제 1 장
데이터베이스 개요

목차

데이터베이스 란

“ 어느 특정 조직의 응용 업무에 공동 사용하기 위하여 운영상 필요한 데이터를 **완벽화, 비 중복화, 구조화**하여 컴퓨터 기억 장치에 저장한 데이터의 집합체”

“발생한 데이터를 통괄적인 관점에서 서로 연관된 정보의 **중복을 최소화**하여 한곳에 모아 저장함으로써 다수의 사용자로 하여금 필요한 정보를 공유하도록 한 정보의 집합체”

컴퓨터와의 대화

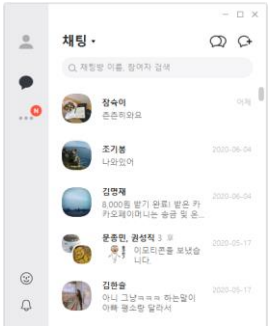
- 초고도 정보화 사회이다.
- 패스트푸드점에서 햄버그를 주문하면 POS에서 모든 것이 처리된다.

- 스마트폰으로 모든 정보를 얻을 수 있다.

게시판

검색엔진

주식거래



메신저

인터넷 신문

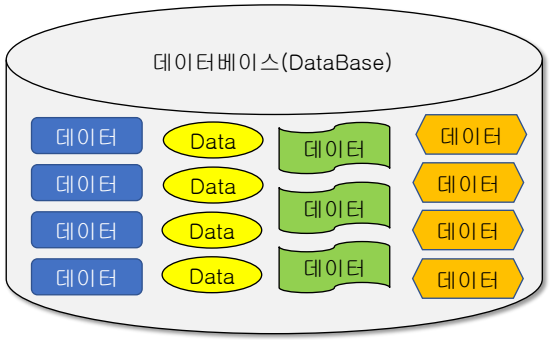
- 이런 모든 것이 가능해진 기술적 배경이 바로 데이터베이스이다.

1.1 데이터베이스의 특징

- 물리적·논리적 데이터 독립을 지원한다.
- 중복을 최소화 자료의 불일치성을 피할 수 있다.
- 데이터를 공유(sharing)할 수 있다.
- 정보를 표준화(standardization)하여 저장한다.
- 보안성(security)을 제공한다.
- 무결성(integrity)이 유지된다.
- 상충되는 요구를 조절한다.

데이터베이스의 요건

- 데이터베이스(DataBase)는 컴퓨터의 기억 능력을 십분 활용하여 자료를 가공 및 저장, 활용하는 일체의 기술이다.



- 자료(Data)를 활용하려면 합산, 집계 등의 알고리즘을 적용하여 정보(Information)로 가공해야 한다.

자료(Data)

학생	국어	영어	수학
1번	85	92	76
2번	100	80	86
3번	88	75	82
⋮			

집계, 가공 →

정보(Information)

학생	총점	평균	석차
1번	253	84.3	2
2번	266	88.6	1
3번	245	81.6	3
⋮			

데이터베이스의 요건

- **현대적인 데이터베이스가 갖추어야 할 요건**
  - 대용량
  - 효율성
  - 무결성
  - 활용성
  - 공유성
  - 보안성
- **거대한 데이터를 저렴하고 완벽하게 저장하며 언제든 누구나 활용할 수 있으면서도 안전해야 한다.**
- **고도의 기술을 요하며 수십년에 걸친 연구, 개발이 필요하다.**
- **데이터베이스 시스템은 비싸고 그 기술을 다 배우고 제대로 활용하는 것도 만만치 않다.**

1.2 데이터베이스의 시스템

●데이터베이스
- ➢중앙집중식 데이터베이스, 분산식 데이터베이스

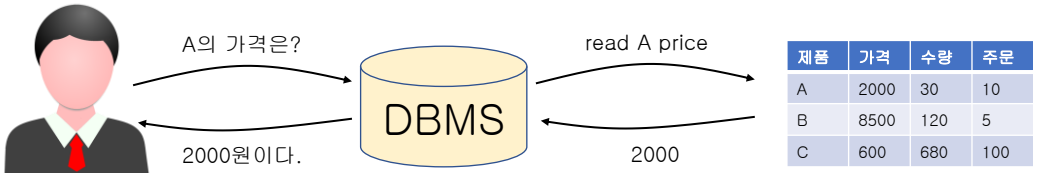
●하드웨어
- ➢범용컴퓨터, 전용컴퓨터

●소프트웨어
- ➢데이터베이스 관리 시스템(DBMS) = DB = 대용량 솔루션 = OracleDB

●사용자
- ➢데이터베이스 관리자(database administrator)
- ➢응용프로그래머(application programmer)
- ➢단말 사용자(end user)

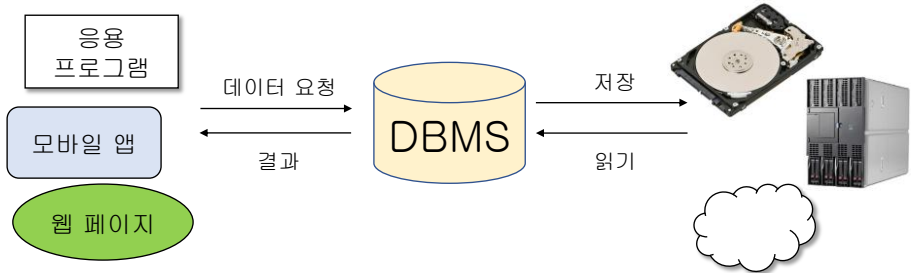
DBMS

- DBMS(DataBase Management System : 데이터베이스 관리 시스템)는 정보의 저장과 관리를 전담하는 특수한 소프트웨어



제품	가격	수량	주문
A	2000	30	10
B	8500	120	5
C	600	680	100

- 데이터 관리를 DBMS가 전담하면 프로그램과 데이터가 분리되고 종속성이 제거된다.
- 약간의 성능 저하가 발생하는 대신 유연성이 극적으로 증가한다.
- DBMS에 의해 응용 프로그램과 데이터가 다:다 양쪽으로 연결된다.



