDBMS의 도입을 검토하고 있는 기업의 전산 담당자나 한 두 개의 DBMS의 사용 경험 그것도 한정된 경험의 개발자들의 이해를 돕기 위하여 〈표 2〉에서 RDBMS의 종류별 특성을 비교해보고, 계속해서 〈표 3〉에서는 RDBMS를 활용하여 성공적인 DB 시스템을 구축하기 위하여 필요한 기능들을 나열했으며, 〈표 4〉에서는 RDBMS의 도입시의 고려 사항을 살펴보았다. 〈표 2〉는 많이 사용되는 대표적인 RDBMS의 장점과 단점을 나열하였다.

객관적으로 부족한 기능이 많더라도 자신이 사용해본 제품을 타 제품보다 선호하는 개발자들의 평가를 뒤로하고, 일반적으로 사용자들이 평가하는 항목들이 예시되어 있다. 특히 각 제품들은 업종별, 사용자 단체의 규모별, 사업장의 분산 유무별, 그리고 운영체계 등의 RDBMS 주변 환경과 비교하여 각각 강점과 약점을 가지고 있다.

개발자 입장에서는 애플리케이션 개발에 도움을 주는 유틸리티(Utility)의 다양성, SQL 함수의 다양성, 성능 튜닝을 지원하는 유틸리티의 존재와 RDBMS가 지원하는 무결성 관련 프로시저의 지원 및 3rd Party 개발도구의 지원 가능성 등은 매우 중요하다.

RDBMS의 도입자 입장에서는 이기종간의 호환성 및 이식성과 분산 처리의 지원 기능, 대규모 데이터의 처리가 가능한 병행처리 프로세서와 백업, 복구 등의 기능이 매우 중요하다. 특히 RDBMS의 도입시 사후 지원과 문제점 발생시의 해결 지원 체제 등은 RDBMS의 기능 못지 않게 중요하다.

〈표 3〉은 RDBMS로서 갖추어야 할 구비 조건을 나타낸 것이다. 데이터 타입은 애플리케이션 개발시에 도움을 주는 특성임과 동시에 다른 RDBMS와의 호환성을 가능하게 해주는 SQL 표준과 관련이 있는 항목이다.

질의어 최적화 기능은 SQL을 잘 구사하고, RDBMS를 활용한 정보 시스템을 효과적으로 구축하기 위하여 파일 시스템의 컴파일러와 유사한 SQL 최적화기(Optimizer)의 기능이 매우 중요한데, 규칙기반 최적화기(Rule-Based Optimizer)와 비용기반 최적화기(Cost-Based Optimizer)의 두 가지가 있다.

RDBMS의 물리적인 디스크 및 메모리 관리 기능의 자동화 지원 등이 시스템 운

영의 최적화를 보장해 주는 중요한 구비 조건이 된다. 최근 들어 멀티미디어 데이터를 저장할 수 있도록 RDBMS의 기능이 확장되고 있는데 이 또한 고려되어야 할 중요한 구비 조건이다.

대부분의 RDBMS는 클라이언트/서버 구조를 지원하도록 설계가 되는데 분산 처리 기능, 성능, 클라이언트 애플리케이션의 지원 기능 등을 제공하여 구축될 시스템의 성능 향상 및 백업, 보안에 크게 영향을 미친다.

〈표 4〉는 RDBMS 도입시의 고려사항을 나열하고 있다. 이 항목들은 도입 실무자의 사용할 환경을 기준으로 하여 취사 선택이 될 항목들을 다수 포함하고 있다. 나열된 여러 항목들의 활용 방법과 구현 방법 및 운영 방법 등에 관심을 기울이는 것은 매우 중요하다. 최근에는 국내 모든 기업이나 공공기관의 정보시스템 환경에 RDBMS가 필수적으로 설치되어 운용되고 있기 때문에, RDBMS의 도입 시에 참조를 할 수 있는 기회가 많아졌다.

