

# DBMS

## Mysql (DDL,DML 등)

---

5주차 발표



DATABASE

# 데이터베이스

데이터베이스(DataBase, DB)란?

“여러 사람이 공유하고 사용할 목적으로 통합관리되는 정보의 집합”

논리적으로 연관된 하나 이상의 자료의 모음으로 그 내용을 고도로 구조화함으로써 검색 및 갱신의 효율을 높인다.

# 데이터베이스

## 데이터베이스 특징

- **실시간 접근성** : 사용자의 질의에 대하여 즉시 처리하여 응답한다.
- **계속적인 진화** : 삽입, 삭제, 갱신을 통하여 항상 최근의 정확한 데이터를 동적으로 유지한다.
- **동시공유** : 여러 사용자가 동시에 원하는 데이터를 공유할 수 있는 특징을 갖는다.
- **내용에 의한 참조** : 데이터베이스에 있는 데이터를 참조할 때 튜플의 주소나 위치가 아닌 사용자가 요구하는 데이터 내용에 따라 참조한다.
- **데이터 논리적 독립성** : 응용프로그램과 데이터베이스를 독립시킴으로써 데이터 논리적 구조를 변경시키더라도 응용프로그램은 변경되지 않는다.

# 데이터베이스

## 데이터베이스 구성

- **통합 데이터(integrated data)** : 중복을 배제하나 경우에 따라 불가피하게 중복을 허용하는 데이터
- **저장 데이터(stored data)** : 컴퓨터의 저장매체에 저장하여 관리하는 데이터
- **운영 데이터(operation data)** : 단순한 데이터의 집합이 아니라 그 조직의 기능을 수행하는 데 없어서는 안될 필수 데이터
- **공용 데이터(shared data)** : 어느 하나의 응용프로그램이나 응용시스템을 위한 데이터가 아니라, 그 조직의 여러 사용자와 여러 응용시스템들이 서로 다른 목적으로 공동 데이터



DBMS

# 데이터베이스 관리시스템

데이터베이스 관리 시스템(DBMS, DataBase Management System)이란?

“데이터베이스를 조작하는 별도의 소프트웨어로써, DBMS를 통해 데이터베이스를 관리하여 응용프로그램들이 데이터베이스를 공유하고, 사용할 수 있는 환경을 제공”

DBMS는 DB 내의 정보를 구성하는 컴퓨터 프로그램의 집합으로서 자료의 중복성을 제거하고 다른 특징들 중에 무결성, 일관성, 유용성을 보장하기 위해서 자료를 제거하고 관리하는 소프트웨어 체계이다.

# 데이터베이스 관리시스템

## DBMS의 기능

- **정의** : 데이터에 대한 형식, 구조, 제약조건들을 명세하는 기능
- **구축** : DBMS가 관리하는 기억장치에 데이터를 저장하는 기능
- **조작** : 특정한 데이터를 검색하기 위한 질의, 데이터베이스의 갱신, 보고서 생성 기능
- **공유** : 여러 사용자와 프로그램이 데이터베이스에 동시에 접근하도록 하는 기능
- **보호** : 하드웨어나 소프트웨어의 오동작 또는 권한이 없는 악의적인 접근으로부터 시스템 보호기능
- **유지보수** : 시간이 지남에 따라 변화하는 요구사항을 반영할 수 있도록 하는 기능



# 데이터베이스 관리시스템

## DBMS의 종류

DBMS Ranking, March 2022

Rank			DBMS	Database Model	Score		
Mar 2022	Feb 2022	Mar 2021			Mar 2022	Feb 2022	Mar 2021
1.	1.	1.	Oracle +	Relational, Multi-model i	1251.32	-5.51	-70.42
2.	2.	2.	MySQL +	Relational, Multi-model i	1198.23	-16.45	-56.59
3.	3.	3.	Microsoft SQL Server +	Relational, Multi-model i	933.78	-15.27	-81.52
4.	4.	4.	PostgreSQL +	Relational, Multi-model i	616.93	+7.54	+67.64
5.	5.	5.	MongoDB +	Document, Multi-model i	485.66	-2.98	+23.27
6.	6.	↑ 7.	Redis +	Key-value, Multi-model i	176.76	+0.96	+22.61
7.	7.	↓ 6.	IBM Db2	Relational, Multi-model i	162.15	-0.73	+6.14
8.	8.	8.	Elasticsearch	Search engine, Multi-model i	159.95	-2.35	+7.61
9.	9.	↑ 10.	Microsoft Access	Relational	135.43	+4.17	+17.29
10.	10.	↓ 9.	SQLite +	Relational	132.18	+3.81	+9.54

# 데이터베이스 관리시스템

## 관계형 DBMS vs NoSQL

	관계형 DBMS	NoSQL
확장성	수직적 확장	수평적 확장
일관성	스키마 O	스키마 X

# 데이터베이스 종류

## Oracle

- 다양한 운영체제에 설치 가능
- MySQL, MSSQL보다 빅데이터 처리 용이
- 가장 널리 사용되는 관계형 DBMS
- 폐쇄적인 운영으로 인해 오픈소스 사용 불가



# 데이터베이스 종류

## MySQL

- 다양한 운영체제에 설치 가능
- 오픈소스로 이루어져 있는 무료 프로그램
- 표준 SQL 형식 사용
- 상업적으로 사용 시 비용 발생, 다른 관계형 DBMS보다 가격 측면에서 경쟁력



# 데이터베이스 종류

## MSSQL

- 다양한 운영체제에 사용 가능하지만, 윈도우에 최적화
- 리눅스를 제외한 다른 운영체제에서는 오픈소스 사용 불가
- 표준 SQL을 몰라도 쉽게 관리 가능
- C#과 가장 높은 호환성을 갖고 있음



