

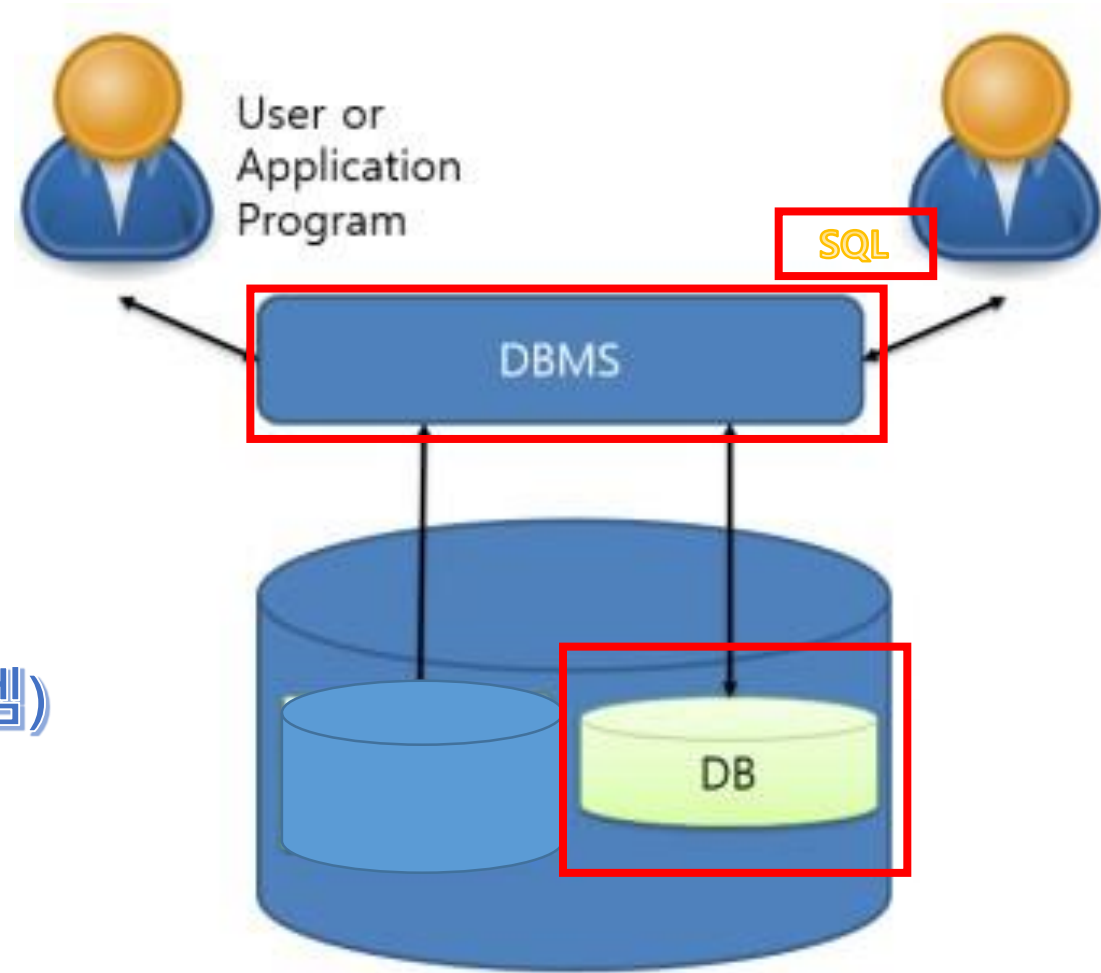
# SQL(시퀀)

구조화 질의 언어

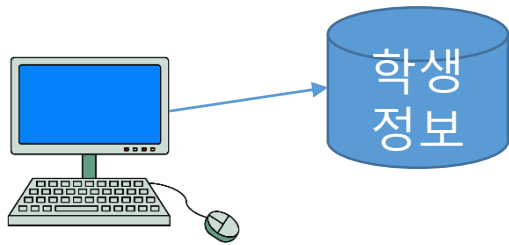
DB(데이터베이스)

DBMS(데이터베이스 관리시스템)

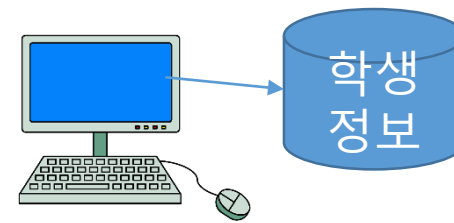
SQL(구조화된 질의 언어)

