

# 삼성청년 SW·AI아카데미

DB

## <알림>

본 강의는 삼성청년SW·AI아카데미의 콘텐츠로  
보안서약서에 의거하여  
강의 내용을 어떠한 사유로도 임의로 복사, 촬영,  
녹음, 복제, 보관, 전송하거나  
허가 받지 않은 저장매체를  
이용한 보관, 제3자에게 누설, 공개,  
또는 사용하는 등의 행위를 금합니다.

# Day1-1. DB 트렌드와 MySQL 개요

# 챕터의 포인트

- Database의 필요성
- Database 트렌드
- MySQL 구조
- 데이터타입 종류
- 테이블 생성

## Database의 필요성

## DB

- Database
- 데이터의 기지 (Base), Data들의 묶음
- 데이터의 집합을 뜻하는 개념적 단어

## DBMS

- Database (Data들) 을 관리하기 도구들을 모아 둔 시스템
- MySQL / Oracle / Mongo DB / PostgreSQL 등 존재
- 프로그래밍에서 DB라고 부르는 것은 DBMS를 포함

## DBMS를 통상 DB라고 부른다.

- 대표적인 DB 종류

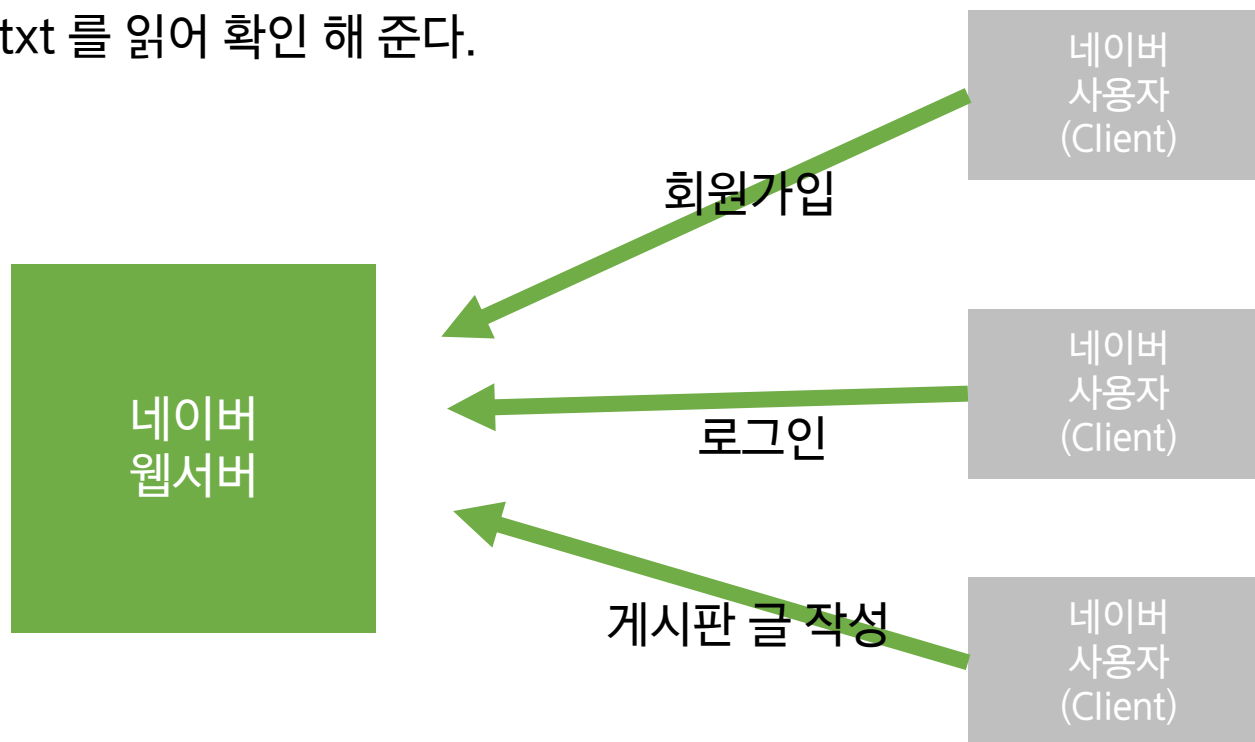
- Oracle사가 소유한 MySQL - 오픈소스 업계 1위
- Oracle사가 소유한 Oracle - 유료(기업대상) 업계 1위

The Oracle logo consists of the word "ORACLE" in white, uppercase, sans-serif font, centered within a solid red rectangular background.

