

데이터베이스 설계

1. 데이터베이스 설계

- ❖ 현실 세계의 업무적인 프로세서를 물리적으로 데이터베이스화 하기 위한 과정
- ❖ 설계과정
 - ✓ 요구 조건 분석: 사용자가 원하는 데이터베이스의 용도를 파악하는 단계
 - ✓ 개념적 설계: 사용자의 요구사항을 이해하기 쉬운 형식으로 간단히 기술하는 단계로 보통 E-R Diagram 작성
 - ✓ 논리적 설계: 개념적 설계에서 만들어진 구조를 실제 DBMS에 맞도록 스키마를 설계하는 단계로 보통 이 단계에서 정규화 과정을 거침
 - ✓ 물리적 설계: 데이터베이스의 물리적 구조와 접근 방법을 설계
 - ✓ 구현: 실제 데이터베이스 생성



2. 요구 조건 분석

- ❖ 고객이 무엇을 원하는지 정확하게 분석하기 위해서 요구 사항들을 모으고 시스템 개발의 목표와 방향성을 기준으로 해당 업무에 대해 철저히 분석
- ❖ 또한 현재 시스템의 운영상태 등을 분석하고 사용자들의 요구사항에 대한 분석까지 포함하는 단계로 시스템 개발의 성패를 좌우하는 가장 중요한 단계
- ❖ 전체 업무 내용과 업무 흐름에 대한 내용을 정리한 후 회사에서 사용하고 있는 문서를 파악하고 설문 조사나 인터뷰를 통해서 요구 사항을 파악
- ❖ 이렇게 파악된 내용을 가지고 요구 사항 정의서를 생성

Ex)

- 1.회사에는 다수의 사원이 근무하고 있으면 사원들은 각자 부서에 소속되어 근무한다.
- 2.각 사원에 대해서 사원번호, 사원명, 직책, 급여를 명시해야 하며 입사일은 년, 월, 일을 세분하여 나타낸다.
- 3.한 명의 사원은 한 부서에만 소속되며 각 부서에 대해서 부서번호, 부서명, 부서가 위치한 지역을 나타낸다.



3. 개념적 설계

- ❖ 요구 분석 단계에서 산출된 요구 사항 정의서를 가지고 DBMS에 독립적인 표현법인 ERD를 작성
- ❖ 이 단계에서는 Entity, Attribute, Relation을 파악

- ✓ Entity(개체)는 데이터베이스에 데이터로 표현하려는 것으로 사람이 생각하고 있는 하나의 개념이나 정보 단위로 독립적으로 존재할 수 있고 그 자체로 다른 Entity와 구분되는 것으로 Record 라고도 함

Ex)사원 엔티티, 부서 엔티티

- ✓ Attribute(속성)는 데이터의 가장 작은 논리적 단위로 하나의 Entity는 하나 이상의 Attribute로 구성되며 각각의 Attribute는 Entity의 특성, 상태 등을 나타내는 것으로 item 또는 field라고도 함

Ex)사원 엔티티의 속성: 사원번호, 이름, 주소, 전화번호, 소속부서

부서 엔티티의 속성: 부서번호, 부서명, 위치


- ✓ Relation은 Entity 와 Entity 또는 Entity 와 Attribute 간의 연관성을 가리키며 일반적으로 동사를 이용해서 표현

Ex)사원은 부서에 소속되어 있다.



3. 개념적 설계

❖ ERD 표기법

기 호	의 미
	개체
	속성
	기본키
	관계
	개체 타입과 속성을 연결
	개체간의 관계 타입

3. 개념적 설계

❖ 속성

- ✓ 단순 속성(Simple): 더 이상 작은 요소로 분해할 수 없는 속성(이름, 나이 등)
- ✓ 복합 속성(Composite): 여러 개의 단순 속성으로 분해할 수 있는 속성(주소 - 기본주소, 상세주소로 분해 가능, 입사일 - 년, 월, 일로 분해 가능)
- ✓ 단일 값 속성(Single-Value): 하나의 엔티티에 대해서 하나의 값만 갖는 것
- ✓ 다중 값 속성(Multi-Value): 하나의 엔티티에 대해서 여러 개의 값을 갖는 것으로 취미는 한 명의 사원이 여러 개를 갖는 것이 가능
- ✓ 유도 속성: 다른 속성으로부터 유도 되어지는 값(나이 - 생년월일을 알면 나이는 계산이 가능)
- ✓ 저장 속성: 유도 속성의 값을 계산할 수 있는 속성
- ✓ NULL: 아직 알려지지 않은 속성