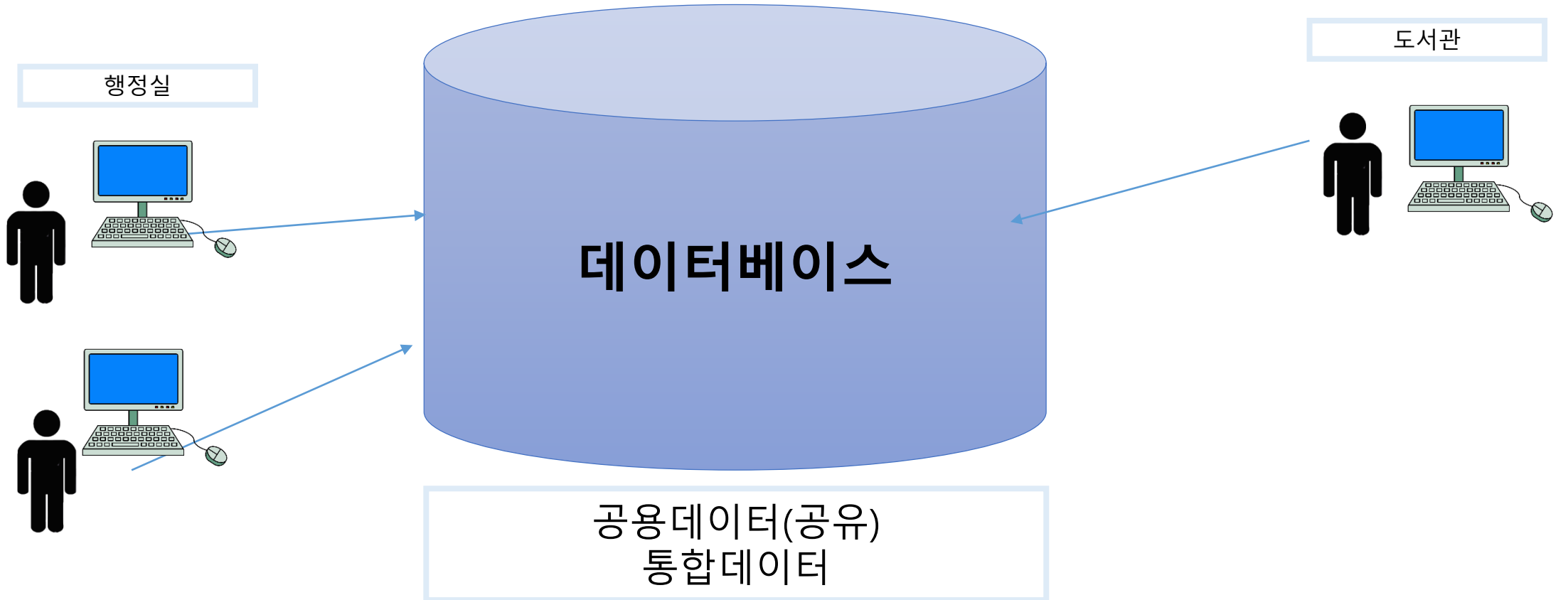


행정실



도서관



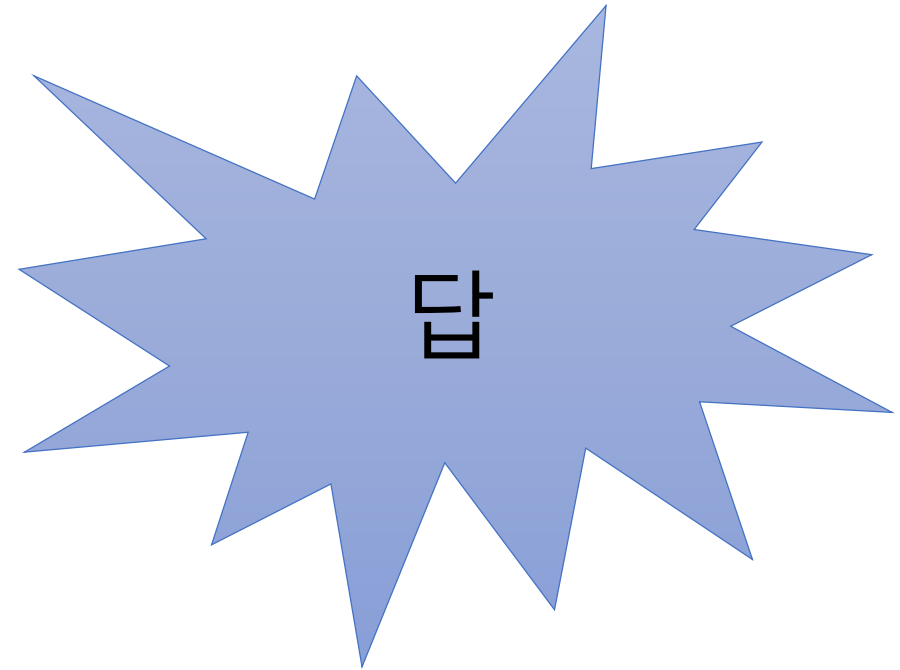


**한 곳에 데이터를 모아놓고 사용하면 좋은 점은?  
반대로 개별적으로 관리 했을 때의 문제점?**

**공간의 효율성**  
(똑같은 데이터를 중복해서 가질 필요 없음)

중복을 제거하는 방법으로 **결합을 줄임**

**동기화문제 해결**



## DB(데이터베이스)

여러 사람에게 공유되어 사용될 목적을 가지고 구조적인 방식으로 관리되는 데이터의 집합이다  
정보의 중복을 최소화하고 한 곳에 저장함으로써 다수의 사용자가 필요한 정보에 효율적으로 접근할 수 있게 한 정보의 집합체

(정리가 잘된 서류캐비닛)

## DBMS(데이터베이스 관리시스템)

데이터베이스의 체계적인 관리는 데이터베이스 관리 시스템을 통해 이루어진다.  
데이터 추가, 변경, 삭제, 검색 기능을 수행한다.

## SQL

데이터베이스 시스템에 질의하는 언어 (사용자 와 DBMS 사이의 약속된 의사소통 언어)

# 데이터베이스 용어

- 테이블

관계형 데이터베이스를 구성하는 기본 데이터 구조로서 **행과 열(레코드, 컬럼)**의 구조를 가지며 이 테이블을 이용하여 데이터를 입력, 수정, 삭제, 추출(조회) 등을 하게된다.