그리고 각 RDBMS는 별반 기능의 차이가 없는 것이 일반적인 통념이 되고 있다. 계속적으로 새로운 기능이 추가되고 있어서 도입자나 개발자들이 RDBMS의 편파적인 비교나, 새로운 다른 RDBMS의 도입보다는 기존의 RDBMS의 장래의 새로운 기능의 면면에 더욱 관심을 기울이고, 더 나은 효과적인 사용에 관심을 기울이는 것이 현명한 일이다.

객체지향형 및 객체관계형의 특성 비교

객체지향 데이터베이스(OODB/ORDB)는 아직 보급이 미흡한 수준이며 각 기능별로 성능면에서 향상시켜야 할

최근에는 데이터 모델링이 간편한 RDB와 멀티미디어 데이터 타입의 저장이 가능한 OODB와의통합 및 확장을 이룬 ORDB가 크게 그 시장 점유율을 높여가고 있는 중이다.

여지가 많이 존재하고 있다. RDBMS가 기술적으로 안정되고 애플리케이션 측면에서도 신뢰도가 입증된 상황에서 비록 개념적으로 우수하다고 할지라도 OODB가 시장에서 확고한 자리를 잡을 것인지 확신과 시기의 예측을 한다는 것은 곤란하다.

그러나, OODB가 조만간 대세를 잡을 것이라는 것은 분명하다. 그 이유로서 첫째는, 객체 식별자(OId; Object Identifier)를 사용하여 잠재적으로 처리 성능이 우수하고 둘째는, 데이터 스키마를 확장하거나 여러 기능들을 효과적으로 적재하여 활용할 수 있고 유전 관계를 통해 코드 등의 재사용이 가능해 졌고 셋째는, 유전 관계와 추상 데이터 타입 (Abs-tract Data Type)의 사용으로 보다 의미적인 정보를 저장할 수있어서 모델링 기능이 강화되었으며 가장 중요한 사항인 멀티미디어 데이터에 대한 확실한 저장과 처리 기능을 제공한다는 다른 DBMS와 비교하여 커다란 차별화된 장점이 있다.

이러한 사실들을 기반으로 현재 사용되고 있는 OODB/ORDB의 종류와 그것들의 특성들을 비교해 놓은 것이 〈표 5〉이다.

객체관계형 데이터베이스(ORDB)는 전통적인 RDB의 서비스 외에 보다 풍부한 객체 구조와 규칙에 대한 지원을 제공한다. 풍부한 객체 구조란 이미지나 음성, 비디오 데이터 등 비전통적인 데이터 요소를 저장 또는 처리하는데 필요한 기능을 말한다.

ORDB는 비절차적 데이터 액세스와 데이터 독립성 등을 포함하여 OODB가 제공하는 캡슐화(Encapsulation), 유전(Inheritance)관계 등의 기능을 제공하며 대표적인 프로그래밍 언어인 C++와의 인터페이스를 공통으로 한다. 풍부한 데이터 타입, 유전관계, 캡슐화등의 기능이 지원되어 RDB에서 표현하지 못한 기능들을 표현할 수 있다.

〈표 6〉은 OODB/ORDBMS를 도입시의 고려사항을 나타낸 것이다. 캐드, 멀티미디어, 지리정보 등 소수의 적용 분야를 바탕으로 그 세력을 확장할 것으로 예상되는 OODBMS는 기업 내부의 데이터의 복잡화로 인하여 그 효용성이 증가되고 있다. RDBMS와 마찬가지로 OODBMS의 도입시 도입 실무자의 운용할 환경의 정의가 선결 조건이 된다.

그러나, OODBMS의 사용에는 객체지향 모델링에 정통한 전문가의 도움과 이를 지원할 자체분석, 설계 요원들의 준비가 필요하다. 다른 DBMS와 달리 객체지향 프로그래밍은 분석, 설계시의 내용을 정확하게 코드로 변환하는 과정일 뿐이어서 분석, 설계의 작업 과정은 매우 중요하다.

