1.1 DBMS 개요

1.1.1 데이터베이스의 정의와 특징

데이터베이스를 ‘데이터의 집합’이라고 정의한다면 DBMS는 이 데이터베이스를 관리/운영하는 역할을 한다.  
여러 명의 사용자나 응용 프로그램이 공유하고 동시에 접근이 가능해야 한다.  
대용량을 관리하거나 여러 명의 사용자가 공유하는 개념이다.

데이터베이스는 ‘데이터의 저장공간’ 자체를 의미하기도 한다.  
MySQL에서는 ‘데이터베이스’를 자료가 저장되는 디스크 공간(주로 파일로 구성됨)으로 취급한다.

**데이터의 무결성(Integrity) 제약조건(Constraint)**

데이터는 어떤 경로를 통해 들어 왔던지 데이터에 오류가 있어서는 안 된다.  
예를 들어 학번은 반드시 있어야 하며, 중복이 되면 안 되는 제약 조건이 있다.

**데이터의 독립성**

데이터베이스의 크기를 변경하거나 데이터 파일의 저장소를 변경하더라도 기존에 작성된 응용 프로그램은 전혀 영향을 받지 않아야 한다.  
즉, 서로 의존적 관계가 아닌 독립적인 관계여야 한다.

**보안**

데이터에 접근이 허가된 사람만 접근할 수 있어야 하고, 계정에 따라 다른 권한을 가져야 한다.

**데이터 중복의 최소화**

동일한 데이터가 여러 개 중복되어 저장되는 것을 방지한다.

**응용 프로그램 제작 및 수정이 쉬워짐**

기존 파일시스템을 사용할 때는 각각 파일의 포맷에 맞춰 개발해야 하는 응용 프로그램을 데이터베이스를 이용함으로써 통일된 방식

**데이터의 안전성 향상**

대부분의 DBMS가 제공하는 백업/복원 기능을 이용함으로써, 데이터가 깨지는 문제가 발생할 경우에 원상으로 복원 또는 복구하는 방법이 명확해진다.

================================================================

1.1.2 데이터베이스의 발전

오프라인으로 관리 -> 파일시스템의 사용 -> 데이터베이스 관리시스템

처음에는 종이로 하다가 컴퓨터를 사용하면서 파일시스템을 사용하게 되었다.  
그러나 기능마다 파일이 생겨야 하는 파일시스템은 데이터의 양이 많아지면서 중복으로 인한 불일치가 생겨 데이터베이스 관리시스템이 나오게 된 것이다.

파일시스템의 단점을 보완하고 대량의 데이터를 보다 효율적으로 관리하고 운영하기 위해서 사용되기 시작한 것이 DBMS(DataBase Management System)다.

================================================================

1.1.3 DBMS

DBMS 유형은 크게 계층형(Hierarchical), 망형(Network), 관계형(Relational), 객체지향형(Object-Oriented), 객체관계형(Object-Relational) 등으로 분류된다.

**계층형 DBMS**

각 계층은 트리(Tree) 형태를 가지며 1:N 관계를 갖는다.  
예를 들어 사장 1명에 부서 3개가 연결되어 있는 구조다.  
계층형 DBMS의 문제는 처음 구축한 이후에는 그 구조를 변경하기가 상당히 까다롭다.  
또한, 주어진 상태에서의 검색은 상당히 빠르지만, 접근의 유연성이 부족해서 임의의 검색에는 어려움이 따른다.

**망형 DBMS**

망형 DBMS는 계층형 DBMS의 문제점을 개선하기 위해 시작되었으며 1:1, 1:N, N:M(다대다) 관계가 지원되어 효과적이고 빠른 데이터 추출이 가능해졌다.  
하지만, 계층형과 마찬가지로 매우 복잡한 내부 포인터를 사용하고 프로그래머가 이 모든 구조를 이해해야만 프로그램의 작성이 가능하다는 단점이 여전히 존재한다.

**관계형 DBMS**

관계형 DBMS의 시초는 Codd라는 학자가 수학 모델에 근거해서 고안하면서 시작되었다.  
RDBMS의 핵심 개념은 “데이터베이스는 테이블이라 불리는 최소 단위로 구성되어 있다. 그리고 이 테이블은 하나 이상의 열로 구성되어 있다.”라고 생각하면 된다.

\* 테이블을 부르는 다른 용어로 릴레이션(Relation), 엔티티(Entity) 등이 있다.

테이블은 데이터를 효율적으로 저장하기 위한 구조이다.  
정보를 저장하기 위해 여러 개의 테이블로 나누어서 저장함으로써 불필요한 공간의 낭비를 줄이고 데이터 저장의 효율성을 보장해 줄 수 있다. 또, 테이블의 관계를 기본 키(Primary Key)와 외래 키(Foreign Key)를 사용해 두 테이블을 부모와 자식 관계로 묶어 줄 수 있다.  
추후에 부모와 자식 테이블을 조합해서 결과를 얻고자 할 경우에는 SQL(Structured Query Language)의 조인(JOIN) 기능을 이용하면 된다.

관계형 DBMS의 ***장점***은 다른 DBMS에 비해서 업무가 변화될 경우 쉽게 변화에 순응할 수 있는 구조이며 유지보수 측면에서도 편리한 특징을 가지고 있다.  
또한, 대용량 데이터의 관리와 데이터 무결성의 보장을 잘 해주기 때문에 동시에 데이터에 접근하는 응용 프로그램을 사용할 경우에 좋다.  
이와 반대로 RDBMS의 가장 큰 ***단점***은 시스템 자원을 많이 차지해서 시스템이 전반적으로 느려진다.  
그러나, 최근 들어 하드웨어의 급속한 발전으로 이러한 단점이 많이 보완되고 있다.

================================================================

1.1.4 SQL 개요

**DBMS 제작 회사와 독립적이다.**

**다른 시스템으로 이식성이 좋다.**

어느곳에서 사용된 SQL을 다른 시스템으로 이식하는데 별 문제가 없다.

**표준이 계속 발전한다.**

**대화식 언어이다.**

기존 프로그래밍 언어는 프로그램 작성, 컴파일 및 디버깅, 실행이라는 과정을 거쳐야만 그 결과를 확인할 수 있었으나 SQL은 이와 달리 바로 질의하고 결과를 얻는 대화식 언어로 구성되어 있다.

**분산형 클라이언트/서버 구조이다.**

클라이언트에서 질의하면 서버에서 그 질의를 받아 처리한 후, 다시 클라이언트에게 전달하는 구조를 가진다.

이때 주의할 점은 모든 DBMS의 SQL문이 완벽하게 동일하지 않다는 점이다. 이를 Oracle에서는 PL/SQL이라고 부르고, SQL Server에서는 T-SQL, MySQL에서는 그냥 SQL로 부른다.

================================================================

1.2 MySQL 소개