- 스키마

테이블에 데이터가 저장되는 방식을 정의하는 것, 데이터베이스의 논리적 구조

- 열(COLUMN)

- 행(ROW)

- PRIMARY KEY(주키)



# 테이블

고객

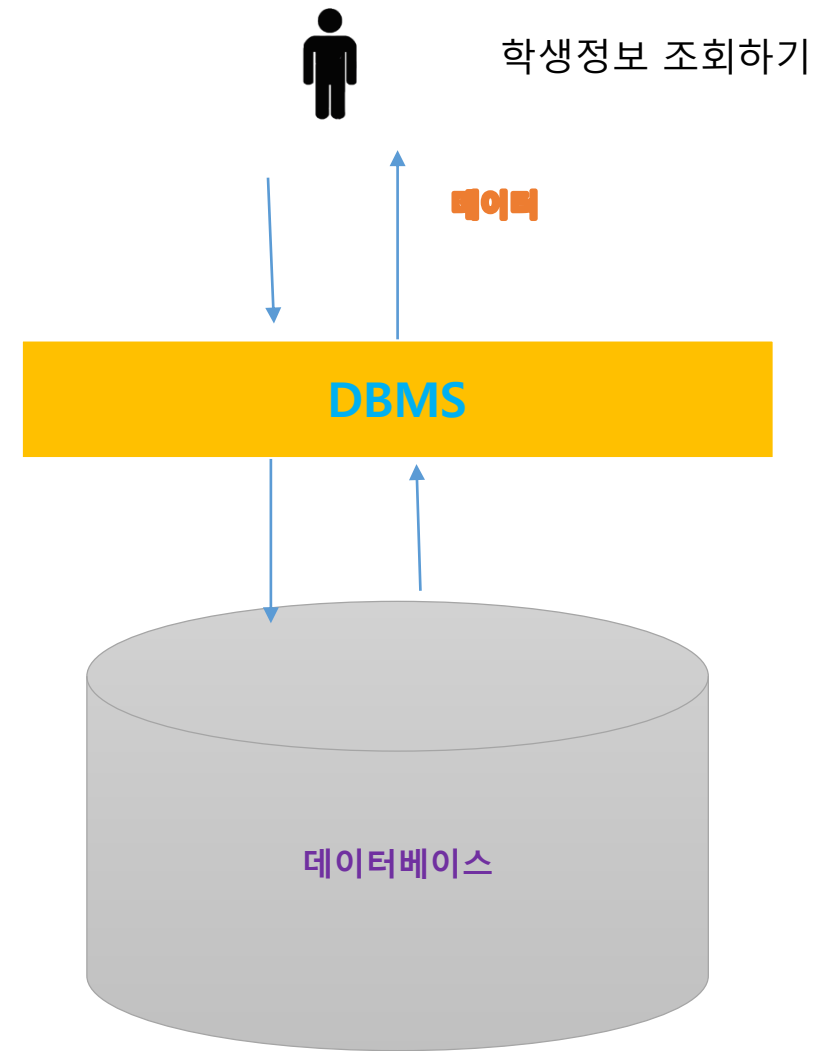
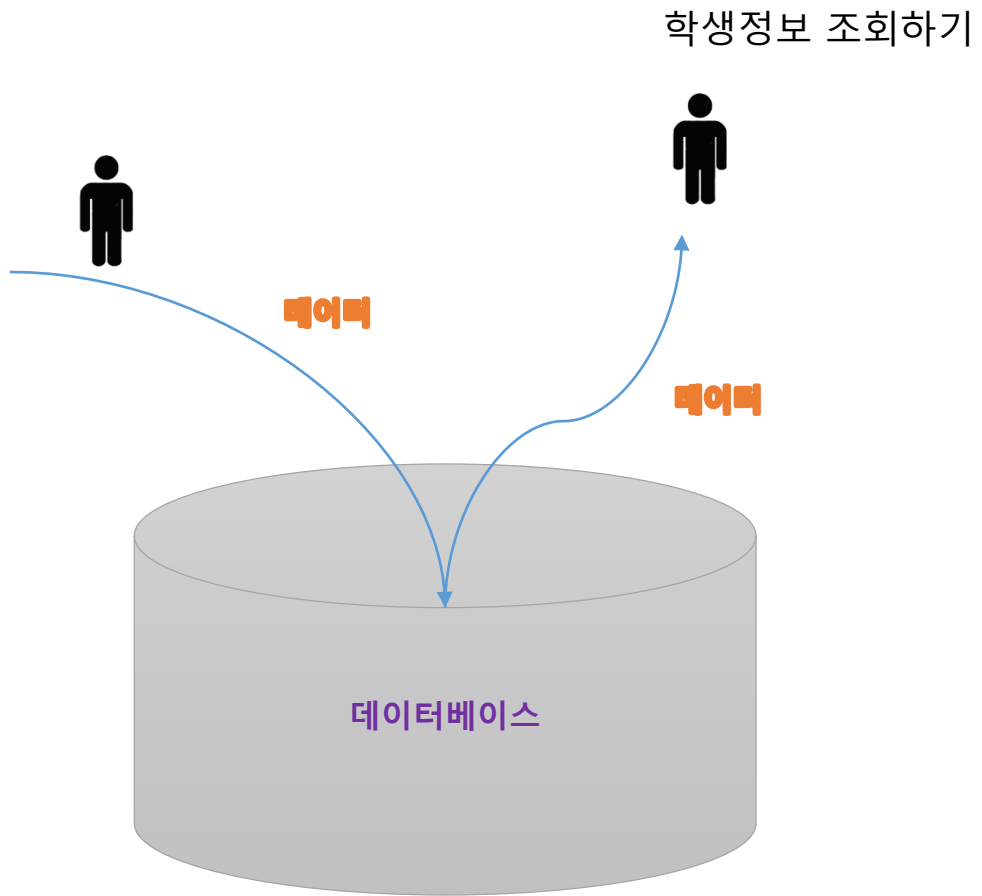
(column)  
컬럼