## 사용자 데이터를 DB가 아닌 Text file에 저장하면 된다.

### • DB 없이 웹 서비스 개발 방법

- 웹서버에 id.txt / pass.txt 파일을 만들어서 저장을 해 둔다.
- 게시판 글은 content.txt 파일을 만들어 저장을 해 둔다.
- 로그인 요청시 id.txt / pass.txt 를 읽어 확인 해 준다.





## 버그 없이 고성능으로 구현해야 한다.

- 개발자가 구현해야 할 것
  - 해커의 모든 행동을 고려하여 구현 필요
  - 로그인 시 빠르게 text file을 찾는 알고리즘 구현
  - 고객 데이터가 버그로 인해 망가지는 일이 없어야 함
- 개발자 실수로 인해 해킹을 당하거나,  
고객 데이터가 망가질 확률이 매우 크다.



고객 데이터를 다루는 일은  
위험 물질을 다루는 것과 같다.

이 민감한 데이터들을 안전하게 관리해준다.

- 데이터들을 안전하게 다루도록 해준다.
- 고속으로 데이터들을 Read / Write 할 수 있음
- 세계 고수들의 알고리즘들이 모두 구현되어 있음

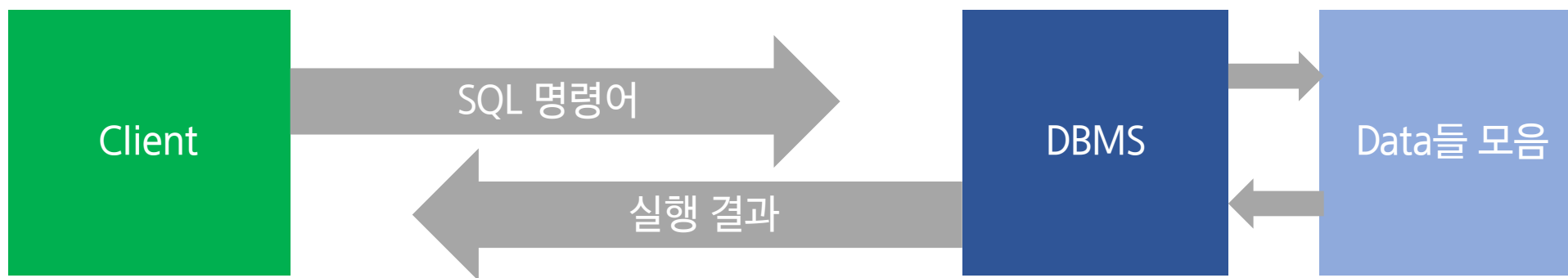
데이터들을 편리하고, 빠르고, 안전하게 사용할 수 있다.