RDB 벤더, ORDB 벤더, 그리고 순수 OODB 벤더들의 시장 쟁탈 속에 OODB/ORDBMS의 확산은 필수적이고, 이에 따라 더 많은 연구가 진행되게 될 것으로 예상된다.

〈표 1〉 FS & DBMS 종류별 특성 비교

구 분	특 징	종 류
파일 시스템 (File System)	o 주요 관점의 대상은 애플리케이션(Application; 응용)이고, 관련된 데이터는 파일에 저장 o 애플리케이션에 특화된 형태로 데이터가 저장되고, 만약 조금이라도 다른 형태의 데이터를 필요로 하는 애플리케이션의 경우, 중복된 데이터라도 다른 형식으로 다시 저장해야 함 o 단, 부수적인 처리가 필요없는 단순한 데이터의 입출력 작업인 경우에는 간단하게 처리가 가능함	o Flat File o ISAM File o VSAM File
계층형 데이터베이스 관리시스템 (HDBMS)	o 애플리케이션 환경의 여러 명의 사용자가 통합된 데이터를 공유 o 파일시스템에서 처리 못한 부수적인 처리들을 DBMS들이 어느 정도 처리 o 그러나 데이터베이스 모델링 및 관리가 용이하지 않음 o Record들을 계층구조로 표현한 데이터 모델 –Database는 세그먼트(레코드 타입)로 이루어진다. –한 레코드 타입은 여러 개의 레코드를 포함한다. –한 레코드는 필드들로 구성된다. –한 필드는 한 개 또는 그이상의 자료항목들(반복그룹)을 포함한다. –한 레코드는 다른 레코드 등에 대해 한 개 이상의 포인터들을 가지고 있다. (부모는 반드시 자식을, 자식은 형제 레코드들의 포인터와 그들만의 자식에 대한 포인터를 가지고 있다.) –Database Search(검색)는 첫 번째 레코드를 찾아낸 후에 다음 Record를 Pointer로써 읽는다. –Data사이의 동적(Dynamic)인 상호 연관사항은 존재하지 않는다. –Database Schema에 대한 동적인 변화는 없다. –Data Independence의 보장이 안 된다. 즉 Record Format이 변하면 애플리케이션도 모두 바뀌어야 한다.	o IBM IMS DB
네트워크형 데이터베이스 관리시스템 (NDBMS)	o HDBMS와 특징이 유사하다. o 데이터베이스는 레코드 타입과 링크(Pointer들의 집합)로 구성된다. o 한 레코드는 자식들과 형제 레코드들에 대한 포인터와 HDBMS에서는 불가능했던 부모 레코드들에 대한 포인터를 가질 수있다. o 데이터 모델링이 복잡하여 사용이 일반화되지 않았다. o SQL(Structured Query Language) 지원	o CODASYL DB
관계형 데이터베이스 관리시스템 (RDBMS)	o 이전에 애플리케이션에서 처리해야 했던 많은 기능들을 DBMS가 지원 –데이터 무결성, 보안, 권한, 트랜젝션 관리, 록킹(Locking)등 o 데이터 모델링이 간편해지고 애플리케이션 개발을 용이하게 지원 o 데이터 모델링이 너무 간단하기 때문에 복합 애플리케이션(CAD/CAM, CASE, Multimedia, GIS 등)에는 적합하지 않음 o 데이터베이스 특징 –데이터베이스는 테이블들로 구성된다. –레코드(로우; 행)는 필드(컬럼)로 구성된다. –한 필드는 단지 하나의 Data Item을 갖는다. –레코드는 다른 레코드에 대하여 어떤 Pointer라도 갖지 못한다. –Data사이의 동적(Dynamic)인 상호 관계는 조인(Join)을 통하여 일어난다. 성능 문제가 중요하게 발생함 –데이터베이스 스키마(Schema)에 대한 동적인 변화들이 가능하다. 예) 테이블에 대한 새로운 필드의 추가, 삭제 –한 필드는 하나의 Data Item만을 포함한다. 레코드들을 중복시킴으로서 성능 함정에 빠질 수 있다. –멀티 미디어 자료 지원이 불가능	o IBM DB2 o ORACLE o INFORMIX o SYBASE o INGRES o MS-SQL 등