Schema(스키마)

Row(행)  
레코드

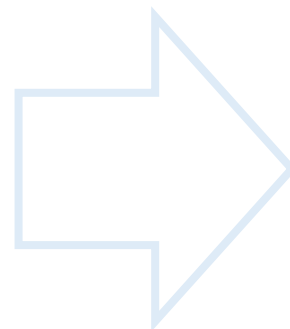
아이디	이름	주소	전화번호
H001	이장우	라스베가스	010-1111-2222
H002	코쿤	L.A	010-2222-3333
H003	기안84	워싱턴D.C	010-3333-4444
H004	박나래	뉴욕	010-4444-5555
H005	전현무	텍사스	010-5555-6666

PRIMARY KEY(주키)



# 데이터베이스 관리시스템의 필요성

여러 사용자가 공용으로 사용을 함  
으로써  
동시성  
성능  
보완  
문제를 해결하기 위함



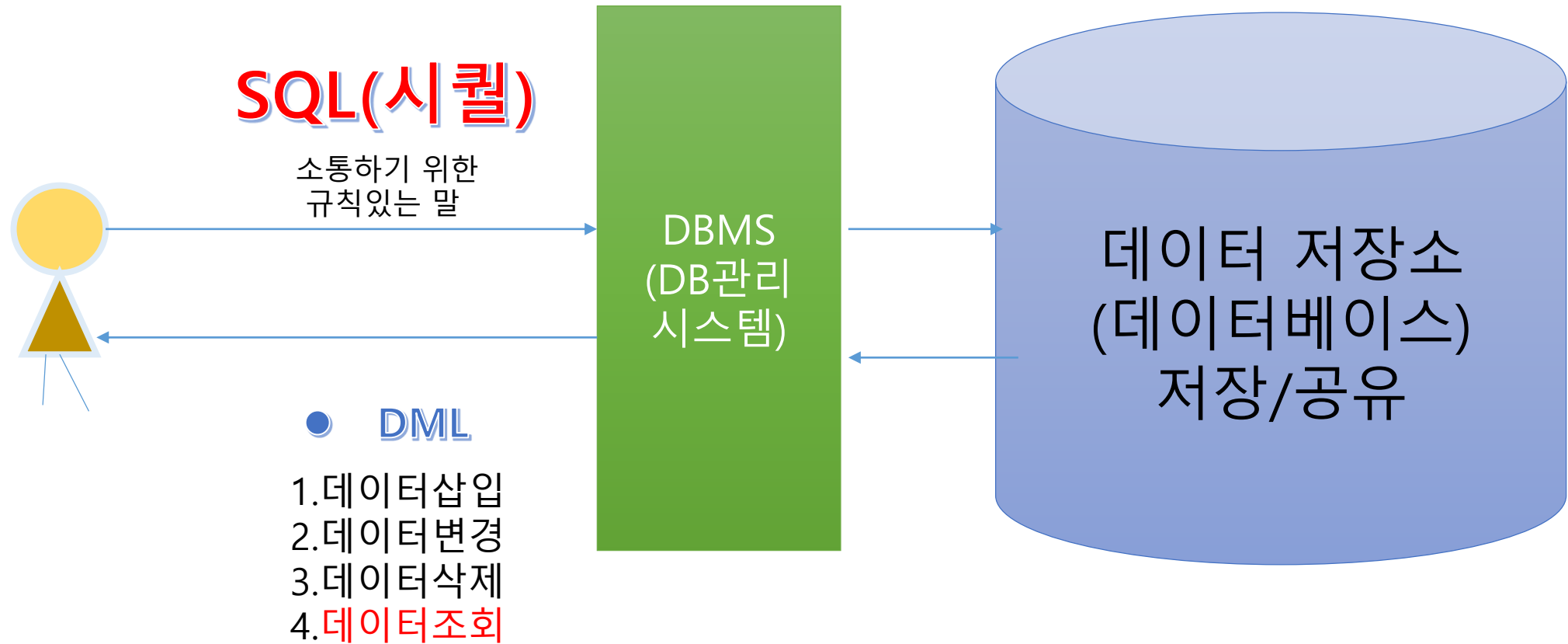
## DBMS

데이터베이스  
관리자역할

# SQL

- **DDL**: 데이터 정의 언어  
CREATE , ALTER, DROP
- **DML**: 데이터 조작언어  
SELECT , INSERT , UPDATE , DELETE
- **DCL**: 데이터 제어언어  
GRANT, REVOKE

## 데이터베이스관리자 역할



# SELECT(데이터 조회)

```
SELECT 컬럼명, 컬럼명 FROM 테이블명 ;
```

```
SELECT ID, NAME, ADDR FROM CUSTOMER ;
```

# SELECT(모든컬럼 조회하기)

```
SELECT * 테이블명 ;
```

```
SELECT * FROM CUSTOMER ;
```

## 데이터 정렬하기( 한 가지 열로 정렬)

오름차순(ASC)