- 따라서, 데이터 관리는 DB가 필수



## 데이터 검색 알고리즘 / 암호화 등을, 직접 구현을 안해도 됨

1. DBMS 에서 데이터 관리 방법을 알고,
2. DBMS를 제어하는 “SQL 명령어” 만 학습하면 데이터 제어를 손쉽게 가능



## Database 트렌드

## DB는 RDBMS (관계형 DBMS) / 비관계형 DBMS으로 나뉜다.

- **관계형 DBMS**

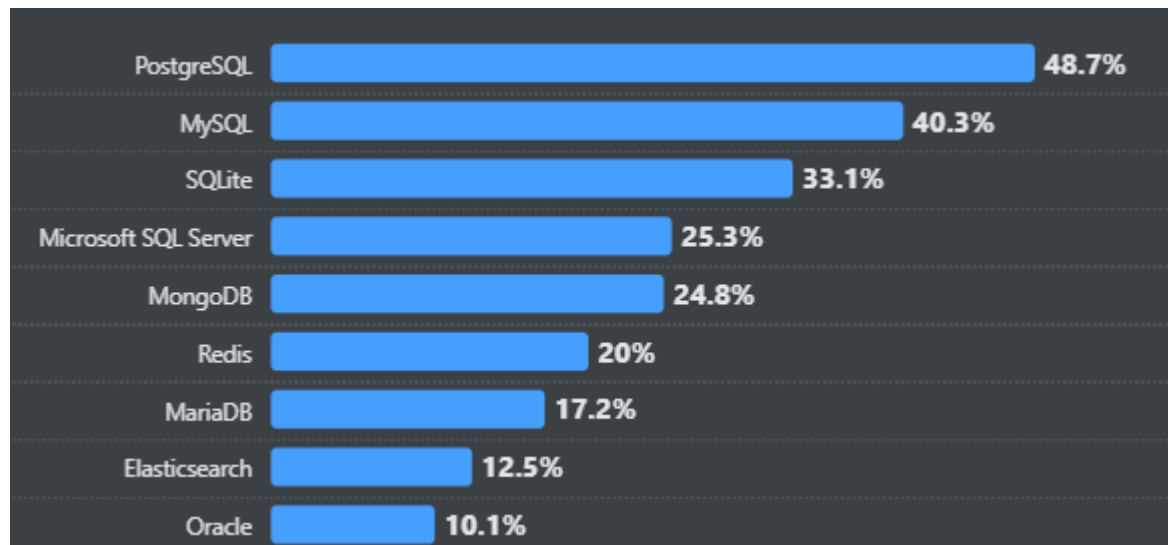
- 구조 및 제약조건 (스키마)를 만들고 값을 채워야 함
- 대표적 RDBMS : MySQL / Oracle

- **비관계형 DBMS**

- RDBMS의 구조로 저장하지 않는 DBMS를 뜻한다.
- 비관계형 DBMS 마다 구조가 다양하다.
- 대표적인 비관계형 DBMS
  - Mongo DB
  - Redis

## Database (DBMS) 순위

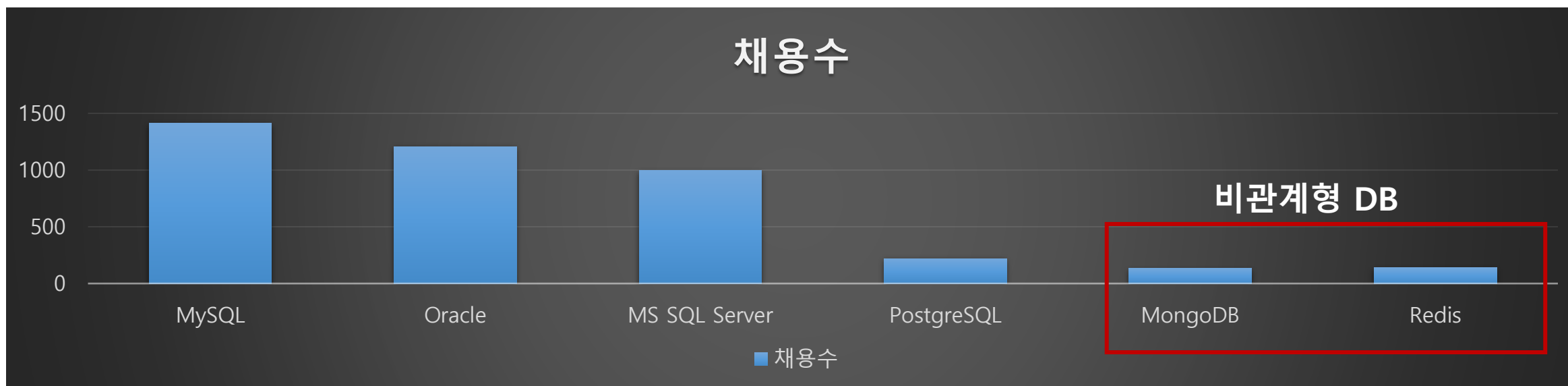
- stackoverflow 2024 Survey
- 전 세계 트렌드 확인 가능 (<https://survey.stackoverflow.co/2024>)



## 국내 채용사이트 기준

- 사람인 2020년, Backend / Server 개발자 기준

- 1위 : MySQL
- 2위 : Oracle (기업에서 유료)
- 3위 : Microsoft SQL Server



## MySQL

- **중요정보**

- Open Source / 기업도 무료
- DBMS에 포함된 도구
  - MySQL Server 를 기본적으로 포함
  - MySQL Client 프로그램
    - GUI 용 : **MySQL Workbench**
    - CLI 용 : mysql