- 요구가 더 복잡해져 응용 프로그램이 해야 할 잡스러운 작업까지 대신한다.

1.3 관계 데이터베이스 관리 시스템(RDBMS)

●기본개념

- 관계데이터 모델은 데이터베이스를 테이블들의 집합으로 나타내며 테이블은 행과 열로 이루어져 있다. 테이블의 각 행(row)은 특정 목적에 따라 연관된 데이터 값들로 구성된다. 테이블간의 관계는 공통 열(column)을 통해서만 이루어지며 어떠한 링크도 존재하지 않는다.

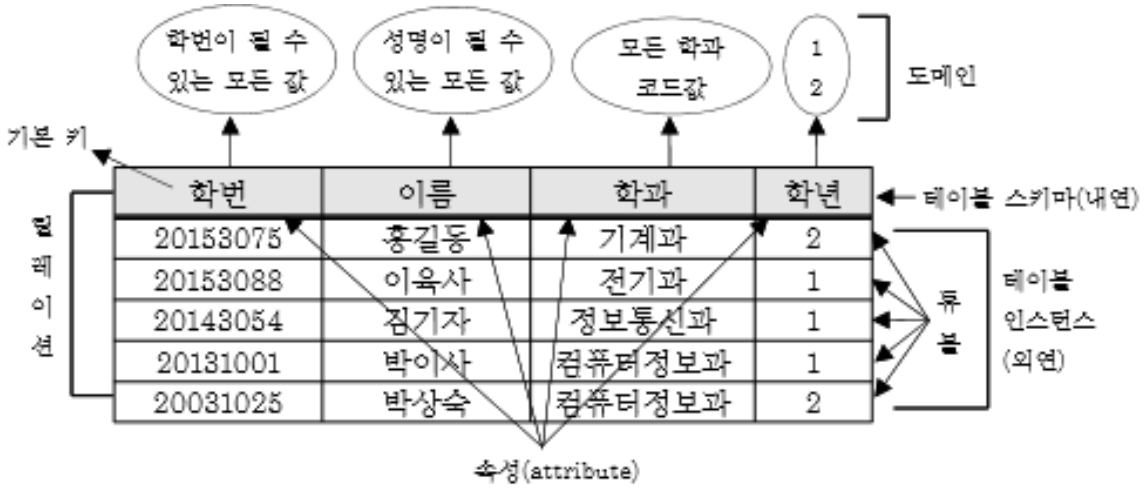


그림 1.2 관계데이터 모델의 예제

1.3 관계 데이터베이스 관리 시스템(RDBMS)

- 릴레이션(relation) = table
  - 중앙집중식 데이터베이스, 분산식 데이터베이스

- 튜플(tuple) = record
  - 테이블의 한 행(row)을 구성하는 <속성 이름, 값>쌍들의 집합으로 물리적인 용어로는 레코드(record)와 같다.

- 속성(attribute) = domain = field(item)
  - 테이블의 각 열(column)을 의미하며 레코드 구조의 필드에 대응된다. 또한 속성(attribute)은 릴레이션이 갖는 성질(property)을 의미하며 관계데이터 모델에서 데이터의 가장 작은 논리적인 단위이다.

1.3 관계 데이터베이스 관리 시스템(RDBMS)

- **도메인(domain)**
  - 각 속성이 취할 수 있는 값의 집합으로 같은 타입이어야 한다.

- **단순 도메인(simple domain)**
  - 위에 정의된 도메인을 단순 도메인이라 하고, 이 단순 도메인으로 정의된 속성을 단순 속성이라 한다.

- **복합 도메인(composite domain) => join, subquery**
  - 단순 도메인들을 결합하여 구성된 도메인을 복합 도메인이라 하고, 이 복합 도메인으로 정의된 속성을 복합 속성이라 한다.

1.3 관계 데이터베이스 관리 시스템(RDBMS)

- **릴레이션 스키마(schema)**
  - 데이터베이스에 저장될 자료들의 논리적 구조 및 관계를 의미하고 릴레이션 이름과 속성들 그리고 스키마의 무결성 제약 조건으로 구성되며 시간에 무관한 정적 성질을 갖는 릴레이션의 영구 부분. 다른 이름으로 엔티티 유형(type), 릴레이션 유형, 릴레이션 내포(intension)라고도 부른다.

- **릴레이션 인스턴스(instance)**
  - 어느 한 시점에서 릴레이션이 포함하고 있는 전체 튜플을 의미하며 시간에 따라 변화하는 동 적 성질을 갖고 있다. 엔티티 집합, 릴레이션 상태(state), 릴레이션 어커런스(occurrence), 릴레이션 외연(extension), 단순히 릴레이션이라고도 부른다.

- **릴레이션의 차수(degree)**
  - 한 릴레이션을 구성하는 속성 수를 의미한다.