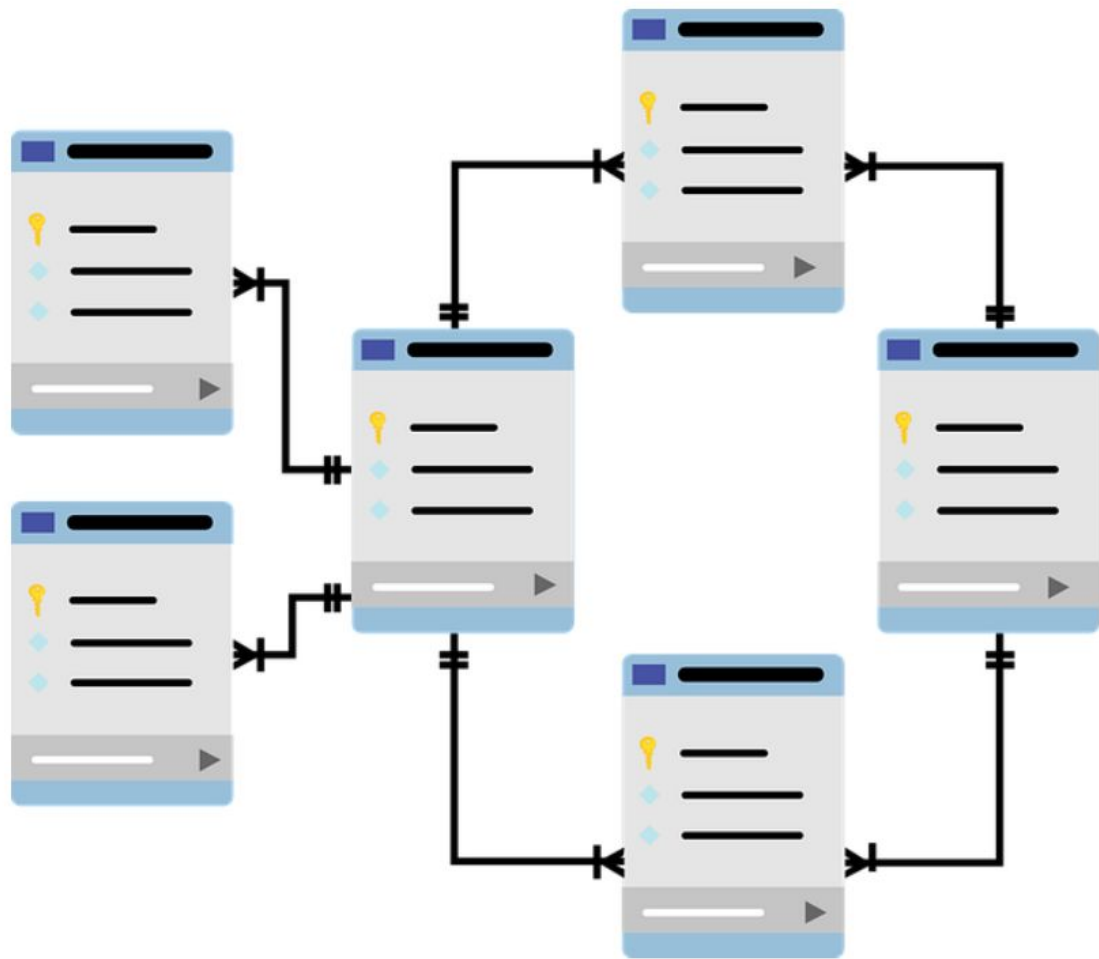
# 데이터베이스 종류

## MariaDB

- MySQL과 매우 유사한 관계형 DBMS
- MySQL에 비해 속도가 4배정도 빠름
- 불확실한 라이선스문제를 해결하기 위해
- C++과 가장 높은 호환성을 갖고 있음

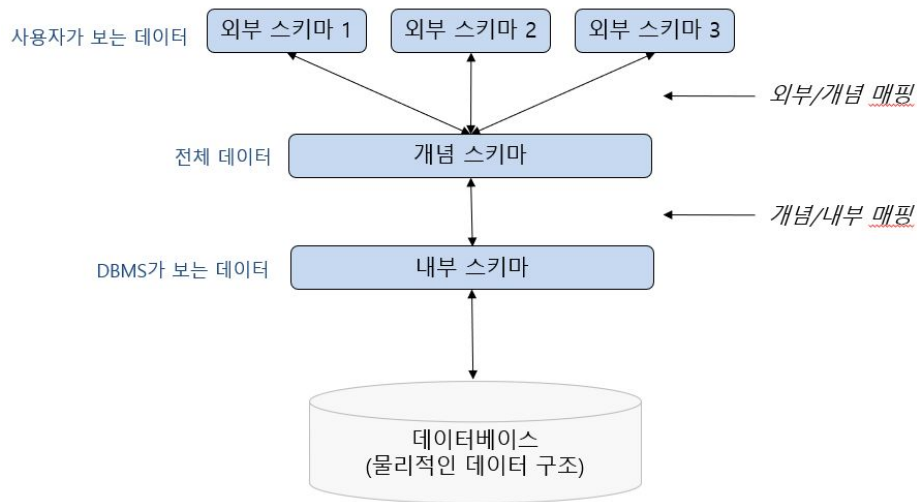


# MySQL 기본구조





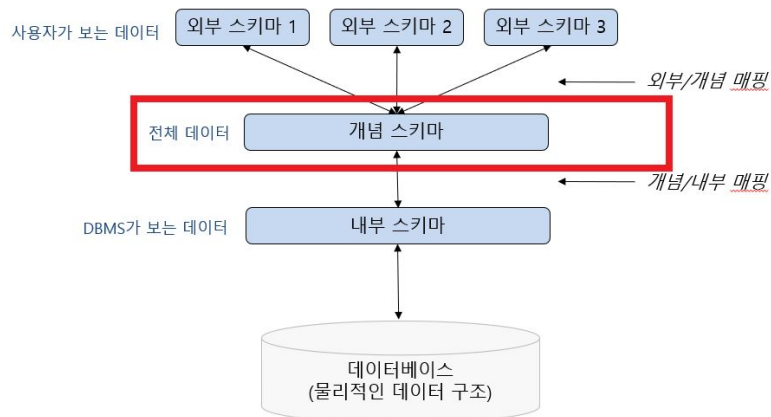
# 3단계 데이터 베이스 구조



스키마란?

“데이터베이스 내에 어떤 구조로 데이터가 저장되는지를 나타내는 데이터베이스 구조”

# 개념 스키마



1. 전체 데이터베이스의 정의를 의미한다.
2. 통합 조직별로 하나만 존재하며 DBA가 관리한다.
3. 하나의 데이터베이스에는 하나의 개념 스키마 (conceptual schema)가 있다.