- **덜 중요 정보**

- C / C++ 로 제작 됨
- 제작자 딸 이름 “My” 에서 지어진 이름
- Oracle 이 인수 함



## DB 명령어 (SQL문)을 작성하는 방식의 코딩테스트도 실행 중

- Web 개발 Startup 회사들

- 당근마켓
- 왓챠 등

- 대기업 IT 계열사

- SK 주식회사 (C&C)

문제 설명

GAME\_USERS 테이블은 XX런 게임 유저의 정보를 담고 있는 테이블입니다. GAME\_USERS 테이블 구조는 다음과 같으며, ID, DISTANCE, TIME\_SPENT, BEST\_DATE 는 각각 유저의 아이디, 최고 기록(미터 단위), 게임을 한 시간, 최고기록 경신 날짜를 나타냅니다.

NAME	TYPE
ID	VARCHAR
DISTANCE	INT
TIME_SPENT	DECIMAL
BEST_DATE	DATETIME

CHARACTERS 테이블은 XX런게임에서 살 수 있는 캐릭터의 정보를 담고 있는 테이블입니다. CHARACTERS 테이블 구조는 다음과 같으며, NAME, SPEED, BOOST\_SPEED, BOOST\_TIME, PRICE 는 각각 캐릭터의 이름, 속도, 부스트 속도, 부스트 지속 시간, 가격을 나타냅니다.

NAME	TYPE
NAME	VARCHAR
SPEED	INT
BOOST_SPEED	INT
BOOST_TIME	INT
PRICE	INT

PURCHASES 테이블은 XX런 게임의 유료캐릭터 구매내역을 담고 있는 테이블입니다. PURCHASES 테이블 구조는 다음과 같으며, ID, USER\_ID, PURCHASE\_DATE, ITEM 은 각각 ID, 유저의 ID, 구매 날짜, 산 캐릭터의 이름을 나타냅니다.