- **카디널리티(cardinality)**
  - 특정 테이블의 튜플 개수를 의미한다.

1.3 관계 데이터베이스 관리 시스템(RDBMS)

릴레이션(table)의 특징

- **튜플의 유일성**
  - 릴레이션의 인스턴스는 튜플들의 집합이고 집합은 중복된 원소를 포함하지 않으므로 릴레이션에는 **중복된 튜플이 존재하지 않는다.**

- **튜플의 무순서(위에서 아래로)**
  - 릴레이션에 있는 **튜플들의 순서는 의미가 없다.**

- **속성의 무순서(왼쪽에서 오른쪽으로)**
  - 릴레이션의 **속성 사이의 순서는 의미가 없다.**

- **속성의 원자 값**
  - **모든 속성의 값은 원자 값이다.**

1.3 관계 데이터베이스 관리 시스템(RDBMS)

키(key)의 종류

- **슈퍼 키(super key)**
  - 테이블의 각 튜플을 유일하게 식별 할 수 있는 속성들의 조합으로 이루어진 키를 슈퍼 키라 한다. 두 개 이상의 속성으로 구성된 슈퍼 키 중에는 어떤 속성을 제거하더라도 각 튜플을 유일하게 식별 할 수 있다.
- **후보 키(candidate key)**
  - 속성 집합으로 구성된 테이블의 각 튜플을 유일하게 식별할 수 있는 속성이나 속성의 조합 들을 테이블의 후보 키라 하며. 속성의 조합으로 구성될 경우 어느 한 속성을 제거하면 튜 플의 유일성을 잃게 되는 점이 슈퍼 키와 다르다.
- **기본 키(primary key)**
  - 한 테이블 내에서 각 튜플을 유일하게 식별 할 수 있는 속성 또는 속성의 조합으로 구성된 키로 후보 키 중 하나가 선택된다. 일반적으로 적은 개수의 속성으로 된 후보 키를 기본 키로 선택하는 것이 좋다.

1.3 관계 데이터베이스 관리 시스템(RDBMS)

●대체 키(alternate key)

- 후보 키가 하나 이상일 때 그 중 하나를 기본 키로 지정하면 나머지 후보 키들은 대체 키가 된다.

●외래 키(foreign key)

- 릴레이션 스키마 R의 어떤 속성이나 속성집합(외래 키(FK)라 가정)이 다른 릴레이션 스키마 S의 기본 키가 될 때, R의 속성 또는 속성집합을 외래 키라 한다, 이때 외래 키는 테이블 S를 참조(reference)한다고 하고, R에 있는 튜플들의 외래 키값은 S의 어떤 기본 키 값과 일치하거나 널을 가져야 한다 .

1.3 관계 데이터베이스 관리 시스템(RDBMS)

제약조건

- **도메인 제약 조건**
  - 각 속성 값은 반드시 해당 도메인에 속하는 원자 값이어야 한다는 조건
- **키 제약 조건**
  - 릴레이션에는 릴레이션의 각 튜플을 유일하게 식별할 수 있는 수단 즉, 최소한 하나의 기본 키를 가지고 있어야 한다는 제약조건.
- **무결성 제약조건**
  - 엔티티 무결성(entity integrity)
    - 기본 키에 속해 있는 속성들의 값은 어떠한 경우에도 널 값을 가질 수 없다는 의미로서 만약 널 값을 갖게 되면 튜플을 유일하게 식별할 수 없게 된다.
  - 참조 무결성(referential integrity)
    - 한 테이블에 있는 튜플이 다른 테이블에 있는 튜플을 참조하려면 반드시 참조되는 튜플 이 그 테이블 내에 존재해야 한다는 의미로 외래 키와 관련되어 있다.

1.4 데이터 모델과 데이터베이스의 발전사

●데이터모델

- 현실세계의 정보들 을 컴퓨터에 표현하기 위해 단순화, 추상화 형태로 체계적으로 표현할 수 있는 개념적 모형, 또한 데이터 모델을 이용하여 현실세계의 정보구조를 표현하려는 작업을 데이터 모델링(data modeling)이라고 한다.

●계층형 데이터 모델, 네트워크(망)형 데이터 모델, 관계형 데이터 모델, 객체지향형 데이터 모델 등이 있다.

●최근에는 관계형 데이터베이스 관리시스템과 객체 지향 데이터 모델의 장점을 기반으로 한 객체관계형 데이터베이스 관리 시스템(object-relational database management system)이 시장을 형성해 가고 있다.

1.4 데이터 모델과 데이터베이스의 발전사

●파일시스템

➢응용 프로그램에서 여러 파일을 다룬다면 프로그램에서 각각 의 파일과 연결 하여 처리하여야 함.

●데이터베이스 => Lotus1,2,3 => MSSQL, MySQL Oracle

➢기존의 파일 시스템에 비해 훨씬 효과적으로 데이터를 관리, 운용

➢파일 시스템과는 다르게 데이터베이스 관리시스템과의 연결 하나만으로 데이터베이스 내의 모든 데이터에 작업을 할 수 있다.