※ DBA : Database Administration 데이터베이스 관리자

# 개념 스키마 예시

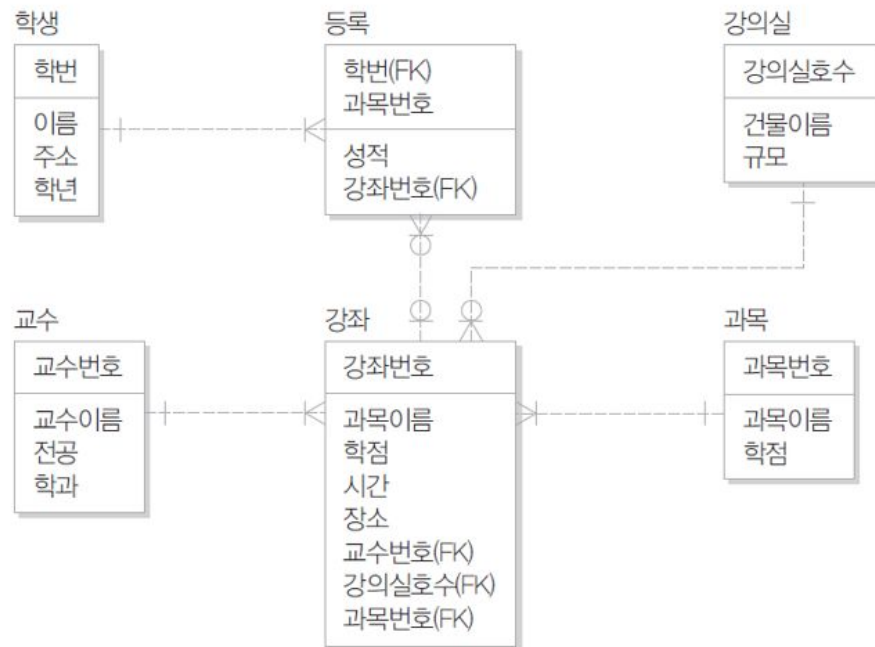
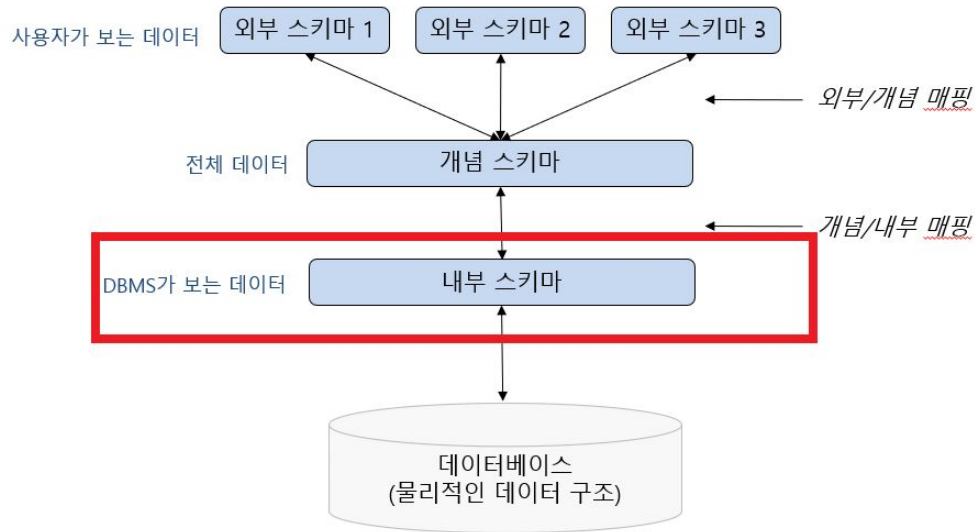


그림 1-28 수강신청 데이터베이스의 개념 스키마

# 내부 스키마



1. 물리적 저장 장치에 데이터베이스가 실제로 저장되는 방법의 표현
2. 내부 스키마(internal schema)는 하나
3. 인덱스, 데이터 레코드의 배치 방법, 데이터 압축 등에 관한 사항이 포함된다.

# 내부 스키마 예시

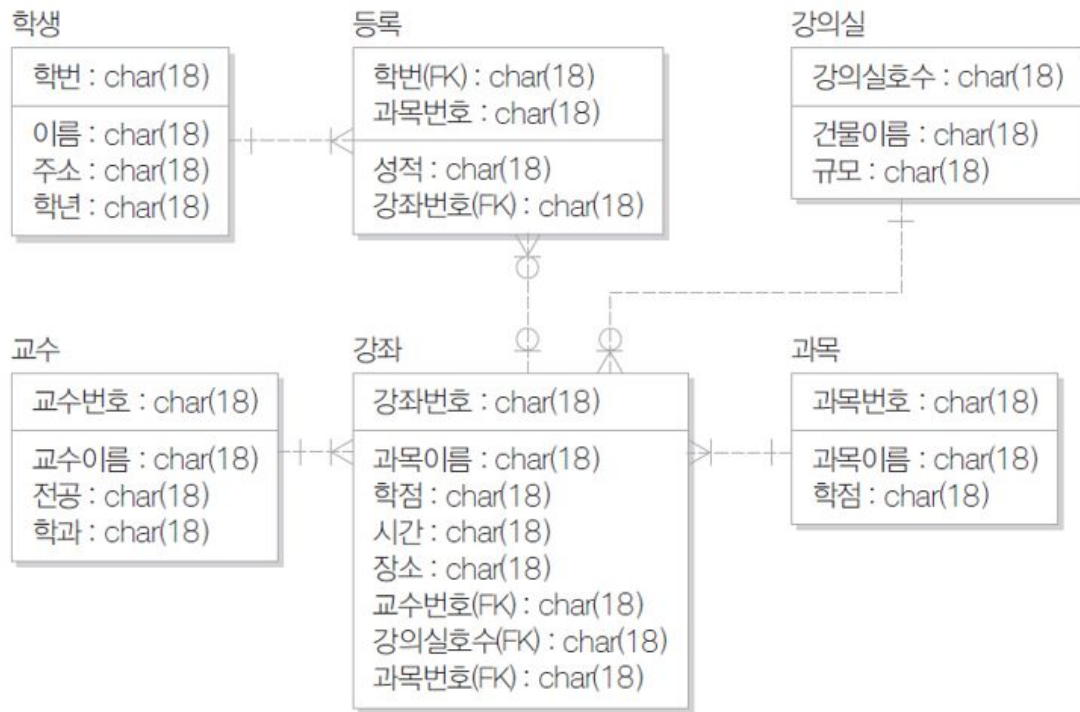
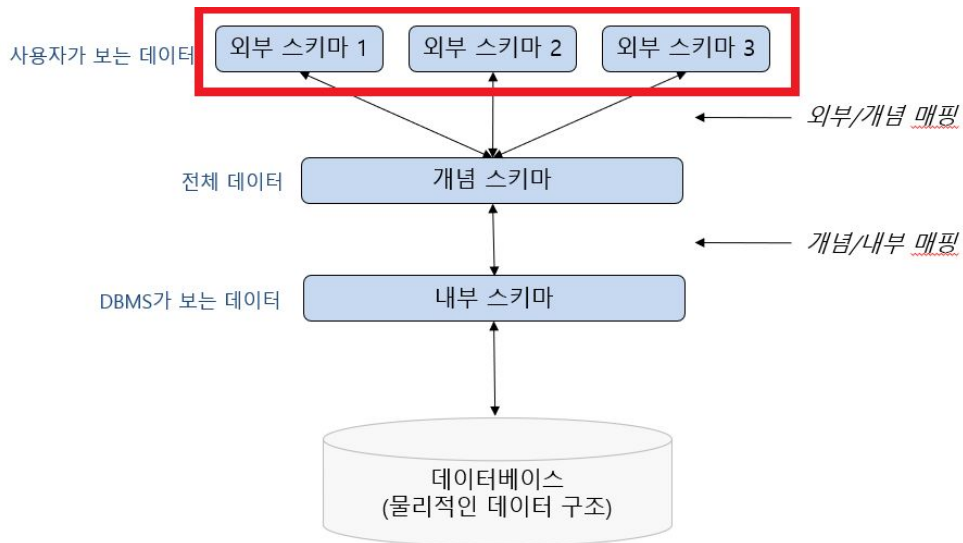