3. 개념적 설계

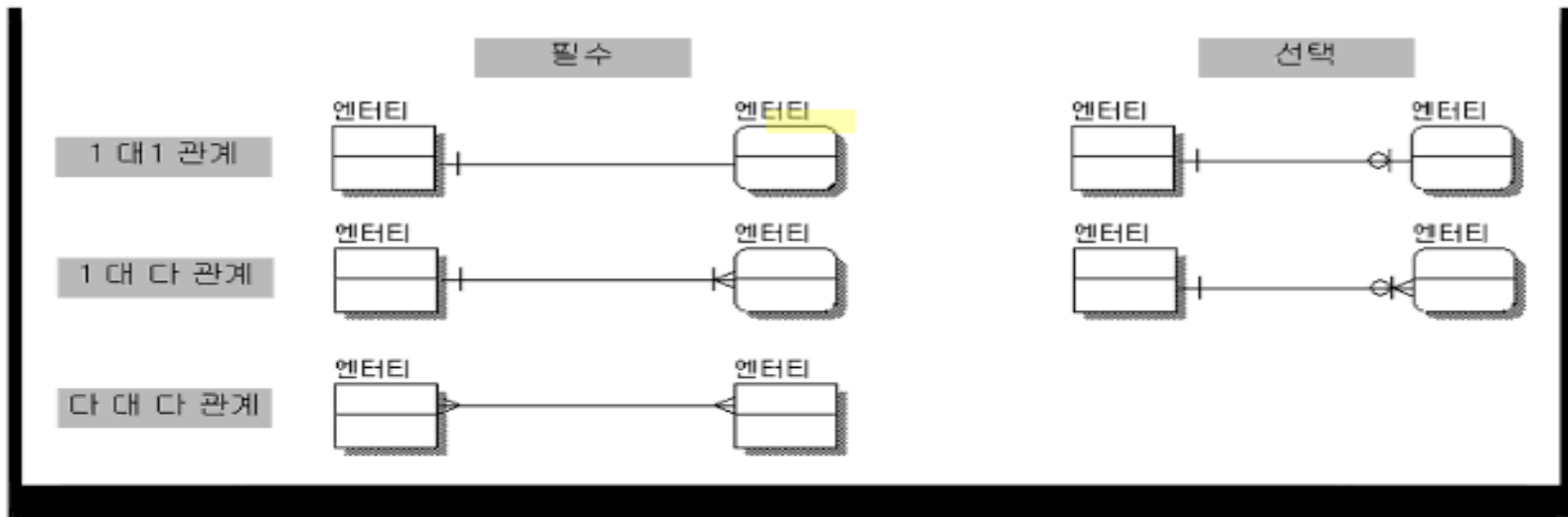
- ❖ 요구 사항 정의서에서의 Entity 도출
 - ✓ 요구 사항 정의서에서 명사를 찾아 표시
 - ✓ 중복된 명사는 하나만 도출
 - ✓ 개념이 명확하지 않거나 업무의 진행 과정을 나타내는 단어는 제거
- ❖ Primary Key 도출
 - ✓ Entity 내에서 다른 Entity를 구별하기 위한 속성을 생성
- ❖ Relation
 - ✓ Entity 사이의 관계를 표현
 - ✓ Entity 사이의 대응 수(Cardinality)를 파악해서 1:1, 1:N, N:M 관계인지를 파악해서 ERD에 표시



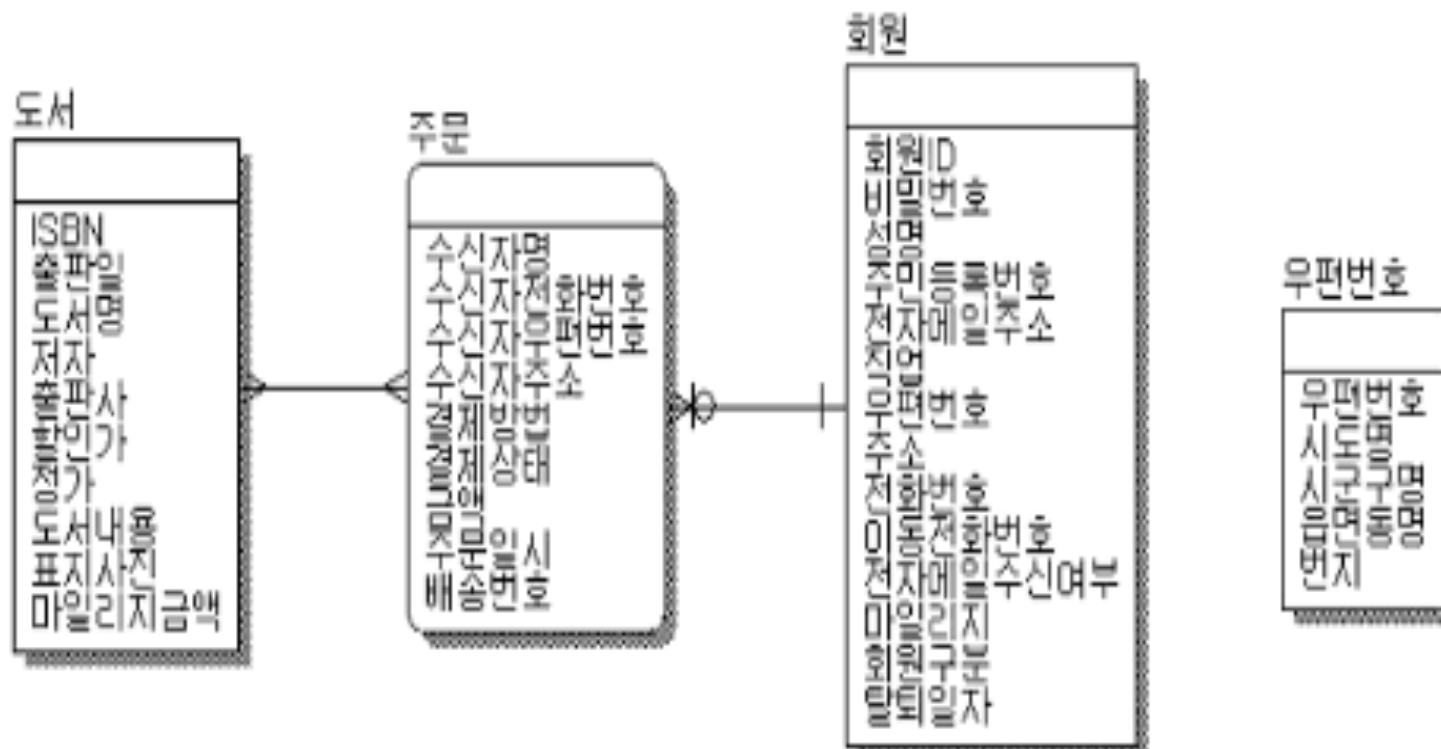
4. 논리적 설계

- ❖ 개념적 설계에서 만든 ERD를 바탕으로 DBMS를 특정하여 논리적 설계를 수행
- ❖ 표기법은 2가지인데 하나는 IE(Information Engineering, 정보공학 표기법) 이고 다른 하나는 Ideflx(Information DEfinition for Information Modeling)
- ❖ 보통은 IE 방식을 사용하고 미 국방성 프로젝트의 경우는 Ideflx가 표준 표기법

다음 그림은 위에서 설명한 엔터티 사이의 관계에 대한 E-R(Entry-Relationship) 모델의 정보 공학(IE, Information Engineering) 표기법을 보여준다.