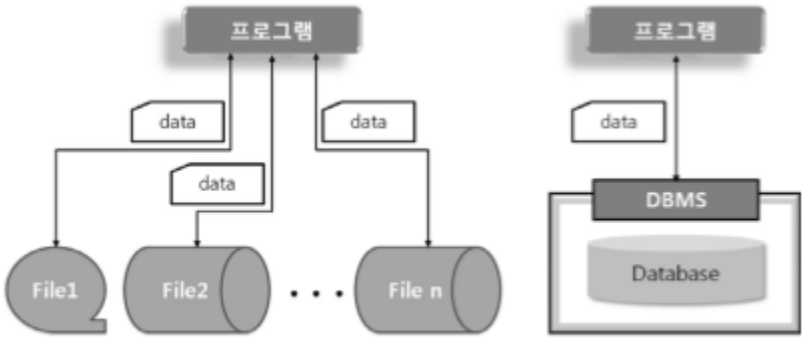
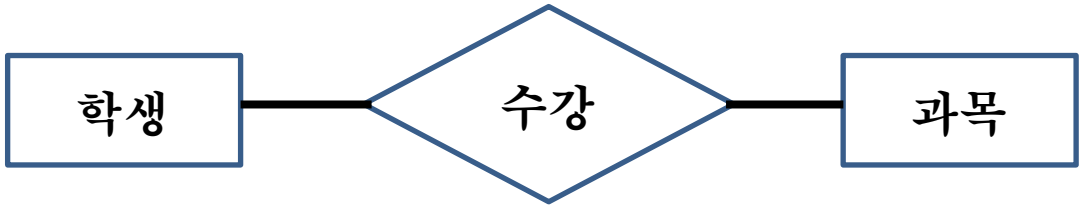


그림 1.3 데이터베이스와 파일시스템의 차이

1.4 데이터 모델과 데이터베이스의 발전사

●관계형 모델(Relation Model)

➢ 1970년 E.F.Codd에 의해 제안, 데이터를 2차원 테이블 형태로 저장



학번	이름	학과	학년
20153075	옥한빛	기계	1
20153088	이태연	기계	1
20143054	유가인	기계	2
20152088	조민우	전기전자	1
20142021	심수정	전기전자	2
20132003	박희철	전기전자	3
20151062	김인중	컴퓨터정보	1
20141007	진현무	컴퓨터정보	2
20131001	김종헌	컴퓨터정보	3
20131025	옥성우	컴퓨터정보	3

과목번호	학번	점수
101	20131001	80
104	20131001	56
106	20132003	72
103	20152088	45
101	20131025	65
104	20131025	65
108	20151062	81
107	20143054	41
102	20153075	66
105	20153075	56
102	20153088	61
105	20153088	78

과목	과목이름	교수명	학년	학과
111	데이터베이스	이재영	2	컴퓨터정보
110	자동제어	정순정	2	전기전자
109	자동화설계	박민영	3	기계
101	컴퓨터개론	강종영	3	컴퓨터정보
102	기계공작법	김태영	1	기계
103	기초전자실험	김유석	1	전기전자
104	시스템분석설계	강석현	3	컴퓨터정보
105	기계요소설계	김명성	1	기계
106	전자회로실험	최영민	3	전기전자
107	CAD응용실습	구봉규	2	기계
108	소프트웨어공학	권민성	1	컴퓨터정보

학생 테이블

```
create   table   student(  --student 라는 테이블 생성
    stu_no            number(8),  --학번, 숫자로 8자리 까지 입력 가능
    stu_name   varchar2(12) not null,  --이름, 문자열로 12자리 까지 입력가능인데, 반드시 입력( not null) 해야 합니다.
    stu_dept   VARCHAR(20) not null,  --학과
    stu_grade   NUMBER(1) not null,   --학년
    stu_class    CHAR(1)     not null,        --반
    stu_gender   CHAR(1) not null,    --성별
    stu_height   NUMBER(5,2),          --키
    stu_weight  NUMBER(5,2)not null,  --몸무게
    constraint stu_no_pk primary key(stu_no)  --제약조건, 학번을 PK로 사용합니다.
);
```

```
-- 학생 테이블에 데이터를 입력 :
 학번      이름     학과    학년  반  남성    키   몸무게
INSERT  INTO  student  VALUES(20153075,  '옥한빛',  '기계',   1, 'C',   'M',   177,  80);
INSERT INTO   student  VALUES(20153088, '이태연',   '기계',   1, 'C',   'F',    162, 50);
INSERT INTO   student  VALUES(20143054, '유가인',   '기계',   2, 'C',   'F',    154, 47);
INSERT INTO   student  VALUES(20152088, '조민우',   '전기전자', 1, 'C',  'M',  188,  90);
INSERT INTO   student  VALUES(20142021, '심수정',   '전기전자', 2, 'A',  'F',  168,  45);
INSERT INTO   student  VALUES(20132003, '박희철',   '전기전자', 3, 'B',  'M',  null,  63);
INSERT INTO   student  VALUES(20151062, '김인중',   '컴퓨터정보', 1,  'B',  'M',166,  67);
INSERT INTO   student  VALUES(20141007, '진현무',   '컴퓨터정보', 2,  'A',  'M',174,  64);
INSERT INTO   student  VALUES(20131001, '김종헌',   '컴퓨터정보', 3,  'C',  'M', null, 72);
INSERT INTO   student  VALUES(20131025, '옥성우',   '컴퓨터정보', 3,  'A',  'F', 172,  63);

select  *  from  student; --10
```