구 분	특 징	종 류
객체지향형 데이터베이스 관리시스템 (OODBMS)	o 멀티미디어 데이터의 지원이 가능 o 재사용 가능한 객체 모듈의 지원 o 캡슐화(Encapsulation), 계승(유전; Inheritance Hierarchy), 폴리모피즘(다형성 : Polymorphism)등 프로그래밍에 도움을 주는 특성의 보유 o 레코드와 레코드 사이의 데이터 검색이나 작업이 포인터(Pointer)에 의하여 이루어지므로 성능 문제가 대두되지 않는다. o 분석 · 설계(데이터 모델링)의 품질이 프로그래밍에 절대적인 영향을 준다. o 상용 DBMS 대부분이 아직 개발자들을 위한 충분한 개발지원도구와 이기종의 시스템이나 운영체계 및 호환을 위한 인터페이스를 갖추고 있지 못하다. o 객체지향 방법론이 완벽하지 못하다.	o Objectivity o O2 o Versant o Ontos o Gemstone
객체관계형 데이터베이스 관리시스템 (ORDBMS)	o OODBMS와 특징이 비슷 o OODBMS의 장점과 RDBMS의 장점만을 취함 –RDBMS의 데이터 모델을 그대로 활용하여 어렵고 까다로운 OODBMS의 데이터 모델링 문제를 해결 –기존의 RDBMS를 기반으로 하는 많은 DB시스템과의 호환 가능 –RDB의 중요한 문제점들인 반복그룹, 포인터 추적, 자료형의 한계를 제거 –복잡한 DB Schema Modeling에 대하여 Encapsulation, Inheritance의 이점 추가 –로우(행, 레코드)나 컬럼(필드, 속성)이 한 개 이상의 Data Item값을 갖도록 반복그룹의 허용	o UniSQL o Object Store o 순수 RDBMS의 확장 모델등

〈표 2〉 관계형 DBMS의 종류별 특성 비교

종 류	장 점	단 점
Oracle	o 많은 사용자가 존재함 o 입증된 제품의 우수성 o PC급에서 Mainframe급까지 모두 설치됨 o 3rd Party의 강력한 지원 o 분산처리 지원 기능의 우수성 o SMP 및 MPP의 지원	o 신제품의 출시가 늦어짐 (버전 발표의 늑장) o DBMS를 운영하기 위하여 많은 하드웨어 자원의 필요 o 복잡한 DBMS 관리 o 가격이 동종의 DBMS보다 비쌈 o 모든 제품에 Kernel이 필요 o 애프터서비스의 부족 o 배우기 힘든 제품 기능들의 존재
Sybase	o Client/Server용으로 설계되어 성능과 분산처리 지원이 탁월 o 타 DB에 비하여 DBMS를 운영하기 위하여 적은 하드웨어 자원만으로도 충분 o 3rd Party 지원도구의 지원 우수 o Open Server/MDI Gateway 지원 o PowerBuilder와의 결합 및 지원 우수 o Replication Server 특성 우수 o Record Tracking 우수	o 복잡한 DBMS 관리 o 부족한 확장성 o Sybase 자체 개발 등의 지원도구의 부족 o 예전 버전에서의 표준SQL의 지원 불량 o Microsoft에 의한 대체 DBMS의 등장으로 독창성 훼손