4. 논리적 설계



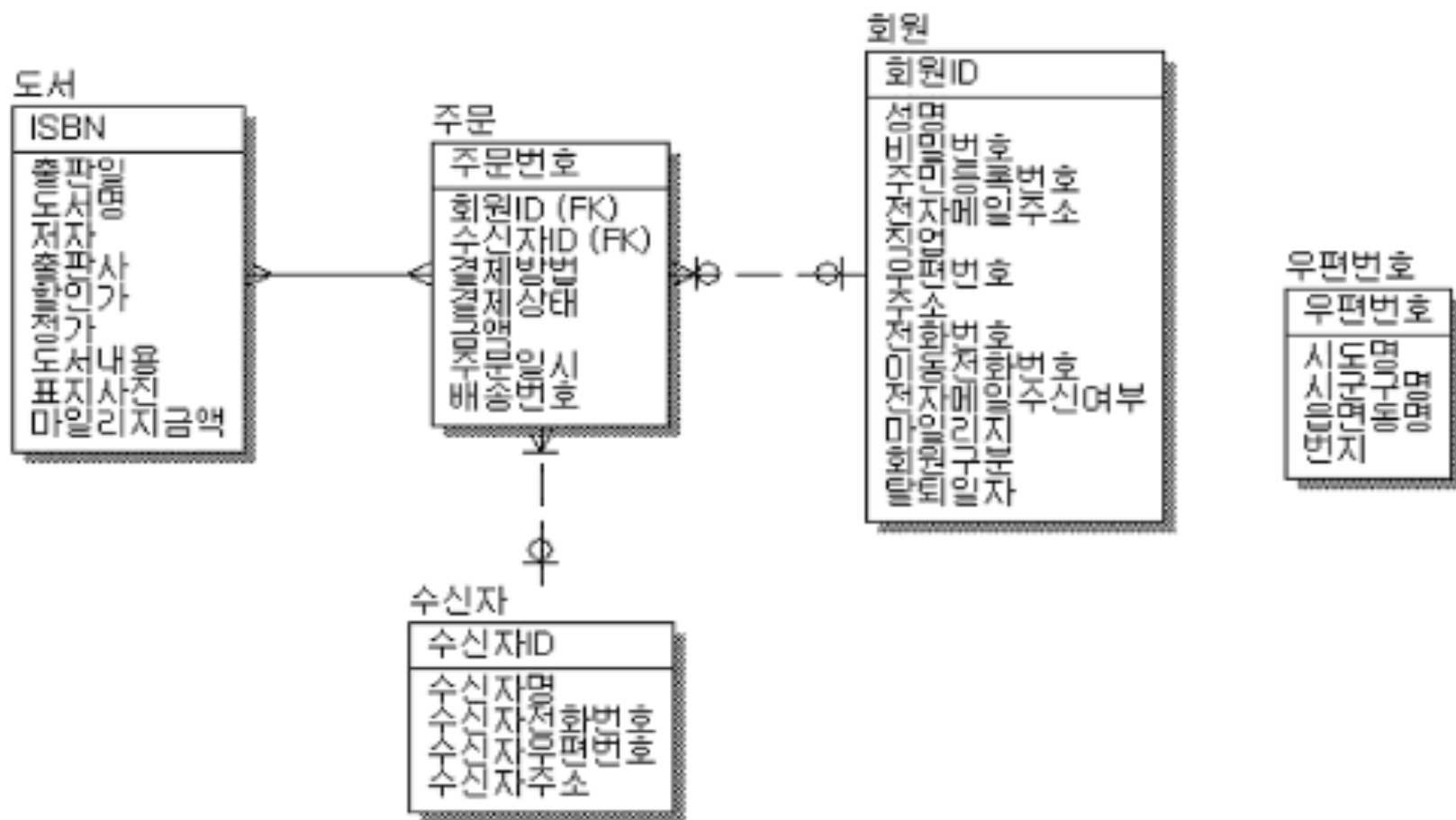
4. 논리적 설계

❖ Foreign Key(외래키, 참조키)

- ✓ 관련성이 있는 Entity를 부모 - 자식 관계로 구분하는데 먼저 정보를 생성해서 가지고 있는 쪽을 부모로 보고 나중에 부모의 정보를 이용하는 쪽을 자식으로 설정해서 부모의 기본키를 자식의 외래키로 추가
- ✓ 정보의 생성 순서가 명확하지 않을 때는 1:1 관계인 경우는 각각의 기본키를 상대방 테이블의 외래키로 추가하고 1:N 관계인 경우는 1쪽의 기본키를 N쪽에 외래키로 추가하며 N:M 관계일 때는 각각의 기본키를 가지는 별도의 테이블을 생성