- ORDER BY 열이름 사용하기  
`SELECT ID, NAME FROM CUSTOMER ORDER BY NAME;`
- ORDER BY 열위치 사용하기  
`SELECT ID, NAME FROM CUSTOMER ORDER BY 2;`

내림차순(DESC)

- ORDER BY 열이름 사용하기  
`SELECT ID, NAME FROM CUSTOMER ORDER BY NAME DESC ;`
- ORDER BY 열위치 사용하기  
`SELECT ID, NAME FROM CUSTOMER ORDER BY 2 DESC;`



## 데이터 정렬하기( 여러가지열로 정렬)

- ORDER BY 열이름 사용하기

```
SELECT DEPT, NAME FROM CUSTOMER ORDER BY DEPT, NAME;
```

- ORDER BY 열위치 사용하기

```
SELECT DEPT, NAME FROM CUSTOMER ORDER BY 1, 2;
```


- ORDER BY 열이름 사용하기

```
SELECT ID, NAME FROM CUSTOMER ORDER BY DEPT ASC , NAME DESC ;
```

- ORDER BY 열위치 사용하기

```
SELECT ID, NAME FROM CUSTOMER ORDER BY 1 ASC, 2 DESC;
```

# WHERE 조건절



원하는  
데이터만  
가져오기

<b>SELECT</b>	컬럼명
<b>FROM</b>	테이블명
<b>WHERE</b>	조건;

조건절 연산자

연산자 종류	설명
=	비교 대상에서 같은 조건을 검색(같은것)
!= , <>	비교 대상에서 같지 않은 조건을 검색 (같지않은것)
>	비교 대상에서 큰 조건을 검색 (큰것)
>=	비교 대상에서 크거나 같은 조건을 검색( 이상)
<	비교 대상에서 작은 조건을 검색 ( 작은것)
<=	비교 대상에서 작거나 같은 조건을 검색 (이하)
BETWEEN a AND b	a와 b 사이에 있는 범위 값을 모두 검색(a에서b까지)
IS NULL/ IS NOT NULL	NULL값을 검색/ NULL이 아닌 값을 검색
A AND B	조건이 여러 개 일때 둘 다 만족하는 값 검색
A OR B	A조건이나 B조건 중 한가지라도 만족하는 값을 검색
NOT A	A가 아닌 모든 조건을 검색(조건부정)
IN(a,b,c)	a이거나 b이거나 c인 조건을 검색(괄호안에 있는것만)
NOT IN(a)	괄호안에 있는것만 제외하고 검색
LIKE	특정패턴을 가지고 있는 조건을 검색( '% , _ ) % :글자수에 제한이 없음, _ : 한글자만

# 텍스트 마이닝

# SELECT 절에서 사용되는 키워드

- **DISTINCT 키워드**: 중복을 제거하고 조회함

```
SELECT BAN FROM CLASS_1 ;
```

- **AS 키워드** : 조회시 컬럼명 대신에 별칭을 사용할 수 있음

```
SELECT ID AS USER_ID , NAME AS USER_NAME FROM CUSTOMER ;
```

AS는 생략이 가능함

# 연결연산자 사용하기 ||

컬럼 합치기  
컬럼에 문자열 합치기

```
SELECT  M_ID || M_NAME AS INFO , M_NAME || '님' AS NAME  
FROM    MEMBER_TBL_11 ;
```

함수 ?

기능

함수사용법  
(입력, 반환값)