종 류	장 점	단 점
Informix	o 안정된 Kernel(시스템의 안정성) o 사용자들의 만족도 우수 o Low-end Unix에서의 운영 우수성 o 풍부한 4GL 도구의 지원 o 풍부한 경험을 가진 개발자 및 사용자의 존재	o PC급 지원기능의 한계 o VMS의 지원 불가 o 3rd Party 지원도구의 부족(ODBC를 통한 DB접속 의존) o 경쟁시장에서의 마케팅 부족
Ingres	o 통합된 도구세트의 지원기능 훌륭 o 사용자 정의 데이터 타입, 함수, 연산자 등의 지원 o VMS, Unix 사용자의 큰 만족도 o Replication Service기능의 우수	o PC전략의 부재 o 3rd Party 지원도구의 부족 (ODBC를 통한 DB접속 의존) o 도큐먼테이션 기능의 부족 o 회사의 불확실한 미래(CA의 인수후)
Microsoft SQL Server	o Sybase의 장점들을 물려 받음 o 저렴한 제품 가격 o Windows NT환경에서 최적의 기능 및 성능을 발휘하도록 설계됨(SMP, 하드웨어 확장성 우수) o Microsoft의 '토털 솔루션' 전략의 중심축 -Networking -Database -Tools o 단순한 데이터베이스 기능	o Sybase의 특성들을 물려받음 (Microsoft의 Family로서의 기능을 충분히 하도록 범용 RDBMS의 기능을 대폭 축소) o Microsoft는 RDBMS만에 주력하는 회사가 아니므로 충분한 지원 및 더 나은 DBMS로의 발전 전망 불투명 o 최근 구현된 Sybase의 신 기능의 결여 o DBMS 전문가의 부족 o 단순한 데이터베이스 기능
Progress	o 사용자 만족도 우수 o 통합 개발지원도구 o DBMS의 안정성 우수 o 소규모 Unix Platform들을 기반으로 할 때 운용성 우수	o 3rd Party 지원도구의 부족 o Server의 주요 기능들의 부족 -Replication -SMP(Symmetrical Multi-Processor) -MPP(Massive Parallel Processor) -개발지원 도구 -성능튜닝 도구 o 완전한 관계형 SQL API의 조건 미흡
Qupta SQL Base	o 간단한 DBMS 관리 o 통합 윈도우 개발 및 사용자용 도구의 지원 o IBM DB2와의 호환성 o Client/server용으로의 구조의 우수성	o 제품의 안정성 미흡 o 3rd Party 지원도구의 부족 o 제한된 호환성 o Stored Procedure와 Trigger의 결여 o Router의 높은 가격 o 고객 지원의 미흡
DB2	o IBM Product Line의 호환성 o DRDA(Distributed Relational Database Architecture) 구조 o 저렴한 가격 o RDBMS와 Platforms의 안정성 o CICS와의 통합사용이 가능 o 원격관리의 우수성	o Stored Procedure등의 기능 미흡 o 제한된 호환성 o 3rd Party 지원도구의 부족 o OS/2의 성능의 문제점 o 시장이 편중됨(IBM 위주)