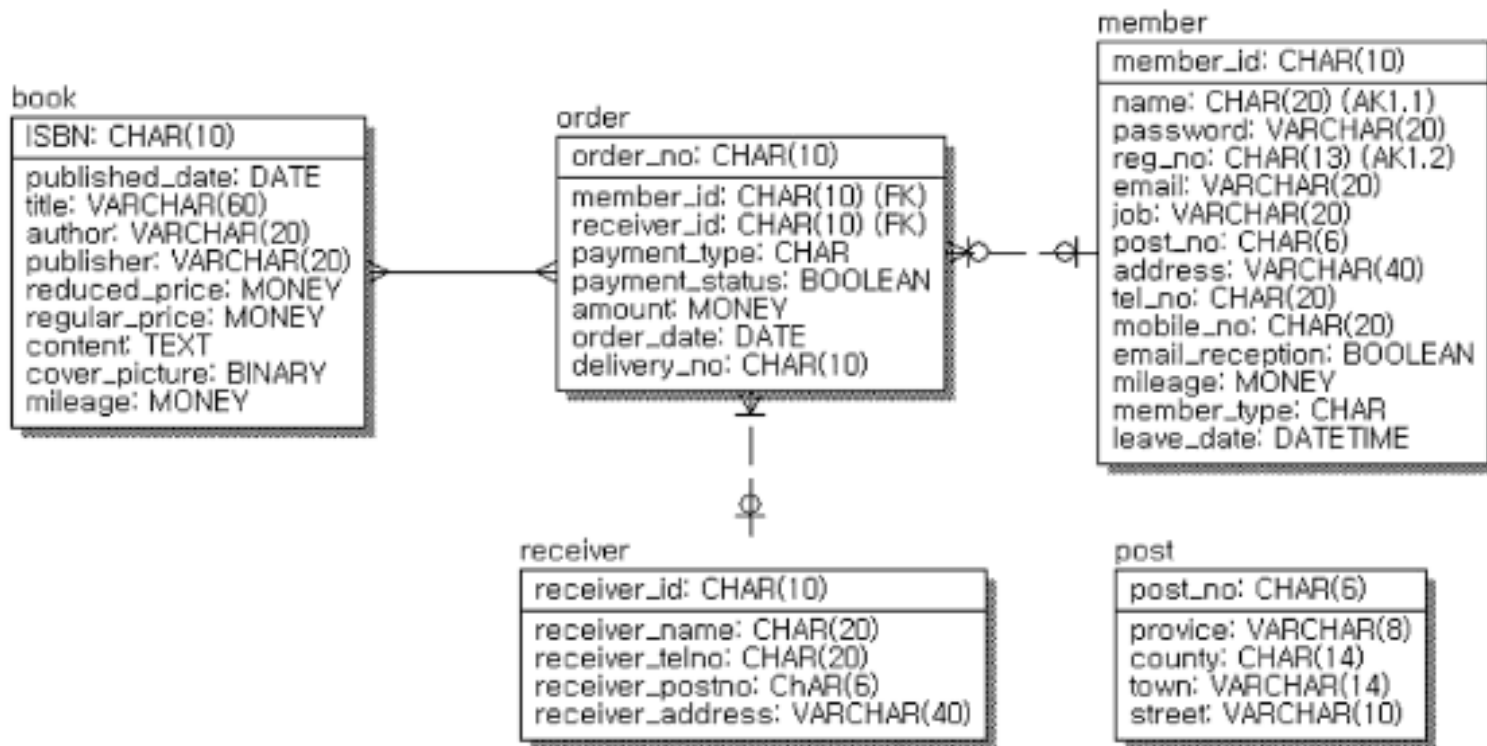


4. 논리적 설계



5. 물리적 설계

- ❖ 개발에 사용할 데이터베이스를 선정하여 특정 데이터베이스로 구현될 수 있도록 구체적으로 설계하는 과정으로 산출물은 테이블 기술서
- ❖ DataDictionary
 - ✓ 엔티티 이름과 속성 이름들의 의미를 정의해 놓은 문서



5. 물리적 설계

- ❖ 테이블 기술서
 - ✓ 개별 테이블에 대한 상세한 문서화 수단

테이블 기술서

테이블명(영문) (한글)	class_apply	관련업무		취트니스 센터 관리 시스템		
	개인수강등록정보	DB사용자 계정		관리자		
영문칼럼(한글)	제약조건	Null여부	F.K 테이블	F.K 컬럼	Data타입	길이
mem_num (회원코드)	PK	NN	FK		varchar	40
class_app_num(수강강좌일련코드)	PK	NN			int	20
mem_class_name (강좌명)					varchar	40
class_apply_date (수강신청일)					varchar	45
class_real_st_date (수강시작일)					varchar	45
class_real_fin_date (수강종료일)					varchar	45
비고						

테이블 관련 DDL

1. 테이블 구조 정의

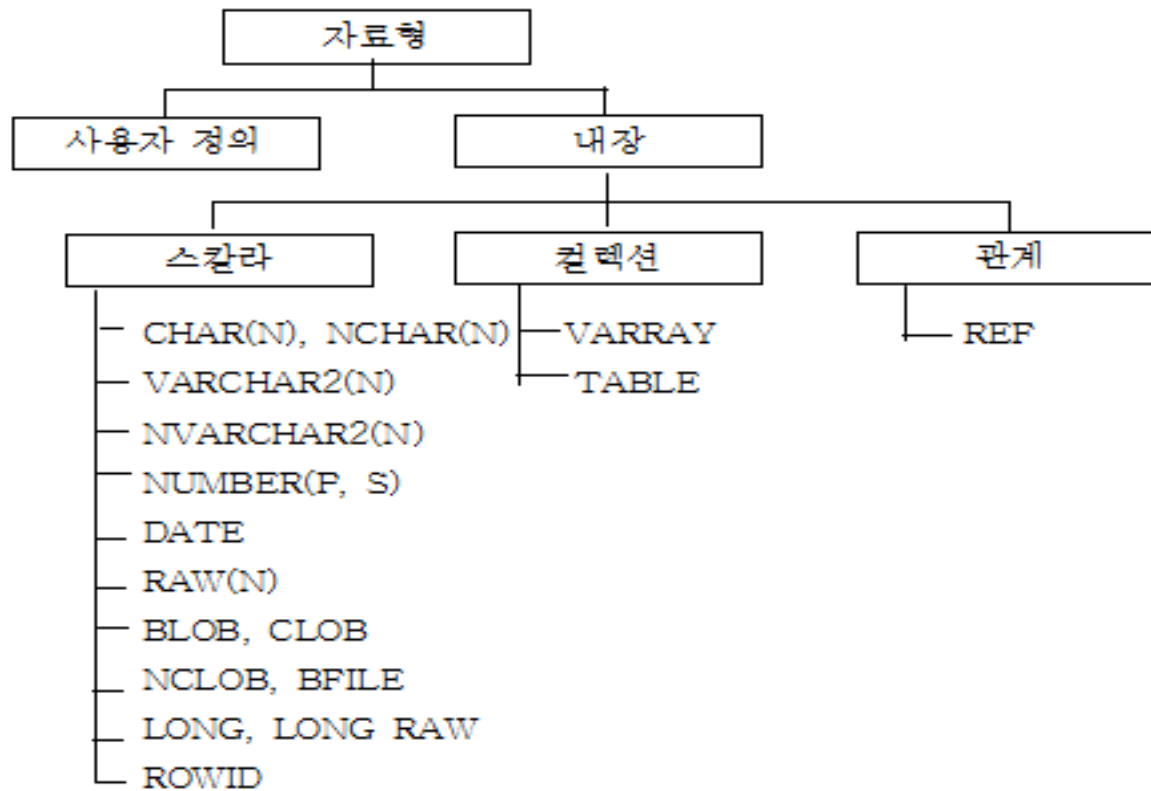
❖ CREATE TABLE 문의 기본 형식

```
CREATE TABLE table_name  
(column_name, data_type expr, ...);
```