그림 1-31 수강신청 데이터베이스의 내부 스키마

# 외부 스키마



1. 일반 사용어나 응용 프로그래머가 접근하는 계층으로 전체 데이터베이스 중에서 하나의 논리적인 부분을 의미한다.
2. 여러 개의 외부 스키마(external schema)가 있을 수 있다.
3. 서브 스키마(sub schema)라고도 하며, 뷰(view)의 개념이다.

# 외부 스키마 예시

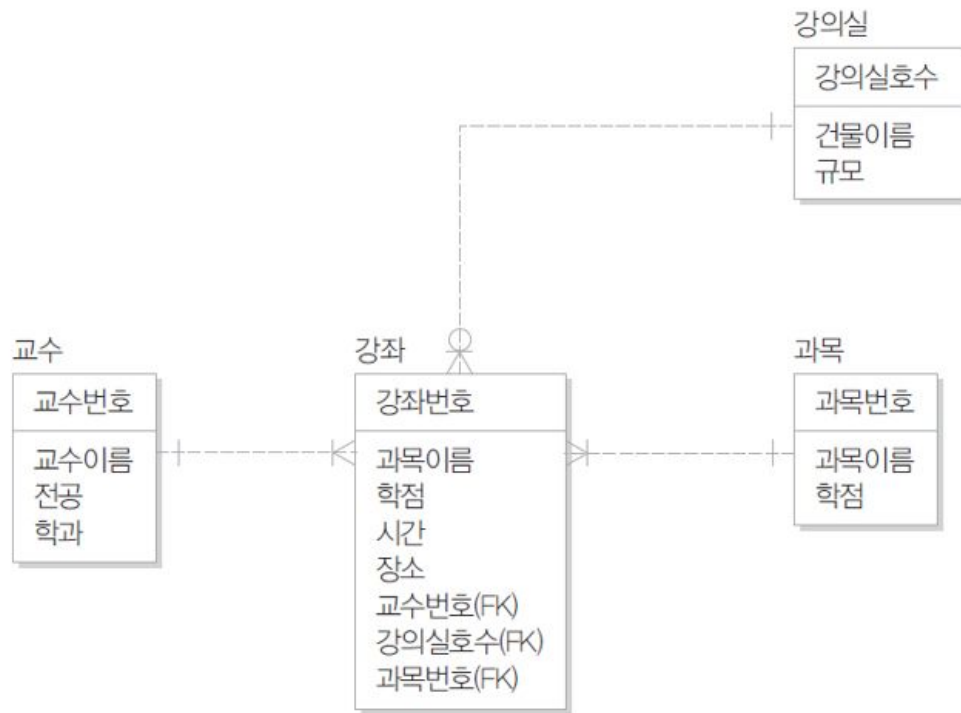
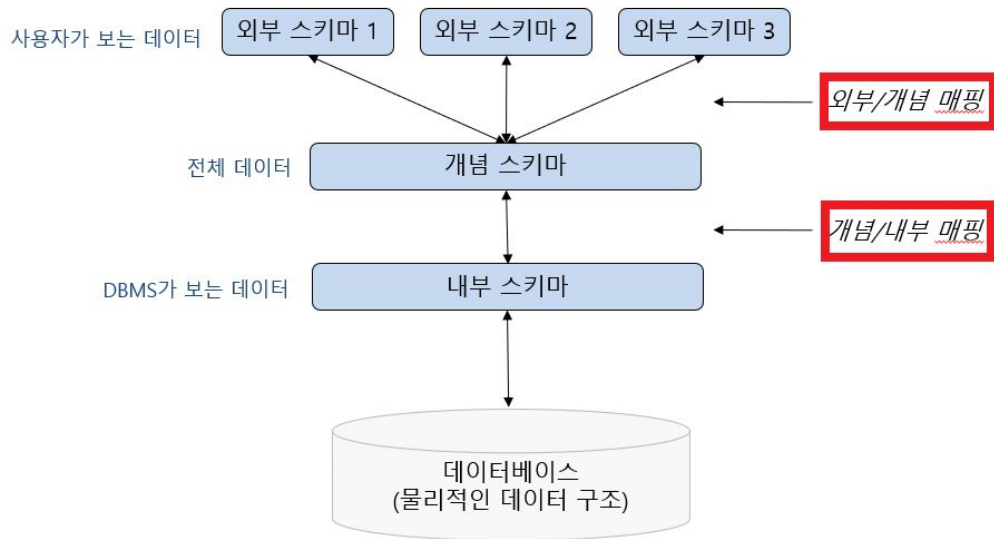


그림 1-30 시간표 담당 부서에서 필요한 데이터베이스(외부 스키마2)

# 매핑



## 외부/개념 매핑?

외부 스키마의 데이터가 개념 스키마의 어느 부분에 해당되는지 대응시킨다.

## 개념/내부 매핑?

개념 스키마의 데이터가 내부 스키마의 물리적 장치 어디에 어떤 방법으로 저장되는지 대응시킨다.



# 데이터 독립성

## 논리적 데이터 독립성(logical data independence)

- 외부 단계(외부 스키마)와 개념 단계(개념 스키마)사이의 독립성
- 개념 스키마가 변경되어도 외부 스키마에는 영향을 미치지 않도록 지원

## 물리적 데이터 독립성(physical data independence)

- 개념 단계(개념 스키마)와 내부 단계(내부 스키마)사이의 독립성
- 저장장치 구조 변경과 같이 내부 스키마가 변경되어도 개념 스키마에 영향을 미치지 않도록 지원

# 테이블 기본구조



MySQL Workbench

Local instance MySQL80 x madang1 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

▼ madang

▼ Tables

author

book

customer

imported\_book

orders

topic

Administration Schemas

Information

No object selected

Query 1 x SQL File 3\*

Limit to 1000 rows

1 select \* from book;

2 select \* from orders;

3 select \* from customer;

4 select \* from orders where custid = 1;

5 select saleprice, orderdate, bookid from orders;

6 select \* from customer, orders; /\* 카디전프로젝트 \*/

7

8 select name, address from orders, customer where customer.custid = orders.custid and bookid = 1;