〈표 3〉 RDBMS 구비 조건

구 분	구분상세	내용설명
데이터 타입	SQL2에서 정의한 데이터 타입	ISO/IEC9075-1992년 정의된 데이터 타입으로 Character, Integer, Small Integer, Decimal, Float, Double Precision, Real 등이 있다.
	사용자 데이터 타입 정의 기능	사용자가 SQL2 데이터 타입을 이용하여 자신이 필요한 데이터 타입을 정의할 수 있는 기능으로, 자주 사용하는 데이터 타입을 편리하게 정의할 수있다.
최적화	디스크 입출력 향상기법	트랜잭션 수행에 필요한 디스크 입출력에 대한 지연시간을 최적화하여 응답시간을 감소시키는 기능
	질의어 최적화 기능	사용자가 입력한 질의어가 가장 효율적으로 실행이 가능하도록 최적의 사용계획을 결정하여 제공하는 기능
	데이터베이스 최적화 기능	데이터베이스가 삭제 및 추가를 반복하면서 저장형태가 변경되어 비효율적으로 변화하는 것을 방지하기 위해 구성 블럭 등을 고려, 최적으로 재구축하는 기능
	저장 프로시저 기능	클라이언트에서 호출시 명령어 전체가 이동하지 않고, 프로시저 이름만 이동하여 사용함으로써 클라이언트와 서버간 통신량을 감소시키는 기능
한글 처리	한글 표준 지원	국가 표준 KSC 5601로 제정된 한글 코드를 지원하는 기능
	유니 코드 지원	국제 표준 ISO 10646으로 제정된 코드 방식으로 한글을 지원하는 기능
	한글 필드명, 테이블명 정의 기능	스키마, 테이블, 뷰, 컬럼등 객체들의 이름을 한글로 정의하여 사용할 수 있는 기능
	한글 검색 기능	한글 문자열을 이용하여 데이터를 검색할 수 있는 기능
	한글 메시지 기능	사용자에게 오류, 설명, 도움말 등의 메시지를 한글로 제공하는 기능
	한글 데이터 입출력 기능	파일 및 주변기기와 한글로 데이터를 입출력하는 기능
멀티미디어 처리 서버 구조	BLOB 지원	멀티미디어 데이터 저장을 위해 BLOB(Binary Large Object) 데이터 타입을 지원하는 기능
	데이터 입력/삭제 기능	멀티미디어 데이터를 입력하거나 삭제할 수 있는 기능
	데이터 검색 기능	멀티미디어 데이터를 검색할 수 있는 기능
SQL 지원	멀티 서버 기능	다수의 서버엔진을 실행하여 동시에 여러 작업을 수행하는 기능
	동적 부하 기능 (Dynamic Load Balancing)	다중 서버시스템에서 특정 서버로 작업이 집중되는 것을 방지하고 서버간 처리량의 균형을 유지하는 기능
데이터베이스 구조 변경	표준 SQL(ISO 9075)	관계형 데이터베이스에 접근하는 언어로서, ISO에서 국제표준(ISO 9075 , 1992)으로 제정한 SQL을 지원한다.
	Embedded SQL	일반 프로그래밍 언어에서 SQL을 사용하여 데이터베이스 작업을 수행할 수 있는 인터페이스를 지원한다.
	확장SQL (Extended SQL)	각 DBMS 업체에서 기능 향상을 목적으로 고기능의 SQL을 표준 SQL에 추가하여 지원한다.
	KSQL 지원	한국 전산망 표준(KIS 0044, 관계형 데이터베이스 언어표준, 1994)으로 제정된 관계형 데이터베이스 접근 언어인 SQL을 지원한다.
성능의 향상	데이터베이스 용량의 동적 설정 기능	데이터베이스의 기존 데이터 영역상의 데이터를 저장하여야 할 경우 새로운 저장공간을 동적으로 할당하는 기능
	컬럼 추가 기능	테이블의 열을 추가함으로써 테이블을 재구성할 수 있는 기능
	인덱스 생성, 삭제, 변경 기능	인덱스를 생성, 삭제하거나 기존의 인덱스를 변경할 수 있는 기능
	액세스 기간의 단축	저장 프로시저, 미들웨어(TP-Monitor등)의 지원 기능
	병렬처리 기능	Multi-Thread 기능, Multi-Processor(= Multi-Server화) 기능
트랜잭션 처리기능	데이터 무결성 지원	데이터 정합성 보존 기능, 트리거 기능
	신뢰성	데이터베이스 복제기능, 네트워크 장애 복구 기능
	운용 관리 기능	Online Backup, Online Restore, 동적 재구성 기능 지원
대규모 시스템 지원	분산 데이터베이스 지원	2단계 커밋, 자료 저장
	동종 DB간의 연계 지원	복수 DB의 일원 관리 및 부하 분산 관리 기능
	이종 DB간의 연계 지원	게이트웨이 소프트웨어 지원