1.1 데이터 형

- ❖ 데이터베이스에는 문자, 숫자, 날짜, 이미지 등과 같은 다양한 형태의 데이터가 저장
- ❖ 오라클에서 제공되는 데이터 형의 종류



1.1 데이터 형

이름	비고
CHAR(size)	고정 길이 문자 데이터. VARCHAR2와 동일한 형태의 자료를 저장할 수 있고, 입력된 자료의 길이와는 상관없이 정해진 길이만큼 저장 영역 차지하고 최소 크기는 1
VARCHAR2(size)	2000 Bytes 가변 길이 문자 데이터. 실제 입력된 문자열의 길이만큼 저장 영역을 차지. 최대 크기는 명시해야 하며 최소 크기는 1
NUMBER	Internal Number Format 최고 40자리까지의 숫자를 저장할 수 있습니다. 이때 소수점이나 부호는 길이에 포함되지 않음
NUMBER(w)	W자리까지의 수치로 최대 38자리까지 가능 (38자리가 유효 숫자이다.)
NUMBER(w, d)	W는 전체 길이, d는 소수점 이하 자릿수이며 소수점은 자릿수에 포함되지 않음
DATE	BC 4712년 1월 1일~AD 4712년 12월 31일까지의 날짜

1.1 데이터 형

이름	비고
LONG	가변 길이의 문자형 데이터 타입, 최대 크기는 2GB
LOB	2GB까지의 가변 길이 바이너리 데이터를 저장 이미지 문서, 실행 파일을 저장할 수 있음
ROWID	ROWID는 Tree-piece Format을 가짐. ROWID는 DB에 저장되어 있지 않으며, DB Data도 아님
BFILE	대용량의 바이너리 데이터를 파일 형태로 저장 최대 4GB
TIMESTAMP(n)	DATE 형의 확장된 형태
INTERVAL YEAR TO MONTH	년과 월을 이용하여 기간을 저장
INTERVAL DAY TO SECOND	일, 시, 분, 초를 이용하여 기간을 저장 두 날짜 값의 정확한 차이를 표현하는데 유용

1.2 LOB

- ❖ LOB(Large OBject) 데이터 형은 텍스트, 그래픽 이미지, 동영상, 사운드와 같이 구조화되지 않은 대용량의 텍스트나 멀티미디어 데이터를 저장하기 위한 데이터 형
- ❖ 최대 4GB 까지 저장 가능합니다. 오라클에서 제공되는 LOB 데이터 형은 BLOB, CLOB, NCLOB, BFILE 등이 있음
- ❖ BLOB는 그래픽 이미지, 동영상, 사운드와 같은 구조화되지 않은 데이터를 저장하기 위해 사용
- ❖ CLOB는 e-BOOK과 같은 대용량의 텍스트 데이터를 저장하기 위해서 사용
- ❖ NCLOB은 국가별 문자 셋 데이터를 저장하고, BFILE은 바이너리 데이터를 파일 형태로 저장



실습

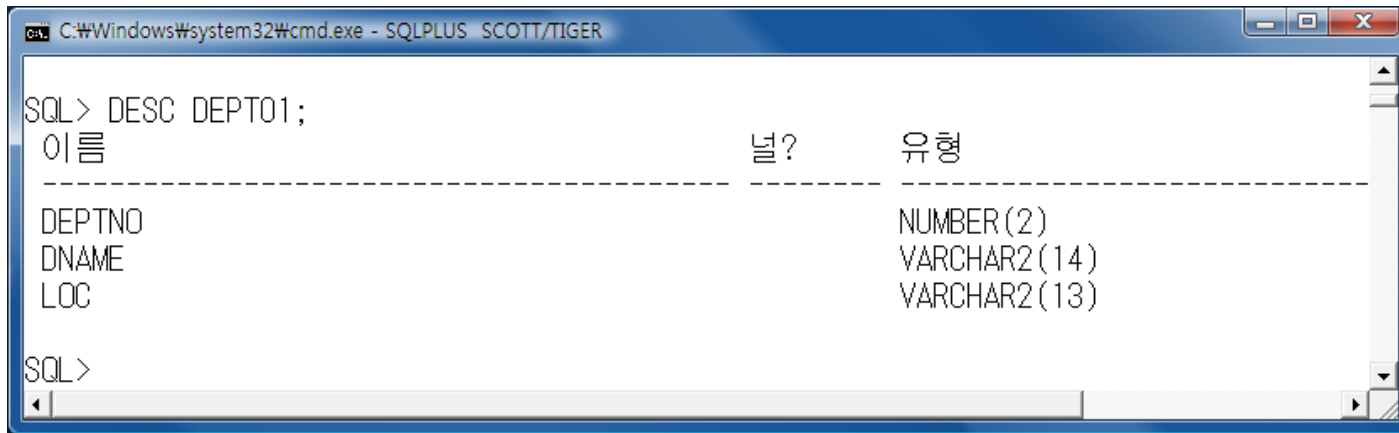
- ❖ 사원번호, 사원이름, 급여 3개의 칼럼으로 구성된 EMP01 테이블을 생성
- ❖ CREATE TABLE 명령어로 EMP01 테이블을 새롭게 생성

```
CREATE TABLE EMP01(  
    EMPNO NUMBER(4),  
    ENAME VARCHAR2(20),  
    SAL NUMBER(7, 2));
```