Result Grid

Filter Rows:

Edit: Export/Imports: Wrap Cell Content:

bookid bookname publisher price

1 축구역사 굿스포츠 7000

2 축구하는여자 나무수 13000

3 축구의이해 대한미디어 22000

4 골프바이블 대한미디어 35000

5 피겨교본 굿스포츠 8000

6 역도단계별기술 굿스포츠 6000

7 야구의추억 이상미디어 20000

8 야구를부탁해 이상미디어 13000

9 올림픽이야기 삼성당 7500

10 OlympicChampions Pearson 13000

NULL NULL NULL NULL

book 1 x

Apply Revert

Context Help Snippets

Output

Action Output

# Time Action Message Duration / Fetch

1 14:13:37 select \* from book LIMIT 0, 1000 10 row(s) returned 0.032 sec / 0.000 sec

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

	bookid	bookname	publisher	price
▶	1	축구의 역사	굿스포츠	7000
	2	축구아는 여자	나무수	13000
	3	축구의 이해	대한미디어	22000
	4	골프 바이블	대한미디어	35000
	5	피겨 교본	굿스포츠	8000
	6	역도 단계별기술	굿스포츠	6000
	7	야구의 추억	이상미디어	20000
	8	야구를 부탁해	이상미디어	13000
	9	올림픽 이야기	삼성당	7500
	10	Olympic Champions	Pearson	13000
▲	NULL	NULL	NULL	NULL

# 릴레이션



그림 2-3 도서 릴레이션

“MySQL 에서 요구하는 조건을 만족하는 테이블”

# 속성(Attribute)



그림 2-3 도서 릴레이션

1. 속성의 유일한 식별을 위해 속성의 명칭은 유일해야 하지만, 속성을 구성하는 값은 동일한 값이 있을 수 있다.  
※ 키로 정의된 속성은 동일한 값을 가질 수 없다.
2. 한 속성에 속한 열은 모두 그 속성에서 정의한 도메인 값만 가질 수 있다. (INTEGER로 정의했으면 정수만 가능)

# 스키마(Schema)



그림 2-3 도서 릴레이션

1. 릴레이션에 어떤 정보가 담길지를 정의한다.
2. 릴레이션이 몇 개의 속성을 가지는가를 나타내기 위해 차수(degree)라는 용어를 사용한다.



# 튜플(Tuple)



그림 2-3 도서 릴레이션

1. 릴레이션 내의 중복된 튜플은 허용하지 않는다.
2. 튜플의 순서는 상관없다.

# 인스턴스(Instance)



그림 2-3 도서 릴레이션

1. 스키마, 첫 행을 제외한 나머지 행들 안 저장된 데이터들을 말한다.

# 릴레이션 예시

```
MySQL 8.0 Command Line Client
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SELECT * FROM Users
-> ;
ERROR 1046 (3D000): No database selected
mysql> SELECT * FROM Users;
ERROR 1046 (3D000): No database selected
mysql> use instagramclonecoding;
Database changed
mysql> SELECT * FROM Users;
+----+-----+-----+-----+
| userid | passwords | name | profileimg |
+----+-----+-----+-----+
| 62hoon99 | 123456 | 유기훈 | ../images/profile_img1.jpg |
+----+-----+-----+-----+
1 row in set (0.01 sec)

mysql> SELECT * FROM Post;
+----+-----+-----+-----+-----+-----+
| id | userid | name | img | text | likes |
+----+-----+-----+-----+-----+-----+
| 1 | 62hoon99 | 유기훈 | ../images/feed_img1.JPG | 역시 베라는 민초지-! | 38 |
| 2 | 62hoon99 | 유기훈 | ../images/feed_img2.JPG | 카페 갔다왔다---! | 32 |
| 3 | 62hoon99 | 유기훈 | ../images/feed_img3.JPG | 갈치랑..ㅎ | 48 |
| 4 | 62hoon99 | 유기훈 | ../images/feed_img4.JPG | 제부도가 좋아서 투따봉 | 51 |
+----+-----+-----+-----+-----+-----+
4 rows in set (0.01 sec)

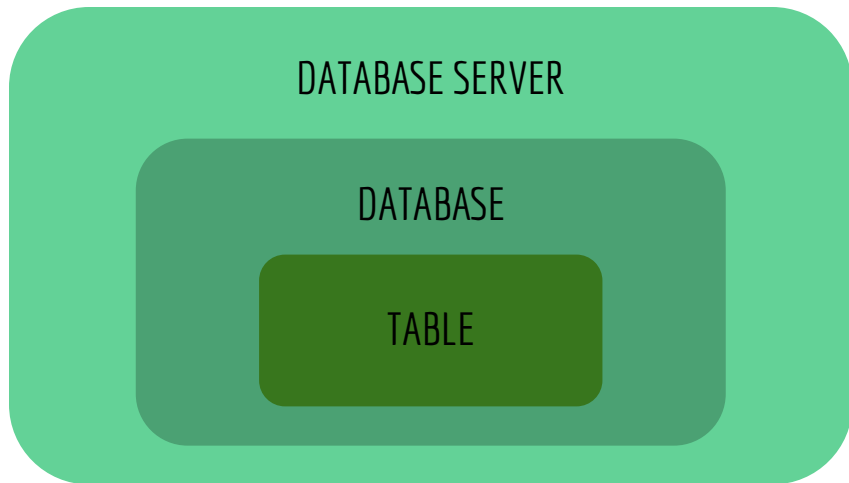
mysql>
```

DDL

# DDL

DDL이란? “Data Define Language”

- 데이터 정의어로서 테이블과 같은 데이터 구조를 정의하는데 사용
- 데이터 구조를 정의함에 있어서 발생할 수 있는 이슈로 생성, 변경, 이름 변경, 삭제의 명령어도 함께 포함



# DDL - create

create 명령어는? “데이터베이스를 생성하거나 테이블을 생성하는 명령어”

```
1  -- 데이터베이스 생성 명령어
2  CREATE DATABASE [dbname];
3
4  -- 데이터베이스 생성 및 속성 설정 명령어
5  CREATE DATABASE [dbname] CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
6  -- 이모지 등 문자 깨지지 않도록(character set ~~)
```

```
1  -- 테이블 생성 명령어
2  CREATE TABLE [tablename] (
3      [column_name1] INT PRIMARY KEY AUTO_INCREMENT,
4      [column_name2] VARCHAR(255) NOT NULL,
5      [column_name3] DATETIME NOT NULL
6  )CHARSET=utf8;
7  -- 한글 등 문자 깨짐 방지(charset=utf8)
```

→ 테이블 생성 시 컬럼명, 데이터타입, 제약 조건 순으로 생성

# Mysql - setting & create DB

- 1 -- 데이터베이스 생성 명령어
- 2 CREATE DATABASE [dbname];

cd : change directory

```
C:\Users\esthel>cd C:\Bitnami\wampstack-8.1.2-0\mysql\bin
```

```
C:\Bitnami\wampstack-8.1.2-0\mysql\bin>mysql -uroot -p
```

Enter password: \*\*\*\*\*

Welcome to the MariaDB monitor. Commands end with ; or \g.