과목 테이블

```
CREATE   TABLE   subject(
sub_no    CHAR(3),   ---과목번호
sub_name    VARCHAR2(30) NOT NULL, --과목이름
sub_prof       VARCHAR2(12) NOT NULL, --교수이름
sub_grade   NUMBER(1)          NOT NULL, --학년
sub_dept     VARCHAR2(20)   NOT NULL, --학과
CONSTRAINT sub_no_pk PRIMARY KEY(sub_no) --제약조건, PK(과목번호)
);
```

```
INSERT INTO subject VALUES('111',   '데이터베이스',  '이재영',  2,  '컴퓨터정보');
INSERT INTO subject VALUES('110',   '자동제어',  '정순정',  2,  '전기전자');
INSERT INTO subject VALUES('109',   '자동화설계',  '박민영',  3,  '기계');
INSERT INTO subject VALUES('101',   '컴퓨터개론',  '강종영',  3,  '컴퓨터정보');
INSERT INTO subject VALUES('102',   '기계공작법',   '김태영',  1,  '기계');
INSERT INTO subject VALUES('103',   '기초전자실험',  '김유석',  1,  '전기전자');
INSERT INTO subject VALUES('104',   '시스템분석설계',  '강석현',  3,  '컴퓨터정보');
INSERT INTO subject VALUES('105',   '기계요소설계',   '김명성',  1,  '기계');
INSERT INTO subject VALUES('106',   '전자회로실험',   '최영민',  3,  '전기전자');
INSERT INTO subject VALUES('107',   'CAD응용실습',   '구봉규',  2,  '기계');
INSERT INTO subject VALUES('108',   '소프트웨어공학',  '권민성', 1,  '컴퓨터정보');

select * from subject; -- 11
```

수강테이블

```
CREATE TABLE enrol(
sub_no  CHAR(3)
  CONSTRAINT enr_sub_no_fk REFERENCES subject(sub_no),
stu_no  number(8)
 CONSTRAINT enr_stu_no_fk REFERENCES student(stu_no),
enr_grade  NUMBER(3),
 CONSTRAINT enr_pk PRIMARY KEY(sub_no, stu_no)
 );
```

```
insert into enrol values('101',  20131001,  80);
insert into enrol values('104',  20131001,  56);
insert into enrol values('106',  20132003,  72);
insert into enrol values('103',  20152088,  45);
insert into enrol values('101',  20131025,  65);
insert into enrol values('104',  20131025,  65);
insert into enrol values('108',  20151062,  81);
insert into enrol values('107',  20143054,  41);
insert into enrol values('102',  20153075,  66);
insert into enrol values('105',  20153075,  56);
insert into enrol values('102',  20153088,  61);
insert into enrol values('105',  20153088,  78);

select  *  from enrol; --12
```

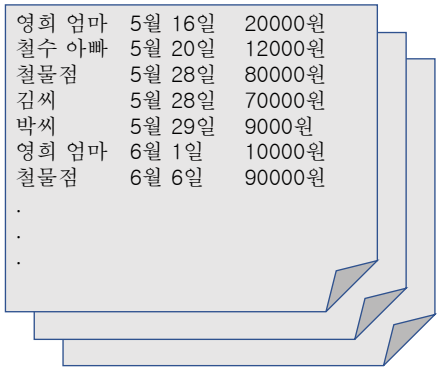
1.4 데이터 모델과 데이터베이스의 발전사

●오라클

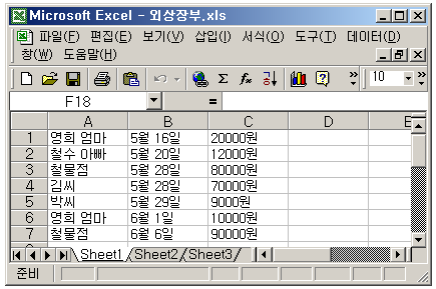
- ➢가장 많이 쓰이는 데이터베이스 관리시스템이다.
- ➢1983년부터 트랜잭션 처리기능, 데이터의 일관성 처리 기능, 분산 처리 기능을 포함한 오리클 버전 3, 4, 5를 출시
- ➢1989년 PL/SQL 기능을 포함한 오라클 버전6 출시
- ➢1993년 오라클 버전7 참조무결성 기능, 저장 프로시저(stored procedure) 기능, 트리거 처리기능의 포함
- ➢1999년 객체지향 기능을 포함한 객체관계형 데이터베이스 관리시스템인 오라클 8버전이 출시, 추 후 다양한 기능이 보완된 오라클 8i 버전 출시
- ➢오라클 9i, 오라클 10g를 거쳐, 2007년에는 오라클 11g 버전을 출시
- ➢클라우드 컴퓨팅의 개념을 포함한 12c를 2013년 에 발표

역사

- 가장 고전적인 방법은 종이에 적어 놓는 것이다. 검색이나 수정은 어려워 활용성이 떨어진다.



- 파일 시스템 : 메모장에 텍스트 형태로 기록하거나 엑셀을 사용한다. 저장할 뿐 관리 능력이 미흡하고 여러 사람이 동시에 사용할 수 없다.