연습문제

1. 다음과 같은 구조의 테이블을 CREATE TABLE 명령어로 생성하되 테이블의 이름은 DEPT01



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - SQLPLUS SCOTT/TIGER". The user has entered the command "SQL> DESC DEPT01;". The output displays the table structure for DEPT01, with columns listed on the left and their data types on the right, separated by a dashed line. The columns are DEPTNO, DNAME, and LOC. The data types are NUMBER(2), VARCHAR2(14), and VARCHAR2(13) respectively. The prompt "SQL>" is visible at the bottom left of the window.

이름	널?	유형
DEPTNO		NUMBER(2)
DNAME		VARCHAR2(14)
LOC		VARCHAR2(13)

서브 쿼리로 테이블 생성하기

- ❖ CREATE TABLE 문에서 서브 쿼리를 사용하여 이미 존재하는 테이블과 동일한 구조와 내용을 갖는 새로운 테이블을 생성할 수 있음
- ❖ CREATE TABLE 명령어 다음에 컬럼을 일일이 정의하는 대신 AS 절을 추가하여 EMP 테이블과 동일한 내용과 구조를 갖는 EMP02 테이블을 생성

```
CREATE TABLE EMP02  
AS  
SELECT * FROM EMP;
```



복제 테이블 생성

- ❖ 기존 테이블에서 원하는 컬럼만 선택적으로 복사해서 생성할 수도 있음
- ❖ 서브 쿼리문의 SELECT 절에 * 대신 원하는 컬럼명을 명시하면 기존 테이블에서 일부의 컬럼만 복사할 수 있음

```
CREATE TABLE EMP03  
AS  
SELECT EMPNO, ENAME FROM EMP;
```



연습문제

2. EMP 테이블을 복사하되 사원번호(EMPNO), 사원이름(ENAME), 급여(SAL) 컬럼으로 구성된 테이블을 생성(테이블의 이름은 EMP04)



```
SQL> SELECT * FROM EMP04;
```

EMPNO	ENAME	SAL
7369	SMITH	800
7499	ALLEN	1600
7521	WARD	1250
7566	JONES	2975
7654	MARTIN	1250
7698	BLAKE	2850
7782	CLARK	2450
7788	SCOTT	3000
7839	KING	5000
7844	TURNER	1500
7876	ADAMS	1100
7900	JAMES	950
7902	FORD	3000
7934	MILLER	1300

14 개의 행이 선택되었습니다.

복제 테이블 생성

- ❖ 기존 테이블에서 원하는 행만 선택적으로 복사해서 생성할 수도 있음
- ❖ 서브 쿼리문의 SELECT 문을 구성할 때 WHERE 절을 추가하여 원하는 조건을 제시하면 기존 테이블에서 일부의 행만 복사

```
CREATE TABLE EMP05  
AS  
SELECT * FROM EMP  
WHERE DEPTNO=10;
```



1.2 테이블의 구조만 복사하기

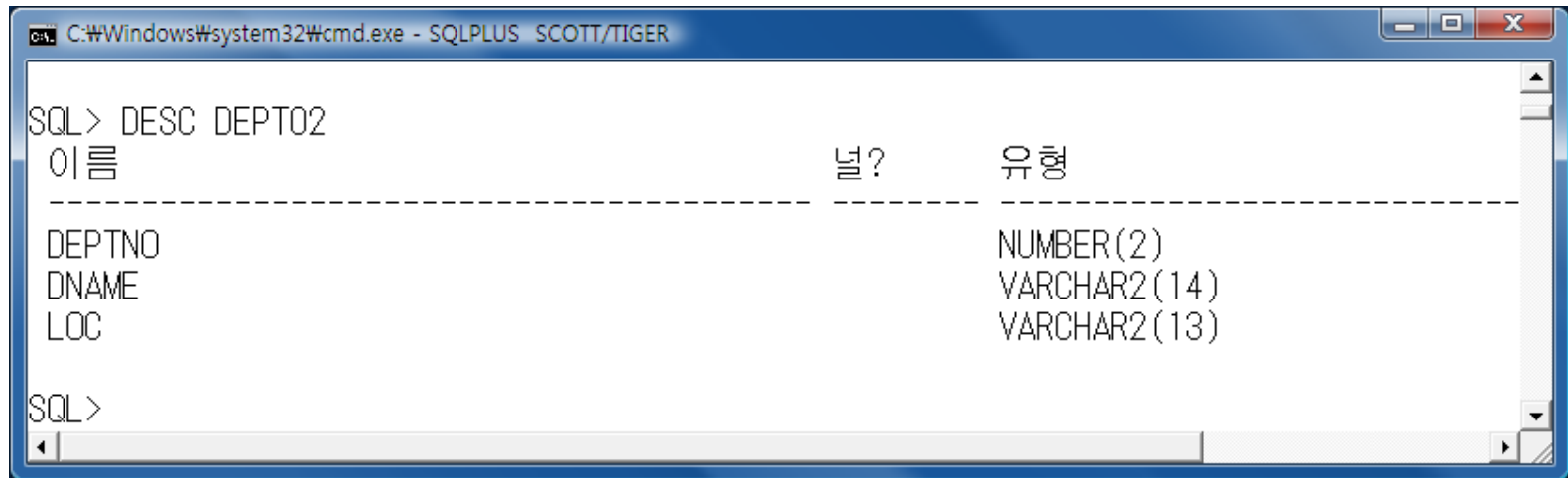
- ❖ 테이블의 구조만 복사하는 것은 별도의 명령이 있는 것이 아니고 서브 쿼리를 이용해야 하는데 WHERE 조건 절에 항상 거짓이 되는 조건을 지정하게 되면 테이블에서 얻어질 수 있는 row가 없게 되므로 빈 테이블이 생성됨
- ❖ WHERE 1=0; 조건은 항상 거짓
- ❖ 이를 이용하여 테이블의 데이터는 가져오지 않고 구조만 복사

```
CREATE TABLE EMP06  
AS  
SELECT * FROM EMP WHERE 1=0;
```