Your MariaDB connection id is 10

Server version: 10.4.22-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

```
MariaDB [(none)]> SHOW DATABASES;
```

Database
information_schema
mysql
opentutorials
performance_schema
test

5 rows in set (0.001 sec)

DB에 루트 권한으로 접속

DB 이름

```
MariaDB [(none)]> CREATE DATABASE week5_test;
```

Query OK, 1 row affected (0.004 sec)

```
MariaDB [(none)]> SHOW DATABASES;
```

Database
information_schema
mysql
opentutorials
performance_schema
test
<u>week5_test</u>

6 rows in set (0.001 sec)

```
MariaDB [(none)]>
```

# Mysql - create TABLE

```
1  -- 테이블 생성 명령어
2  CREATE TABLE [tablename] (
3      [column_name1] INT PRIMARY KEY
4      AUTO_INCREMENT,
5      [column_name2] VARCHAR(255) NOT NULL,
6      [column_name3] DATETIME NOT NULL
7  )CHARSET=utf8;
8  -- 한글 등 문자 깨짐 방지(charset=utf8)
```

- **id, description, created** : column\_name
- **PRIMARY KEY** : id는 고유하게 처리하기 위함  
해당 column은 값을 다 다르게 가짐
- **AUTO\_INCREMENT** : 자동으로 증가
- **DATETIME** : 날짜 시간 입력받기  
(9999-12-31 23:59:59)
- **NOT NULL** : 빈값을 허용하지 않음

→ 테이블 생성 시 컬럼명, 데이터타입, 제약 조건 순으로 생성

```
MariaDB [(none)]> USE week5_test;
Database changed
MariaDB [week5_test]> SHOW TABLES;
Empty set (0.001 sec)

MariaDB [week5_test]> CREATE TABLE abc(
    -> id INT PRIMARY KEY AUTO_INCREMENT,
    -> description VARCHAR(255) NOT NULL,
    -> created DATETIME NOT NULL
    -> );
Query OK, 0 rows affected (0.029 sec)

MariaDB [week5_test]> SHOW TABLES;
+-----+
| Tables_in_week5_test |
+-----+
| abc                   |
+-----+
1 row in set (0.001 sec)

MariaDB [week5_test]>
```



# DDL - alter

alter 명령어는? “테이블의 내용을 수정할 수 있도록 하는 명령어”

```
1  -- 테이블에 컬럼 추가하기
2  ALTER TABLE [table_name] ADD COLUMN [column_name] [column_type];
3
4  -- 테이블의 컬럼 타입 변경하기
5  ALTER TABLE [table_name] MODIFY COLUMN [column_name] [column_type];
6
7  -- 테이블의 컬럼 이름 변경하기
8  ALTER TABLE [table_name] CHANGE COLUMN [old_column_name] [new_column_name] [new_column_type];
9
10 -- 테이블의 컬럼 삭제하기
11 ALTER TABLE [table_name] DROP COLUMN [column_name];
12
13 -- 테이블에 컬럼 인덱스 주기
14 ALTER TABLE [table_name] DROP INDEX [index_name];
15
16 -- 테이블에 PRIMARY KEY 만들기
17 ALTER TABLE [table_name] ADD PRIMARY KEY( column_name_on_this_table );
18
19 -- 테이블에 PRIMARY KEY 삭제하기
20 ALTER TABLE [table_name] DROP PRIMARY KEY;
21
22 -- 테이블 명 바꾸기
23 ALTER TABLE [old_table_name] RENAME [new_table_name];
```

# Mysql - alter

- 1 -- 테이블에 컬럼 추가하기
- 2 ALTER TABLE [table\_name] ADD COLUMN [column\_name] [column\_type];

해당 테이블 구조 보여줌

```
MariaDB [week5_test]> DESC abc;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
description	varchar(255)	NO		NULL	
created	datetime	NO		NULL	

3 rows in set (0.012 sec)

```
MariaDB [week5_test]> ALTER TABLE abc ADD COLUMN text VARCHAR(100);  
Query OK, 0 rows affected (0.024 sec)  
Records: 0 Duplicates: 0 Warnings: 0
```

```
MariaDB [week5_test]> DESC abc;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
description	varchar(255)	NO		NULL	
created	datetime	NO		NULL	
text	varchar(100)	YES		NULL	

4 rows in set (0.014 sec)

# Mysql - alter

- 1 -- 테이블의 컬럼 삭제하기
- 2 ALTER TABLE [table\_name] DROP COLUMN [column\_name];

```
MariaDB [week5_test]> DESC abc;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
description	varchar(255)	NO		NULL	
created	datetime	NO		NULL	
text	varchar(100)	YES		NULL	

```
4 rows in set (0.014 sec)
```

```
MariaDB [week5_test]> ALTER TABLE abc DROP COLUMN text;
```

```
Query OK, 0 rows affected (0.017 sec)
```

```
Records: 0 Duplicates: 0 Warnings: 0
```

```
MariaDB [week5_test]> DESC abc;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
description	varchar(255)	NO		NULL	
created	datetime	NO		NULL	

```
3 rows in set (0.015 sec)
```

# DDL - rename

rename 명령어는? “테이블의 이름을 변경하는 명령어”

```
1  -- 테이블 이름 변경 명령어
2  RENAME TABLE [old_table_name] TO [new_table_name];
3
4  -- 여러 테이블 이름 변경 명령어
5  RENAME TABLE [old_table_name1] TO [new_table_name1], [old_table_name2] TO [new_table_name2];
```