역사

- SAM은 정보를 파일에 순서대로 쟁여 놓는 방법이며 **순차 검색만 가능하다.**
- ISAM은 정보의 순서를 기억하는 인덱스가 있어 **이분 검색이 가능하다.**

이름	나이	주소
이승먼	58	미국
박쟁희	52	서울
전두한	36	부산
노때우	24	광주
김공삼	40	대전
김태중	45	춘천
노무헌	48	인천
이멍박	36	전주
박그네	32	제주
문대인	56	서울

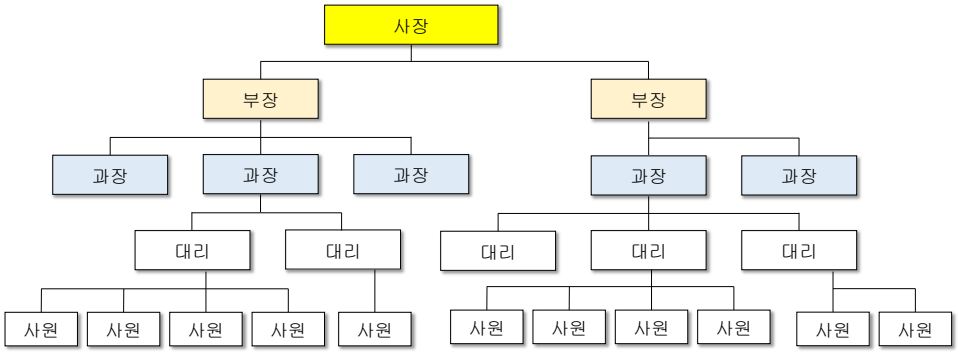
SAM

인덱스
1
2
3
4
5
6
7
8
9
10

이름	나이	주소
이승먼	58	미국
박쟁희	52	서울
전두한	36	부산
노때우	24	광주
김공삼	40	대전
김태중	45	춘천
노무헌	48	인천
이멍박	36	전주
박그네	32	제주
문대인	56	서울

ISAM

- 계층형, 네트워크형 : 복잡도에 비해 실용성이 떨어져 특수한 분야에만 가끔 사용한다.



역사

- 관계형 데이터베이스(Relational Database)는 1969년 Edgar.F.Codd 박사의 논문을 기반으로 탄생
- 모든 데이터를 표 형태의 테이블에 저장하며 관계를 정의한다.

이름	부서	입사일	월급
김상영	영업부	2015.6.25	620
김영주	영업부	2017.4.13	540
문병대	인사과	2012.3.1	550
김규민	오락부	2019.8.1	340
문여울	인사과	2018.9.12	490
문한울	오락부	2020.10.25	430

부서명	책임자	사무실
영업부	구홍녀	A동 3층
인사과	김정수	B동 2층
오락부	김경선	A동 1층

- 유연하여 로직 변화에 신속히 대처할 수 있다. 자원 소모가 많으며 고성능 하드웨어와 고급 튜닝 인력이 필요하다.
- 객체지향형 : 모든 것을 객체로 저장한다. 성능상의 열세를 극복하지 못했고 실용적으로 활용할 분야가 드물어 아직 연구 단계이다.
- 빅데이터 : 현대의 데이터는 대용량(Volume)인데다 형태도 다양하고(Variety) 생성 주기가 빨라(Velocity) RDB로는 수집, 저장, 분석이 어려운 지경에 이르렀다.
- 일관성을 약간 희생하더라도 성능과 용량을 극적으로 향상시킨 빅데이터 기법이 대두되었다. Hadoop 솔루션, 비정형 데이터를 다루는 NoSQL(Not Only SQL) 문법, 데이터마이닝이나 인공지능, 딥러닝을 활용한 분석

SQL 종류

- DBMS의 종류는 우리가 늘상 먹는 라면만큼이나 다양하다.

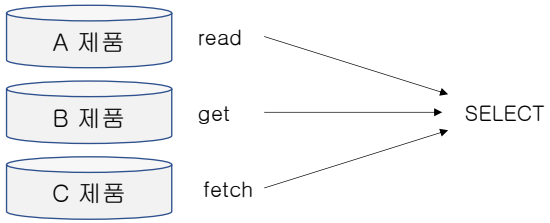
- Oracle : 기능적 완성도가 높고 시장 점유율도 높다. 지존
- SQL Server : 윈도우 환경에서 탁월한 성능을 발휘하며 닷넷과의 통합성도 우수하다.
- MySQL : 오픈소스, 다양한 운영체제를 지원하며 표준 SQL을 준수하여 학습용으로 적합하다. PHP와 궁합이 잘 맞아 웹 게시판 제작 용도로 많이 활용한다.
- DB2 : IBM 제품. 오라클 다음으로 시장 점유율 2위를 기록하고 있다.
- PostgresQL : 오픈소스이며 객체 관계형 DBMS이다. 일본에서 인기가 많다.

순위	DBMS	점수
1	오라클	1350
2	MySQL	1270
3	MS SQL Server	1070
4	PostgreSQL	522
5	MongoDB	437
6	IBM DB2	161

- 부동의 1위는 오라클이고 MSSQL, MySQL, DB2가 각축을 벌이고 있다.

표준어

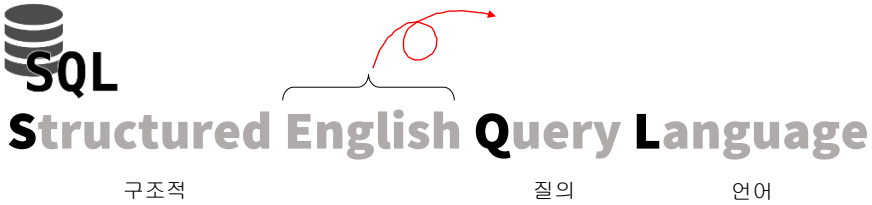
- **초창기의 데이터베이스 제품은 제조사마다 구조가 독특해 관리 방법이 제각각이었다. 데이터를 읽는 기본적인 명령조차 제품마다 달랐다.**