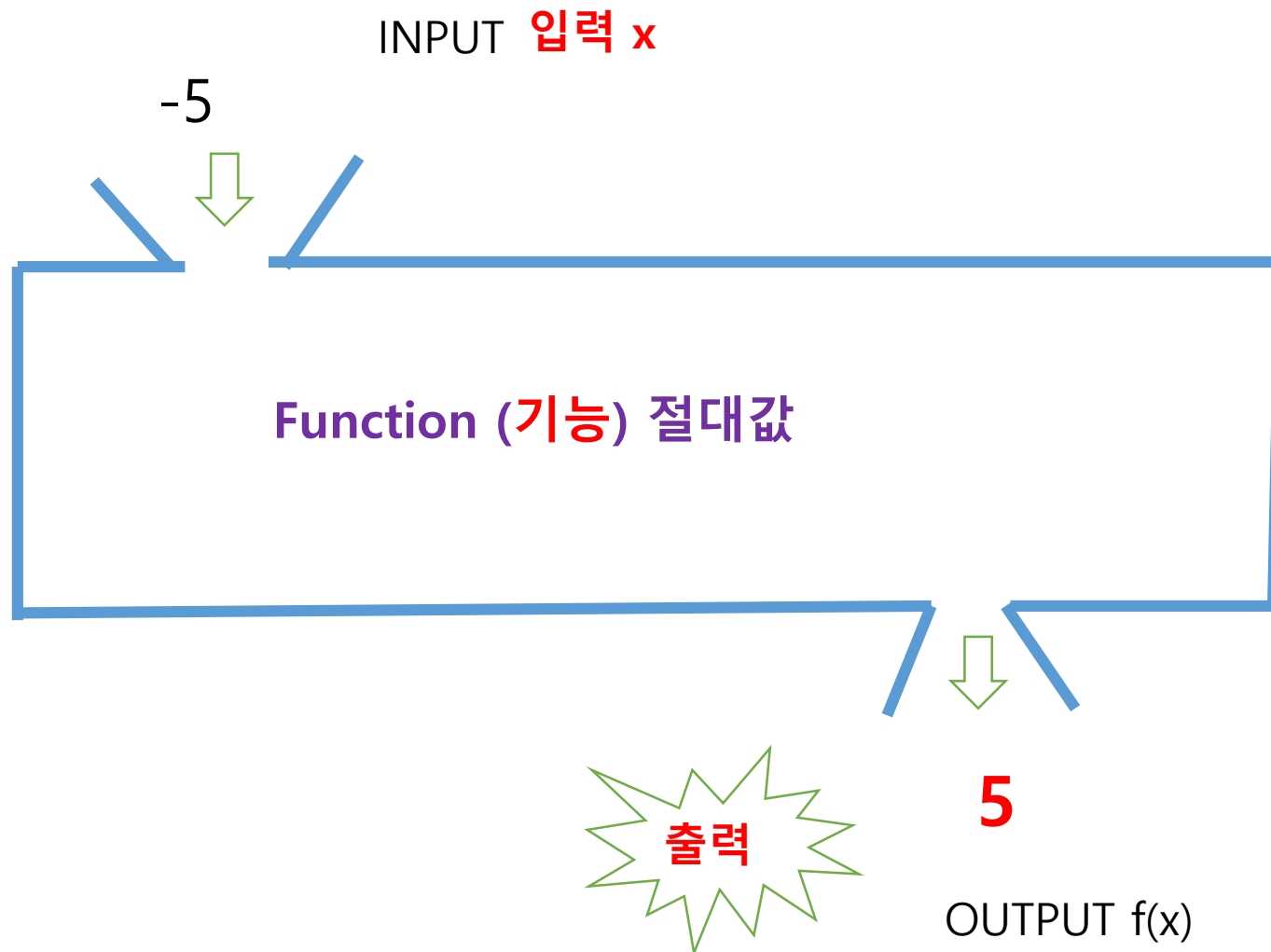
Function

남이 만들어 놓은 함수  
(라이브러리)

# 기본 함수 배우기

자바스크립트 코드  
함수 사용하기

```
let result = Math.abs(-5);  
alert( result);
```





## 문자열함수

함수	설명	비고
LOWER	모든 문자를 소문자로 변환한다	
UPPER	모든 문자를 대문자로 변환한다	
LENGTH	문자의 길이를 나타낸다	
SUBSTR	문자 값 중 원하는 길이만큼만 나타낸다	문자열의 일부만 추출
RTIRM	문자열 오른쪽 공백을 제거한다	
LTRIM	문자열 왼쪽 공백을 제거한다.	
TRIM	문자열 기준 양쪽 공백을 제거한다	
REPLACE	특정문자열을 다른 문자열로 대체한다	
INITCAP	첫글자는 대문자, 나머지는 소문자로 변환한다	

## 숫자함수

함수	설명	비고
ROUND	소수점의 자릿수를 지정하여 반올림한다.	<b>ROUND( 컬럼명, 나타내고 싶은 소수점 자리수)</b> ROUND( 컬럼명 , 1); ROUND( 컬럼명, 0); ROUND( 컬럼명, -1)
TRUNC	해당소수점 자리에서 잘라낼 때 사용한다.	
MOD(M,N)	M을N으로 나눈 나머지를 구한다	
ABS	절대값을 구한다.	

## 날짜함수

함수	설명	비고
ADD_MONTHS	지정한 날짜에 개월 수를 더한 값을 출력	
SYSDATE	현재시스템의 날짜 데이터를 반환한다.	
LAST_DAY	해당 월의 마지막 날짜를 반환한다.	
MONTHS_BETWEEN	두 날짜 사이의 기간을 월로 나타낸다	

## 기타함수

함수	설명	비고
NVL	NULL값 대신 특정값으로 대체하고 싶은 값을 지정할 때 사용한다	
NVL2	NULL값과 NULL값이 아닌경우 대체하고 싶은 값을 지정할 때 사용한다	

## 조건문 이해하기

CASE WHEN 문장

```
SELECT 열이름 , CASE WHEN [조건] THEN [결과값]
                  WHEN [조건] THEN [결과값]
                  ELSE [결과값]
                  END AS 새로운컬럼명
FROM 테이블명 ;
```

# 숫자데이터 요약하기

함수	설명	비고
SUM	행의 합계를 나타낸다.	전체합계: SUM(열이름)
AVG	행의 평균을 나타낸다.	평균: AVG(열이름)
COUNT	행의 수를 나타내다	NULL값을 포함한 전체행의 개수 COUNT(*) NULL값을 제외한 전체행의 개수 COUNT(열이름)
MAX	행의 최대값을 나타낸다	MAX(열이름)
MIN	행의 최소값을 나타낸다	MIN(열이름)

# 문제

포인트금액의 전체합계  
포인트금액의 전체평균  
포인트금액의 최대값 구하기  
포인트금액의 최소값 구하기

```
SELECT SUM(M_POINT)  
FROM MEMBER_TBL_11;
```

```
SELECT AVG(M_POINT)  
FROM MEMBER_TBL_11;
```

```
SELECT MAX(M_POINT)  
FROM MEMBER_TBL_11;
```

```
SELECT MIN(M_POINT)  
FROM MEMBER_TBL_11;
```

SQL> SELECT SUM(M\_POINT)  
2 FROM MEMBER\_TBL\_11;

SUM(M\_POINT)  
-----  
20400

SQL>  
SQL> SELECT AVG(M\_POINT)  
2 FROM MEMBER\_TBL\_11;

AVG(M\_POINT)  
-----  
2266.66667

SQL>  
SQL> SELECT MAX(M\_POINT)  
2 FROM MEMBER\_TBL\_11;