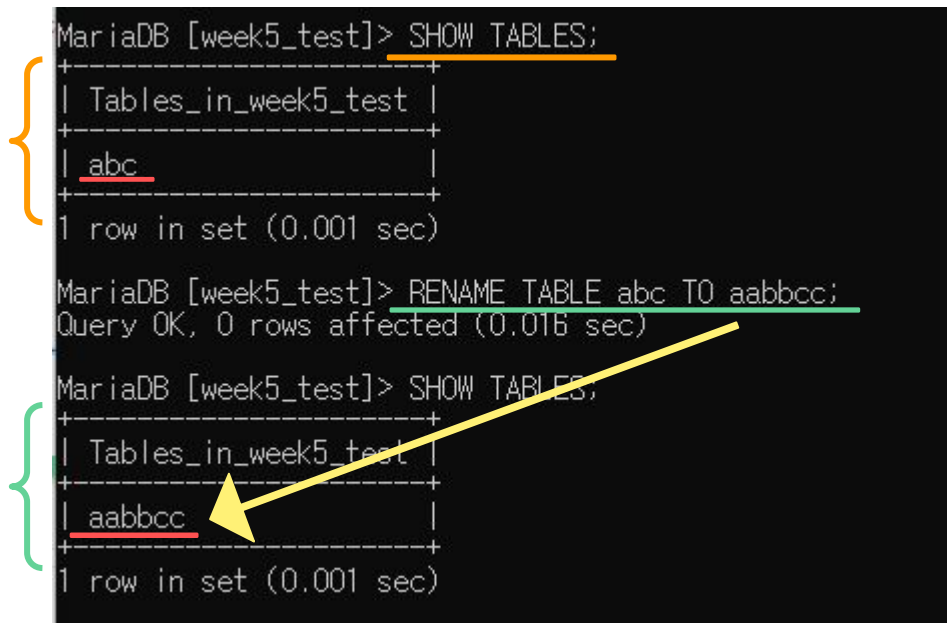
# Mysql - rename

```
1  -- 테이블 이름 변경 명령어
2  RENAME TABLE [old_table_name] TO [new_table_name];
```

```
MariaDB [week5_test]> SHOW TABLES;
+-----+
| Tables_in_week5_test |
+-----+
| abc                  |
+-----+
1 row in set (0.001 sec)

MariaDB [week5_test]> RENAME TABLE abc TO aabbcc;
Query OK, 0 rows affected (0.016 sec)

MariaDB [week5_test]> SHOW TABLES;
+-----+
| Tables_in_week5_test |
+-----+
| aabbcc              |
+-----+
1 row in set (0.001 sec)
```



# DDL - drop

drop 명령어는? “데이터베이스를 삭제하거나 테이블을 삭제하는 명령어”

```
1  -- 데이터베이스 삭제 명령어
2  DROP DATABASE [dbname];
```

```
1  -- 테이블 삭제 명령어
2  DROP TABLE [table_name];
```

# Mysql - drop TABLE

```
1  -- 테이블 삭제 명령어  
2  DROP TABLE [table_name];
```

```
MariaDB [week5_test]> SHOW TABLES;  
+-----+  
| Tables_in_week5_test |  
+-----+  
| abc                   |  
+-----+  
1 row in set (0.001 sec)  
  
MariaDB [week5_test]> DROP TABLE abc;  
Query OK, 0 rows affected (0.015 sec)  
  
{ MariaDB [week5_test]> SHOW TABLES;  
Empty set (0.001 sec)
```

# Mysql - drop DB

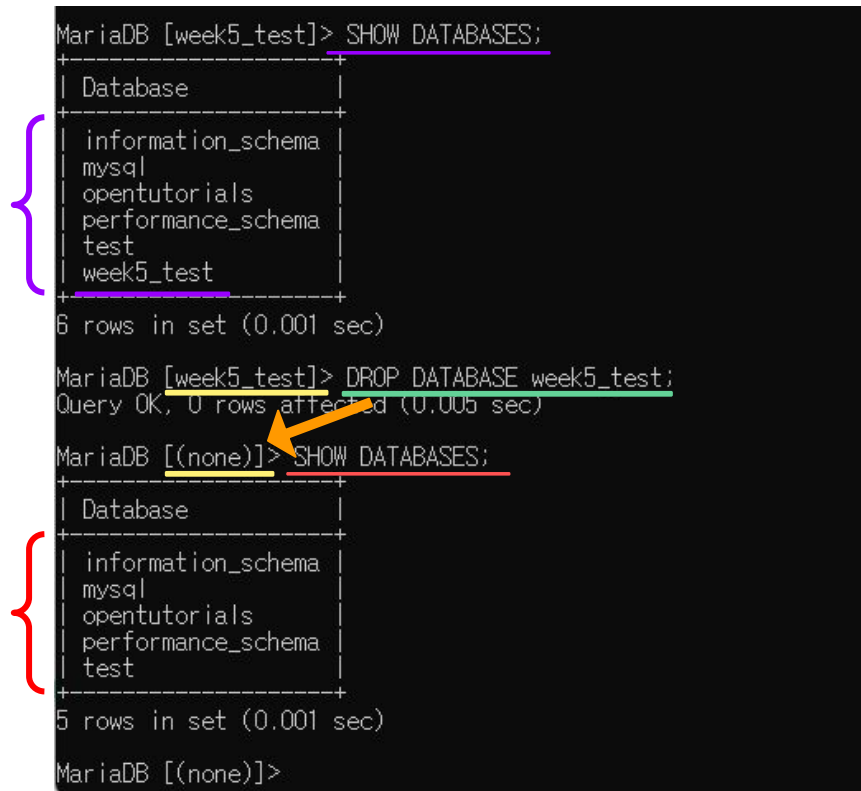
```
1  -- 데이터베이스 삭제 명령어
2  DROP DATABASE [dbname];
```

```
MariaDB [week5_test]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql       |
| opentutorials |
| performance_schema |
| test        |
| week5_test    |
+-----+
6 rows in set (0.001 sec)

MariaDB [week5_test]> DROP DATABASE week5_test;
Query OK, 0 rows affected (0.005 sec)

MariaDB [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql       |
| opentutorials |
| performance_schema |
| test        |
+-----+
5 rows in set (0.001 sec)

MariaDB [(none)]>
```







DML

# DML

DML이란? “Data Manipulation Language”

- 데이터를 조작하기 위해 사용
- 데이터를 검색(SELECT), 삽입(INSERT), 수정(UPDATE), 삭제(DELETE)할 수 있음

# DML - SELECT

SELECT 명령어는? "데이터 검색"

SELECT [ALL | DISTINCT] 속성이름

FROM 테이블이름

[WHERE 검색조건]

[GROUP BY 속성이름]

[HAVING 검색조건]

[ORDER BY 속성이름 [ASC | DESC]]

```

1  -- 테이블 내 모든 데이터 조회
2  SELECT * FROM [테이블이름];
3  SELECT * FROM book;
4
5  -- 테이블 내 원하는 조건을 만족하는 데이터 가져오기
6  SELECT * FROM [테이블이름] WHERE [조건]
7  SELECT * FROM book WHERE bookname='축구의 역사';           //정확하게 '축구의 역사'일때만 가져옴
8  SELECT * FROM book WHERE bookname LIKE '%축구%';           //'축구'가 포함되기만하면 가져옴
9
10 -- 테이블 내 일부 데이터 조회
11 SELECT * FROM [테이블이름] LIMIT [개수];
12 SELECT * FROM [테이블이름] LIMIT [개수] OFFSET [위치];
13 SELECT * FROM [테이블이름] LIMIT [위치], [개수];
14 SELECT * FROM book LIMIT 3, 5;