〈표 4〉 RDBMS의 도입 시의 고려사항

가격 대 성능, 응답 시간, 객체 기능, 자료 리플리케이션(Replication) 기능, 이미지 · 사운드 · 동화상(멀티미디어) 처리, 보안 기능, 데이터 무결성 기능(참조 제어, 트리거등), 백업 및 복구 기능(DB 파티션, 병렬 백업과 복구, 온라인 처리등), 애플리케이션 개발 시간(SQL함수의 다양성, 개발 도구의 지원, 저장 프로시저, 저장 함수, 순환 SQL 지원등), 이식성, 상호 운용성(게이트웨이), EUC환경 · GUI지원, 유지 보수, 한글 지원, 교재 · 매뉴얼, 확장성, 에플리케이션 지원(개발 도구), 튜닝 지원(유틸리티, 대량의 자료처리/병렬 처리 프로세서), SQL 최적화기의 특성(규칙기반/비용기반, Explain), 잠금기능(행/페이지 단위), 사용자 교육, 사후 지원, 분산 처리 기능(2단계 커밋, S/W 변경관리), 지원 데이터 타입, 클라이언트/서버 지원(RPC, 개발도구 지원 기능), 대규모 DB 구축 및 처리 능력(고속 로드 유틸리티), 비관계형 DB와 동시 사용시 데이터 무결성 보장

〈표 5〉 객체지향형 및 객체관계형 DBMS의 종류별 특성 비교

구 분	특 징	종 류
Postgres	o 객체 SQL을 사용하는 Ingres의 객체지향 확장형 –Ingres에 유전관계, 추상데이터 타입, 프로시저 타입의 추가 o 추상 데이터 타입의 연산들은 3GL인 C언어로 정의함 o 추상 데이터 타입에 유전관계를 관련시킬 수가 없음. o 객체지향 개념의 캡슐화 기준에 미흡함	객체관계형 DBMS
IRIS	o HP의 객체지향 데이터베이스 –HP의 Allbase RDBMS 상위에 객체관리자를 구축한 것 o 복잡한 구조를 지원하는 Object SQL을 지원(SQL의 확장) –객체 식별자로서 객체는 시스템이 제공하는 독특한 식별자를 가짐 –사용자 정의 함수로서 사용되는 데이터에 대한 독자적인 함수를 정의하여 질의에서 사용이 가능 –언어에 대한 새로운 사고 방식을 추진하기 위해 구문을 변경 o Object SQL은 유전관계로 타입 계층을 지원 o Object SQL은 4가지 집합 타입을 지원 –SET, BAG, LIST, TUPLE	객체관계형 DBMS
UniSQL	o 관계형과 객체지향 데이터베이스의 통합 o 김 원 박사가 개발 o 기존의 SQL사용자가 쉽게 사용이 가능한 SQL/X의 사용 o 관계형 데이터베이스의 확장 –RDB가 한가지 데이터 타입만을 지원하는 것에 비해, 테이블 자체를 데이터 타입으로 지원 –RDB가 속성 값으로 단위(Atomic) 값만을 허용하는데 에 비해, 복수의 값들을 가질 수 있음 –프로그램을 테이블의 속성으로 저장이 가능 –계층구조로 조직되어 유전관계를 가질 수 있음	객체관계형 DBMS
Objectivity/ DB	o C++를 DDL, DML로 사용하는 언어확장형 데이터베이스 o SQL의 지원 o 분산된 peer-to-peer 구조를 구현함으로서 독립형이나 분산 애플리케이션의 지원 o UNIX, VMS, PC등에서 운영이 가능 o 데이터 브라우저, 타입 브라우저, 디버깅 도구 등의 다양한 개발 도구의 지원 o DEC의 객체지향 데이터베이스, Sybase의 멀티미디어 저작도구인 Gainmomentum의 데이터베이스로서 사용	객체지향형 DBMS
Object Store	o 멀티서버, 멀티 클라이언트 구조의 언어 확장형 데이터베이스 시스템 o C++를 DDL로 사용 o DBMS 런타임, 응용 프로그램 인터페이스, C++ 개발도구 o C언어와의 호환성 o SQL과 다른 형태의 상위 레벨 질의어를 사용 o 객체지향 SQL의 지원 예정 o 대부분의 질의나 DBMS 프로세싱은 클라이언트 쪽에서 수행	객체지향형 DBMS