MAX(M\_POINT)  
-----  
4800

SQL>  
SQL> SELECT MIN(M\_POINT)  
2 FROM MEMBER\_TBL\_11;

MIN(M\_POINT)  
-----  
100

SQL> \_

# 데이터의 그룹화, 필터링

**GROUP BY** 절을 사용하여 데이터를 그룹화하는 방법을 알아보자



## MEMBER\_TBL\_11

	⚡ M_ID	⚡ M_PW	⚡ M_NAME	⚡ M_TEL	⚡ M_BIRTHDAY	⚡ M_POINT	⚡ M_GRADE	
1	m106	9999	이알리	010-7777-8888	04/07/10	2900	01	그룹
2	m101	4444	김상우	010-2222-3333	04/01/01	1500	01	
3	m100	1234	성기훈	010-1111-2222	04/01/01	100	01	
4	m107	0101	김미녀	010-8888-9999	04/06/09	1200	01	
5	m102	5555	김일남	010-3333-4444	04/12/10	2000	02	그룹
6	m103	6666	이준호	010-4444-5555	04/04/10	1900	02	
7	m104	7777	김새벽	010-5555-6666	04/09/12	3000	03	그룹
8	m105	8888	최덕수	010-6666-7777	04/08/10	4800	03	
9	m108	0404	이정재	010-9999-8888	04/05/19	3000	03	
10	m109	0448	박해수	010-7878-1111	04/11/27	(null)	(null)	
11	m110	4848	위하준	010-8888-9090	04/11/09	(null)	(null)	

## 전체 포인트금액 합계 구하기

포인트 전체합계금액 : 20400

## 고객등급별 포인트금액 합계 구하기

고객등급	포인트금액합계
01	5700
02	3900
03	10800

```
SELECT      그룹화할 열 이름, 집계함수  
FROM        테이블명  
GROUP BY   열이름 ;
```

```
SELECT      M_GRADE , SUM(M_POINT)  
FROM        MEMBER_TBL_11  
GROUP BY   M_GRADE ;
```

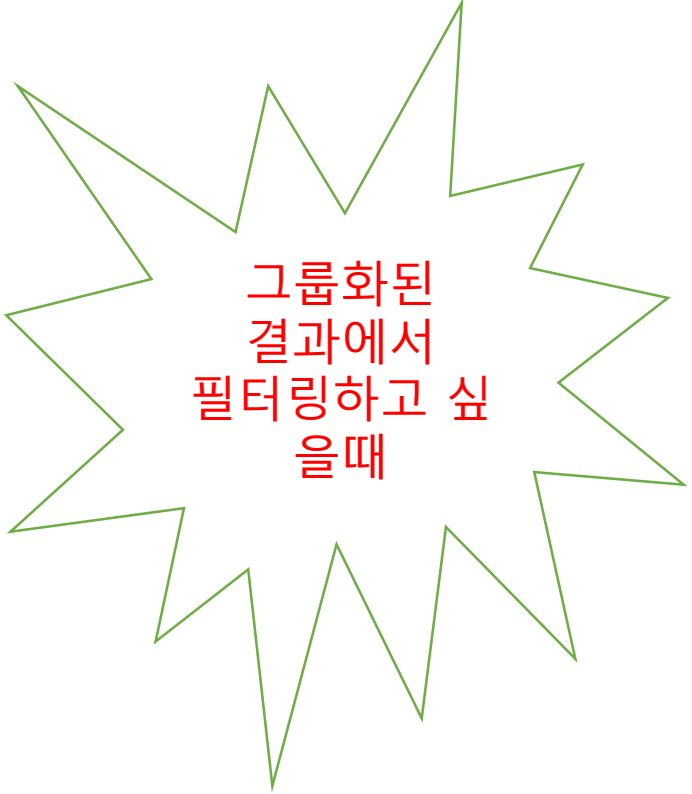
```
SELECT      M_GRADE , SUM(M_POINT)  
FROM        MEMBER_TBL_11  
GROUP BY   M_GRADE  
ORDER BY  M_GRADE;
```

**SELECT**  
**FROM**  
**GROUP BY**  
**HAVING**

그룹화할 열 이름, 집계함수  
테이블명  
열이름  
필터링 조건 ;

**SELECT**  
**FROM**  
**GROUP BY**  
**HAVING**

M\_GRADE , SUM(M\_POINT)  
MEMBER\_TBL\_11  
M\_GRADE  
SUM(M\_POINT) >= 4000 ;



그룹화된  
결과에서  
필터링하고 싶  
을때

	M_GRADE	SUM(M_POINT)
1	01	5700
2	02	3900
3	03	10800
4	(null)	(null)

	M_GRADE	SUM(M_POINT)
1	01	5700
2	03	10800

포인트합계금액  
4000점이상만  
필터링

그룹화된 목록에  
서 필터링 할때  
HAVING 절 사용

**SELECT**  
**FROM**  
**[WHERE]**  
**[ORDER BY ]**

**SELECT 집계함수**  
**FROM**  
**GROUP BY**

**SELECT 집계함수**  
**FROM**  
**GROUP BY**  
**HAVING**

**SELECT**  
**FROM**  
**[WHERE]**  
**[ORDER BY ]**

**SELECT 집계함수**  
**FROM**  
**[WHERE]**  
**GROUP BY**  
**[ORDER BY]**

**SELECT 집계함수**  
**FROM**  
**[WHERE]**  
**GROUP BY**  
**HAVING**  
**[ORDER BY]**

# 조회 쿼리 sql

## 일반

- |   |             |   |
|---|-------------|---|
| ① | SELECT      | 3 |
| ② | FROM        | 1 |
| ③ | [WHERE]     | 2 |
| ④ | [ORDER BY ] | 4 |