```

	bookid	bookname	publisher	price
▶	1	축구의 역사	굿스포츠	7000
	2	축구하는 여자	나무수	13000
	3	축구의 이해	대한미디어	22000
	4	골프 바이블	대한미디어	35000
	5	피겨 교본	굿스포츠	8000
	6	역도 단계별기술	굿스포츠	6000
	7	야구의 추억	이상미디어	20000
	8	야구를 부탁해	이상미디어	13000
	9	올림픽 이야기	삼성당	7500
	10	Olympic Champions	Pearson	13000

	bookid	bookname	publisher	price
▶	1	축구의 역사	굿스포츠	7000

	bookid	bookname	publisher	price
▶	1	축구의 역사	굿스포츠	7000
	2	축구하는 여자	나무수	13000
	3	축구의 이해	대한미디어	22000

	bookid	bookname	publisher	price
▶	4	골프 바이블	대한미디어	35000
	5	피겨 교본	굿스포츠	8000
	6	역도 단계별기술	굿스포츠	6000
	7	야구의 추억	이상미디어	20000
	8	야구를 부탁해	이상미디어	13000

```

1  -- 테이블 내 속성에 따라 정렬하여 가져오기
2  SELECT * FROM [테이블이름] ORDER BY 속성 [ASC | DESC];
3  SELECT * FROM book ORDER BY price (ASC);    //기본 오름차순정렬(ASC없어도 됨)
4  SELECT * FROM book ORDER BY price DESC;      //내림차순 정렬
5  SELECT * FROM book ORDER BY price, name;     //price > name
6
7  -- 테이블 내 데이터 중복 제거해서 조회
8  SELECT DISTINCT * FROM [테이블명];
9  SELECT DISTINCT publisher FROM book;

```

	bookid	bookname	publisher	price
▶	6	역도 단계별기술	굿스포츠	6000
	1	축구의 역사	굿스포츠	7000
	9	올림픽 이야기	삼성당	7500
	5	피겨 교본	굿스포츠	8000
	2	축구하는 여자	나무수	13000
	8	야구를 부탁해	이상미디어	13000
	10	Olympic Champions	Pearson	13000
	7	야구의 추억	이상미디어	20000
	3	축구의 이해	대한미디어	22000
	4	골프 바이블	대한미디어	35000

	bookid	bookname	publisher	price
▶	6	역도 단계별기술	굿스포츠	6000
	1	축구의 역사	굿스포츠	7000
	9	올림픽 이야기	삼성당	7500
	5	피겨 교본	굿스포츠	8000
	10	Olympic Champions	Pearson	13000
	8	야구를 부탁해	이상미디어	13000
	2	축구하는 여자	나무수	13000
	7	야구의 추억	이상미디어	20000
	3	축구의 이해	대한미디어	22000
	4	골프 바이블	대한미디어	35000

	publisher
▶	굿스포츠
	나무수
	대한미디어
	대한미디어
	굿스포츠
	굿스포츠
	이상미디어
	이상미디어
	삼성당
	Pearson

	publisher
▶	굿스포츠
	나무수
	대한미디어
	이상미디어
	삼성당
	Pearson

```

1  -- 집계함수
2  count() / avg() / max() / min() / sum()
3
4  -- 테이블 내 데이터 그룹핑 조회
5  SELECT * FROM [테이블명] GROUP BY [속성];
6  SELECT * FROM [테이블이름] GROUP BY [속성] HAVING [조건];
7
8  ex)
9  SELECT avg(price) as '평균가격' FROM book;
   SELECT publisher, avg(price) as '출판사별 평균가격' FROM book GROUP BY publisher;
   SELECT publisher, avg(price) as '출판사별 평균가격' FROM book GROUP BY publisher HAVING avg(price)<10000;

```

	평균가격
▶	14450.0000

	publisher	출판사별 평균가격
▶	굿스포츠	7000.0000
	나무수	13000.0000
	대한미디어	28500.0000
	이상미디어	16500.0000
	삼성당	7500.0000
	Pearson	13000.0000

	publisher	출판사별 평균가격
▶	굿스포츠	7000.0000
	삼성당	7500.0000

# DML - INSERT

INSERT 명령어는? “데이터 삽입”

```
1  -- 테이블 내 한 튜플 삽입
2  INSERT INTO [테이블이름](속성1, 속성2) VALUES ('값1', '값2');
3  INSERT INTO [테이블이름] SET [속성1]=[값1], [속성2]=[값2]...;
4
5  ex)
6  INSERT INTO book(bookid, bookname, publisher, price) VALUES (14, 'OYR', 'sejong', 10000);
7  INSERT INTO book SET bookid=14, bookname='OYR', publisher='sejong', price=10000;
```

	bookid	bookname	publisher	price
▶	1	축구의 역사	굿스포츠	7000
	2	축구하는 여자	나무수	13000
	3	축구의 이해	대한미디어	22000
	4	골프 바이블	대한미디어	35000
	5	피겨 교본	굿스포츠	8000
	6	역도 단계별기술	굿스포츠	6000
	7	야구의 추억	이상미디어	20000
	8	야구를 부탁해	이상미디어	13000
	9	올림픽 이야기	삼성당	7500
	10	Olympic Champions	Pearson	13000
	14	OYR	sejong	10000

# DML - UPDATE

UPDATE 명령어는? “데이터 수정”

```
1  -- 테이블 내 튜플 한개 수정
2  UPDATE [테이블이름] SET [속성] = [값] (WHERE [조건]);
3
4  ex)
5  UPDATE book SET price=9000 WHERE bookid=14;
```

	bookid	bookname	publisher	price
▶	1	축구의 역사	굿스포츠	7000
	2	축구하는 여자	나무수	13000
	3	축구의 이해	대한미디어	22000
	4	골프 바이블	대한미디어	35000
	5	피겨 교본	굿스포츠	8000
	6	역도 단계별기술	굿스포츠	6000
	7	야구의 추억	이상미디어	20000
	8	야구를 부탁해	이상미디어	13000
	9	올림픽 이야기	삼성당	7500
	10	Olympic Champions	Pearson	13000
	14	OYR	sejong	9000



# DML - DELETE

DELETE 명령어는? “데이터 삭제”

```
1  -- 테이블 내 튜플 한개 삭제
2  DELETE FROM [테이블이름] (WHERE [조건])
3
4  ex)
5  DELETE FROM book;           //book테이블의 전체값 삭제
   DELETE FROM book WHERE bookid=14;    //원하는 조건의 데이터만 삭제
```

감사합니다