연습문제

3. DEPT 테이블과 동일한 구조의 빈 테이블을 생성(테이블의 이름은 DEPT02)



```
C:\Windows\system32\cmd.exe - SQLPLUS SCOTT/TIGER

SQL> DESC DEPT02
이름                  날?      유형
-----
DEPTNO                NUMBER(2)
DNAME                 VARCHAR2(14)
LOC                   VARCHAR2(13)

SQL>
```

2. 테이블 구조 변경

- ❖ ALTER TABLE 명령문은 기존 테이블의 구조를 변경하기 위한 DDL 명령문
- ❖ 테이블에 대한 구조 변경은 컬럼의 추가, 삭제, 컬럼의 타입이나 길이를 변경
- ❖ 테이블의 구조를 변경하게 되면 기존에 저장되어 있던 데이터에 영향을 줄 수 있음
- ❖ ALTER TABLE로 칼럼 추가, 수정, 삭제하기 위해서는 다음과 같은 명령어를 사용
 - ✓ ADD COLUMN 절을 사용하여 새로운 칼럼을 추가
 - ✓ MODIFY COLUMN 절을 사용하여 기존 칼럼을 수정
 - ✓ DROP COLUMN 절을 사용하여 기존 칼럼을 삭제



2.1 새로운 컬럼 추가

- ❖ ALTER TABLE ADD 문은 기존 테이블에 새로운 컬럼을 추가
- ❖ 새로운 컬럼은 테이블 맨 마지막에 추가되므로 자신이 원하는 위치에 만들어 넣을 수 없음
- ❖ 이전에 추가해 놓은 데이터가 존재한다면 그 데이터에도 컬럼이 추가되지만 컬럼의 값은 NULL 값으로 설정

```
ALTER TABLE table_name  
ADD (column_name, data_type expr, ...);
```



실습하기

- ❖ EMP01 테이블에 문자 타입의 직급(JOB) 칼럼을 추가

```
ALTER TABLE EMP01  
ADD(JOB VARCHAR2(9));
```



연습문제

3. DEPT02 테이블에 문자 타입의 부서장(DMGR) 칼럼을 추가

```
C:\Windows\system32\cmd.exe - SQLPLUS SCOTT/TIGER

SQL> DESC DEPT02;
이름                                널?       유형
-----
DEPTNO                             NUMBER(2)
DNAME                              VARCHAR2(14)
LOC                                 VARCHAR2(13)
DMGR                                NUMBER(4)

SQL>
```



2.2 기존 컬럼 속성 변경하기

- ❖ ALTER TABLE MODIFY 문을 다음과 같은 형식으로 사용하면 테이블에 이미 존재하는 컬럼을 변경할 수 있음

```
ALTER TABLE table_name  
MODIFY (column_name, data_type expr, ...);
```

- ❖ 컬럼을 변경한다는 것은 컬럼에 대해서 데이터 타입이나 크기, 기본 값들을 변경한다는 의미



실습하기

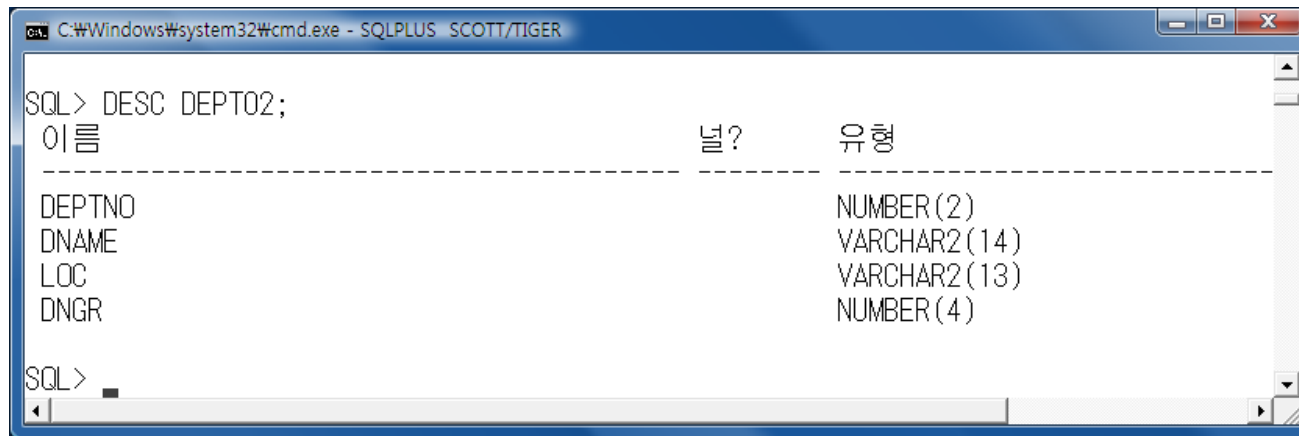
- ❖ emp01 테이블의 직급(JOB) 칼럼을 최대 30글자까지 저장할 수 있게 변경

```
ALTER TABLE EMP01  
MODIFY(JOB VARCHAR2(30));
```



연습문제

- ❖ DEPT02 테이블의 부서장(DMGR) 칼럼을 숫자 타입으로 변경



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - SQLPLUS SCOTT/TIGER". The user has entered the command "SQL> DESC DEPT02;". The output displays the table structure for DEPT02, listing columns and their data types. The columns are DEPTNO, DNAME, LOC, and DMGR. The data types are NUMBER(2), VARCHAR2(14), VARCHAR2(13), and NUMBER(4) respectively. The output is formatted with a header row and a dashed line separator.