- **제품을 바꿀 때마다 다시 배워야 한다. 사회적 낭비를 초래한다.**
- **공통적인 표준 언어로 탄생한 것이 SQL이다. 제품마다 구조나 관리 방식이 달라도 SQL을 통하면 똑같은 방법으로 다룰 수 있다.**
  - 키가 170 이상인 회원의 목록을 조사하라.
  - 이번달 매출 합계와 평균을 조사하라.
  - 모든 상품의 가격을 20% 인하한 값으로 변경하라.
- **자연어로 되어 있지는 않으며 고유의 문법 체계를 가지고 있다.**
  - SELECT * FROM tMember WHERE height >= 170;
- **SQL 문도 의사 소통을 위한 언어라는 면에서 자연어와 동질성이 있다.**

SQL의 역사

- SQL의 시초는 1970년에 IBM의 시스템 R에서 도입한 SEQUEL(Structured English Query Language)
- 상표권 분쟁으로 인해 SQL(Structured Query Language)로 이름 변경
- 쿼리(Query)는 DBMS에게 요청한다는 뜻이며 한국말로는 "질의"로 번역한다.



- 일시적인 통일을 이룬 후 경쟁에 의해 방언(Dialect)이 생기기 시작했다.
- 공신력 있는 국제 표준 단체에서 SQL의 표준을 만들고 관리한다.

SQL 표준	특징
SQL86	ANSI에서 제정한 최초의 표준
SQL92	대규모 개정 및 정리. 실질적인 첫 표준
SQL99	정규 표현식, 트리거, 절차적 흐름.
SQL2003	XML 관련 기능 추가. 시퀀스 생성기. MERGE 구문 추가
SQL2008	INSTEAD OF 트리거 추가. TRUNCATE 구문 추가
SQL2011	임시 데이터베이스 지원
SQL2016	JSON 지원. 행 패턴 인식. DECFLOAT 타입 추가

- SQL99가 실질적인 표준이다. 제조사마다 확장 문법을 제공한다.

■■■ JDBC(Java DataBase Connectivity) ■■■

1. JDBC(Java DataBase Connectivity)는 자바 프로그램이 DBMS에 일관된 방식으로 접근할 수 있도록 API를 제공하는
   자바 클래스들의 모임으로 다음의 특징을 가진다.

   1) JDBC는 함수 호출용 SQL 인터페이스
   2) JDBC는 ANSI SQL-92 표준을 지원
   3) JDBC는 공통된 SQL 인터페이스를 바탕
   4) JDBC는 배우고 사용하기 쉽다(?)

--> JDBC란 데이터베이스에 연결 및 작업을 하기 위한 자바 표준 인터페이스이다.

2. JDBC 구성

1) 응용 프로그램
- 데이터베이스에 연결을 요청
- 데이터베이스에 SQL문을 전송
- SQL문의 결과를 요청
- 오류가 발생하는 경우에 오류 처리
- 트랜잭션을 제어
- 연결 종료

2) 드라이버 매니저
- 데이터베이스에 맞는 드라이버 검색
- JDBC 초기화를 위한 작업을 수행

3) 드라이버
- 데이터베이스에 연결
- 데이터베이스에 SQL문을 전달
- 응용프로그램에 검색 결과를 전달
- 필요한 경우 커서를 조작
- 필요한 경우 트랜잭션을 시작

4) DBMS
- 데이터가 저장되어 있는 장소

3. System Architecture

JDBC API 는 2-tier 와 3-tier 를 모두 지원한다.

1) 2-tier

- 자바 애플릿이나 어플리케이션이 JDBC를 이용하여 DBMS 데이터베이스에 직접 접근
- 가장 대표적인 클라이언트/서버 구조
- 프로그래밍이 간단하다는 장점
- 보안, 로드밸런싱, 확장성(오브젝트 재사용) 등의 문제점

- 2-tier 디자인이 적합한 경우

  가. 애플리케이션이 하나의 데이터베이스만을 사용

  나. 데이터베이스 엔진이 하나의 CPU에서 동작

  다. 데이터베이스가 계속 거의 같은 크기로 유지

  라. 사용자 기반이 같은 크기로 유지

  마. 요구가 확정되어 변화 가능성이 극히 적거나 없는 경우

  바. 애플리케이션을 종료한 후에도 최소한의 지속성을 요구

2) 3-tier

- 자바 어플리케이션이나 애플릿이 DBMS에 직접 접근하는 것이 아니라, 중간에 미들웨어(미들티어)를 거쳐 데이터베이스에 접근

- 데이터베이스와의 연동 부분을 분리시킴으로써 Presentation Layer가 데이터 저장 방법에 신경을 쓰지 않아도 되는 구조

- 클라이언트는 단지 미들티어를 참조

- 미들 티어 서버는 DBMS와 같이 특정한 작업을 수행하는 최종 서버와 통신을 하여 결과를 얻은 후 클라이언트에 결과를 전달

- 2-tier 모델보다 안정적이고 유연하며 보안이 강화

가. tier-1 : 사용자 인터페이스를 담당하는 클라이언트

나. tier-2 : HTTP나 코바를 지원하는 응용처리 서버

다. tier-3 : DBMS와 같이 사용자가 최종적으로 원하는 기능을 수행할 서버

4. JDBC Driver 유형

1) Type 1

- JDBC-ODBC Bridge
- 특징

가. 데이터베이스를 연동하기 위해 브릿지 기술을 사용

나. ODBC API로의 게이트웨이를 제공하여 실제로는 ODBC의 API를 구현함으로써 데이터베이스를 연동

다. 브릿지 솔루션은 보통 클라이언트에 소프트웨어가 설치될 것을 요구

라. JDBC-ODBC 드라이버는 ODBC 드라이버가 풍부하기 때문에 거의 대부분 데이터베이스 시스템에서 사용할 수 있으며, JDBC-ODBC 드라이버를 사용하는 클라이언트에 사전에 ODBC 드라이버가 설치되어 있는 경우 매우 유용하게 사용할 수 있다.