구 분	특 징	종 류
Ontos	o 멀티 서버, 멀티 클라이언트 구조의 C++ 언어 확장형 데이터베이스 시스템 o C++을 DDL로 사용 o 4GL인 Shorthand, X윈도우상의 GUI 개발도구인 Ontos Studio, SQL의 객체지향 확장인 Ontos SQL, DBDesigner, DBATool등으로 구성됨	객체지향형 DBMS
Versant	o 멀티 서버, 멀티 클라이언트 구조의 계층화된 데이터베이스 시스템 o 최하위 레벨에서 객체지향 데이터베이스 관리 시스템과 계층화된 접근 방식을 사용 o C, C++, Objectworks/Smalltalk, Smalltalk-V/VM, Object SQL등과의 게이트웨이 제공 o UniSQL의 SQL/M의 채택	객체지향형 DBMS
Gemstone	o 멀티 서버, 멀티 클라이언트 구조의 데이터베이스 시스템 o 연산의 데이터베이스에 저장 - 분산 환경에서 서버가 연산 서비스를 제공하므로 클라이언트에 전송할 필요 없이 서버에서 처리할 수있으므로 성능 면에서 탁월함 o Digitalk와 Smalltalk을 모두 확장한 SmalltalkDB 사용 o 객체 관리, 동시성 컨트롤, 트랜잭션과 복구 서비스 등을 제공하는 Gem과 SmalltalkDB의 DB컴파일, 클래스와 매소드를 제공하는 Stone으로 구성됨 o C, C++ 인터페이스 제공	객체지향형 DBMS
O2	o 단일 서버, 멀티 서버 클라이언트 구조의 데이터베이스 시스템 o 클라이언트 시스템은 객체 관리자, 타입 관리자, 매소드 관리자로 구성됨 o 서버(O2 런타임)는 데이터베이스 관리자, 트랜잭션 관리자등으로 구성됨 o 객체 4GL인 O2C, C++을 DLL로 사용하여 스키마 생성 o C에 기반을 두고 상위에 객체지향 층 O2를 포함하여 생성된 혼합형 언어인 O2C로 객체를 정의하고 그들에게 메시지를 전달하며 연산을 수행 o C, C++ 인터페이스 제공	객체지향형 DBMS

〈표 6〉 객체지향형 및 객체관계형 DBMS의 도입시의 고려사항

애플리케이션의 성격정의(CAD나 GIS, 멀티미디어 정보의 구축유무), 객체지향 분석/설계기술의 조직내의 성숙도, 언어확장형과 관계형 확장형의 결정(SQL의 경험이 적거나, 복잡한 연산을 수행할 필요가 있을 경우는 언어 확장형의 선택), 언어 확장형일 경우의 표준 지원 유무/ 완전한 객체 모델의 지원 유무/ GUI개발환경의 지원 유무/ 사용자에게 익숙해진 데이터베이스 보안·복구·무결성의 제공 유무/ 객체SQL과 같은 상위의 질의 언어 제공 유무/ 버전변경·분산처리·장기 트랜잭션의 처리 기능 유무/ 만족할 성능의 제공 유무의 점검, 기존 데이터베이스의 객체지향 데이터베이스로의 이전 용이성 점검.