이름	널?	유형
DEPTNO		NUMBER(2)
DNAME		VARCHAR2(14)
LOC		VARCHAR2(13)
DMGR		NUMBER(4)

SQL>

2.3 기존 칼럼 삭제

- ❖ 테이블에 이미 존재하는 컬럼을 삭제
- ❖ ALTER TABLE ~ DROP COLUMN 명령어로 칼럼을 삭제

```
ALTER TABLE table_name  
DROP COLUMN column_name;
```

- ❖ EMP01 테이블의 직급 칼럼을 삭제

```
ALTER TABLE EMP01  
DROP COLUMN JOB;
```



연습문제

- ❖ DEPT02 테이블의 부서장(DMGR) 칼럼을 삭제

```
C:\Windows\system32\cmd.exe - SQLPLUS SCOTT/TIGER

SQL> DESC DEPT02;
이름                                널?       유형
-----
DEPTNO                             NUMBER(2)
DNAME                              VARCHAR2(14)
LOC                                VARCHAR2(13)

SQL>
```



2.4 SET UNUSED 옵션

- ❖ 특정 테이블(EMP02)에서 컬럼(JOB)을 삭제하는 경우 무조건 삭제하는 것은 위험
- ❖ 테이블에 저장된 내용이 많을 경우(몇 만 건에 대한 자료) 해당 테이블에서 컬럼을 삭제하는 데 꽤 오랜 시간이 걸리게 될 것입니다. 컬럼을 삭제하는 동안에 다른 사용자가 해당 컬럼을 사용하려고 접근하게 되면 지금 현재 테이블이 사용되고 있기 때문에 다른 사용자는 해당 테이블을 이용할 수 없게 되는 경우가 있는데 삽입, 삭제, 갱신 작업시에는 락(lock)이 생성되서 이런 상황이 발생함
- ❖ ALTER TABLE에 SET UNUSED 옵션을 지정하면 컬럼을 삭제하는 것은 아니지만 컬럼의 사용을 논리적으로 제한할 수 있음
- ❖ SET UNUSED 옵션은 사용을 논리적으로 제한할 뿐 실제로 컬럼을 삭제하지 않기 때문에 작업 시간이 오래 걸리지 않으며 그렇기 때문에 위와 같은 상황 발생 가능성이 낮음
- ❖ 실제 삭제는 drop unused columns 명령



실습하기

- ❖ EMP02 테이블의 JOB 컬럼의 사용을 논리적으로 제한
ALTER TABLE EMP02
SET UNUSED(JOB);
- ❖ 가장 사용빈도가 적은 시간에 실제적인 삭제 작업을 진행
ALTER TABLE EMP02
DROP UNUSED COLUMNS;



3. 테이블 구조 삭제

- ❖ DROP TABLE문은 기존 테이블을 제거
DROP TABLE table_name;
- ❖ EMP01 테이블을 삭제
DROP TABLE EMP01;



4. 테이블의 모든 행을 제거

- ❖ 테이블의 모든 데이터를 제거하기 위한 명령어로 TRUNCATE가 제공
TRUNCATE TABLE table_name
- ❖ EMP02 테이블의 모든 데이터를 제거
TRUNCATE TABLE EMP02;



5. 테이블 이름을 변경

- ❖ 기존에 사용하던 테이블의 이름을 변경하기 위한 명령어로 RENAME이 제공
RENAME *old_name* TO *new_name*
- ❖ EMP02 테이블의 이름을 TEST 란 이름으로 변경
RENAME EMP02 TO TEST;



<연습문제>

1. EMP 테이블의 SAL, COMM을 제외한 모든 COLUMN과 행을 포함하는 EMP_DEMO 테이블을 생성하는 SQL문을 작성

2. EMP 테이블과 DEPT 테이블을 이용하여 아래의 내용을 포함하는 테이블(EMP_DEPT)을 생성

EMPNO	ENAME	JOB	DNAME	LOC
-------	-------	-----	-------	-----

7839	KING	PRESIDENT	ACCOUNTING	NEW YORK
------	------	-----------	------------	----------

7698	BLAKE	MANAGER	SALES	CHICAGO
------	-------	---------	-------	---------

7782	CLARK	MANAGER	ACCOUNTING	NEW YORK
------	-------	---------	------------	----------

.....

14 rows selected.



<연습문제>

3. EMP 테이블과 SALGRADE 테이블을 이용하여 아래의 내용을 포함하는 테이블(EMP_GRADE)을 생성

EMPNO	ENAME	JOB	SAL	COMM	GRADE
7839	KING	PRESIDENT	5000		5
7698	BLAKE	MANAGER	2850		4
7782	CLARK	MANAGER	2450		4
7566	JONES	MANAGER	2975		4

.....

14 rows selected.

4. 3번에서 생성한 테이블에 SAL의 정밀도를 정수 부분을 12자리 소수 이하 4자리로 변경하는 SQL 문을 작성

5. 2번과 3번에서 생성한 테이블을 모두 삭제하는 SQL문을 작성