마. 속도와 관련한 가장 큰 문제

JDBC를 통해 호출된 명령이 다시 ODBC를 통해 나가야 하기 때문에 두 개의 브릿지를 거치며,

이로인해 빠른 속도를 기대하기 어렵다.

빠른 성능을 요구하는 애플리케이션의 경우에는 Type 1. JDBC-ODBC 브릿지는 적당하지 않다.

바. JDBC-ODBC 브릿지를 사용하는 시스템에는 반드시 해당 데이터베이스에 연결하기 위한 ODBC 드라이버가 설치되어야 하며, 이 단점은 애플릿에서 JDBC를 사용하여 프로그래밍 하는 경우, 많은 문제가 된다.

애플릿을 사용하여 JDBC-ODBC를 사용할 경우, 애플릿을 다운받은 클라이언트에 미리 해당 ODBC 드라이버가 설치되어 있어야 하기 때문에 배포 등에 많은 문제가 발생하게 된다.

2) Type 2

- Native-API / Partly Java Driver(사용 일부 자바)
- 특징

가. 각각의 데이터베이스 제조업체들이 제공한 C 혹은 C++ 메소드를 자바 코드가 호출하는 방식

나. 부분적으로 자바 드라이버인 원시 API라고 일컬어짐

다. 데이터베이스와 연결되는 부분이 Native Code로 구현되어 있는 만큼 JDBC-ODBC 브릿지에 비해 빠른 속도를 제공한다.

라. JDBC 드라이버를 사용하고자 하는 각각의 클라이언트에 DBMS Vendor 의 데이터베이스 라이브러리가 로드되어야 하기 때문에 인터넷이나 CS 환경에서는 사용하기 적합하지 않다. 또한 Type 3, 4 드라이버에 비해 낮은 성능

3) Type 3

- Net-Protocol / All-Java Driver(순수 자바)
- 클라이언트에서 일반적인 Network API를 이용해 보낸 정보를 서버가 Database에 독점적인 형태로 변환하는 방식
- 특징

가. 클라이언트에 존재하는 JDBC 드라이버는 소켓을 사용하여 서버에 존재하는 미들웨어(Middleware) 애플리케이션에 연결

나. 애플리케이션은 클라이언트의 요청을 사용하고자 하는 데이터베이스에 독점적인 API로 전환

다. 하나의 드라이버로 여러 개의 데이터베이스를 연동

라. 클라이언트에 소프트웨어를 설치할 필요가 없음

4) Type 4

- Native-Protocol / All-Java Driver

- Databae Engine 에 사용되는 Network Protocol을 자바 소켓으로 직접 Database에 교신하는 방식

- 특징

가. 가장 직접적인 순수 자바 솔루션

나. 거의 대부분 데이터베이스의 제조업체가 제공한다.

다. ODBC나 Native Lib 형태로 request 를 변환하지 않기 때문에 Performance가 매우 좋다. 또한 특별하게 Driver 나 Lib, Middleware 등을 설치할 필요가 없기 때문에 배포 등이 매우 용이하다.

5. Oracle JDBC Driver

1) Type 2

오라클 "OCI Driver"로 불리운다.

ex) OCI10 Driver : Oracle 10 데이터베이스를 지원

OCI11 Driver : Oracle 11 데이터베이스를 지원

...

이러한 드라이버는 한정된 플랫폼으로 모든 오라클 드라이버는 JDK 1.0 과 1.1x 에 종속적이며 JDBC 1.22 표준을 지원한다.

또한 Net11을 포함하는 Oracle Client 를 설치해야 사용이 가능하다.

2) Type 4 ★

오라클 "Thin Driver"로 불리운다.

자바로 작성된 Net8의 TCP/IP 버전의 자체적인 실행을 포함하며 플랫폼에 독립적이고 실행시간에 브라우저로 다운로드 된다.

그리고 서버 측에서는 TCP/IP Listener가 필요하며 연결 스트림은 TNSNAMES 엔트리가 아닌 TCP/IP 주소와 포트 번호이다.

URL Format : jdbc:oracle:thin:@[host]:[port]:[database]

6. JDBC API

JDBC API는 데이터베이스의 데이터를 액세스할 수 있도록 제공되는 표준 자바 API 클래스와 인터페이스로 구성되며

다음의 기능을 실행하기 위한 클래스와 인터페이스를 제공한다.

- Java 프로그램에서 데이터베이스 서버에 접속
- SQL문을 구성, 데이터베이스 서버에서 실행
- 데이터베이스 서버가 처리한 결과 가져오기
- 데이터베이스의 정보, 처리 결과에 대한 정보 등을 가져오기

※ java.sql.* 패키지(Package)

- Java Application 으로부터 Database를 조작하는 API는 JDK 코어 API로 『java.sql.*』 패키지에 설정되어 있다.
- JDBC는 데이터베이스에 접속하기 위해 한 개의 클래스(java.sql.DriverManager)와 두 개의 인터페이스(java.sql.Driver와 java.sql.Connection)를 사용한다.

1장을 마치며.....

Q & A