## 그룹핑

- |   |            |           |
|---|------------|-----------|
| ① | SELECT     | 컬럼명, 집계함수 |
| ② | FROM       |           |
| ③ | [WHERE]    |           |
| ④ | GROUP BY   | 컬럼명       |
| ⑤ | [HAVING ]  |           |
| ⑥ | [ORDER BY] |           |

- |   |           |           |
|---|-----------|-----------|
| ① | SELECT    | 컬럼명, 집계함수 |
| ② | FROM      |           |
| ③ | GROUP BY  | 컬럼명       |
| ④ | [HAVING ] |           |



# SQL

**DDL**

CREATE  
ALTER  
DROP

테이블을 만들고 변경하는데 사용하는 언어

 **DML**

SELECT  
INSERT  
UPDATE  
DELETE

테이블의 데이터를 추가하고 변경하는데 사용하는 언어

**DCL**

GRANT  
REVOKE

사용자에게 권한 생성  
사용자에게 권한 회수

# 오라클왕국의 왕



SYSTEM  
계정



**CREATE USER** 사용자 **IDENTIFIED BY** 비밀번호;  
**GRANT** CONNECT , RESOURCE , DBA **TO** 사용자;  
**REVOKE** CONNECT, RESOURCE, DBA **FROM** 사용자 ;



**DCL**  
**맛보기**

**CREATE USER** C##TESTID **IDENTIFIED BY** 1234;  
**GRANT** CONNECT , RESOURCE , DBA **TO** C##TESTID;  
**REVOKE** CONNECT, RESOURCE, DBA **FROM** C##TESTID ;

# DDL

```
CREATE TABLE 회원(  
아이디 VARCHAR2(5) NOT NULL PRIMARY KEY,  
비번   VARCHAR2(10) NOT NULL,  
이름   VARCHAR2(20) NOT NULL ,  
주소   VARCHAR2(20) NOT NULL ,  
이메일 CHAR(20)  
);
```

```
ALTER TABLE 회원 ADD 성별 CHAR(1) ;
```

```
ALTER TABLE 회원 MODIFY 비번 VARCHAR2(15);
```

```
ALTER TABLE 회원 RENAME COLUMN 이메일 TO 이메일주소 ;
```

```
ALTER TABLE 회원 DROP COLUMN 성별 ;
```

```
DROP TABLE 회원;
```

## 문자형데이터

CHAR	고정길이	CHAR(SIZE) : CHAR(3) : 한글 1글자 저장가능
VARCHAR2	가변길이	VCHAR2(50) : 최대 50 의미함
NCHAR	고정길이	NCHAR(SIZE) : NCHAR(3): 한글 3글자 저장가능
CLOB		대용량데이터 (최대 4G)

## 숫자형데이터

NUMBER(4)	9999 최대길이4인 숫자,
NUMBER(6,2)	소수점 2자리를 포함하는 최대6자리 (소수점 2째자리에서 반올림)
NUMBER(6,-2)	소수점 -2째 자리에서 반올림 하는 최대6자리

## 날짜형데이터

DATE	년-월-일
TIMESTAMP	년-월-일:시:분:초

## 데이터 삽입하기(insert)

```
INSERT INTO 회원 VALUES ('angel' , '1234' , '정조' , '노원구 공릉' , 'victo88@naver.com');  
INSERT INTO 회원(아이디, 비번, 이름, 주소) VALUES( 'sung', '5432', '성덕임' , '서울시 경복궁');
```

## 데이터 변경하기(update)

```
UPDATE 회원  
SET      비번 = 'a!12' ;  
WHERE 아이디 = 'angel' ;
```

```
UPDATE 회원  
SET      주소 = '서울시 창덕궁'  
WHERE 아이디= 'sung'
```

## 삭제하기(delete)

```
DELETE FROM 회원  
WHERE 아이디='sung' ;
```

# 테이블만들기

오리온제과

상품테이블

영화예매사이트

영화정보테이블

SM기획사

연예인정보테이블



영화정보

테이블명

영화코드	영화제목	출연배우	장르	개봉일
VARCHAR2(5) 주키	VARCHAR2(30)	VARCHAR2(50)	VARCHAR2(10)	DATE
A1001	라라와 크리스마스요정		애니메이션	2021-12-15
A1002	킹스맨	랄프 파인즈 ,해리스 딕킨슨	액션	2021-12-22
A1003	매트릭스	키아누 리브스, 로렌스 피시번,	액션	2021-12-09
A1004	투모로우 아폴리캡스	조엘 버티, 제니퍼 리 위긴스	멜로	2021-12-01

```
CREATE TABLE 영화정보(  
    영화코드 VARCHAR2(5) NOT NULL PRIMARY KEY,  
    영화제목 VARCHAR2(30) ,  
    출연배우 VARCHAR2(50) ,  
    장르      VARCHAR2(30) ,  
    개봉일    DATE  
);
```

```
INSERT INTO 영화정보 VALUES('A1001', '라라와크리스마스요정', '', '애니메이션' , '2021-12-15');  
INSERT INTO 영화정보 VALUES('A1002', '킹스맨', '랄프 파인즈 ,해리스 딕킨슨' , '액션' , '2021-12-22');
```

--

[illegible]