solution.sql

```
-- 코드를 입력하세요
SELECT
```

SQL 문 작성

mysql / oracle 선택가능

## MySQL 기본기 확립

1. 임베디드 / IoT 개발시 Database 활용을 위함
2. 웹 개발시 활용을 위함
3. Database 코테 합격을 위함

# MySQL 구조

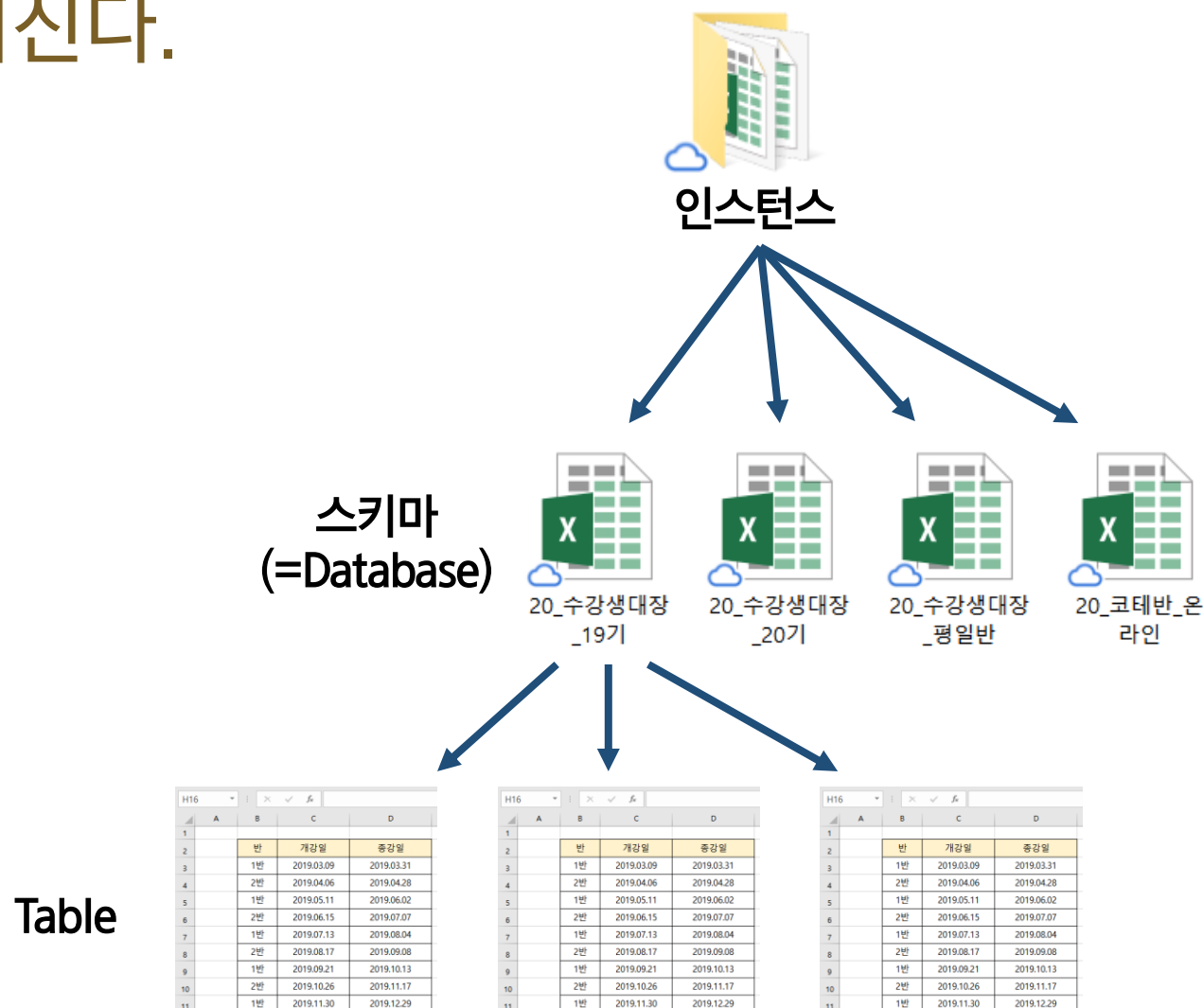
## DB는 3 계층 구조로 이루어진다.

- MySQL 8 기준

- 인스턴스 = DB 서버
- 스키마 = Database
- 테이블

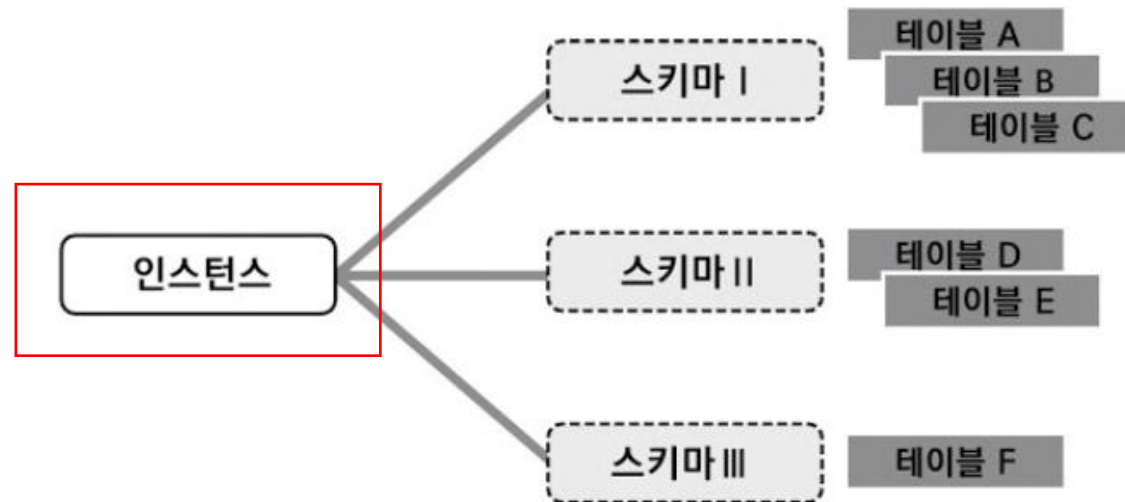
- 유의사항

Oracle에서는  
Database와 Schema 독립된  
4 계층 구조로 되어있어  
MySQL 구조와 혼동되기 쉽다.



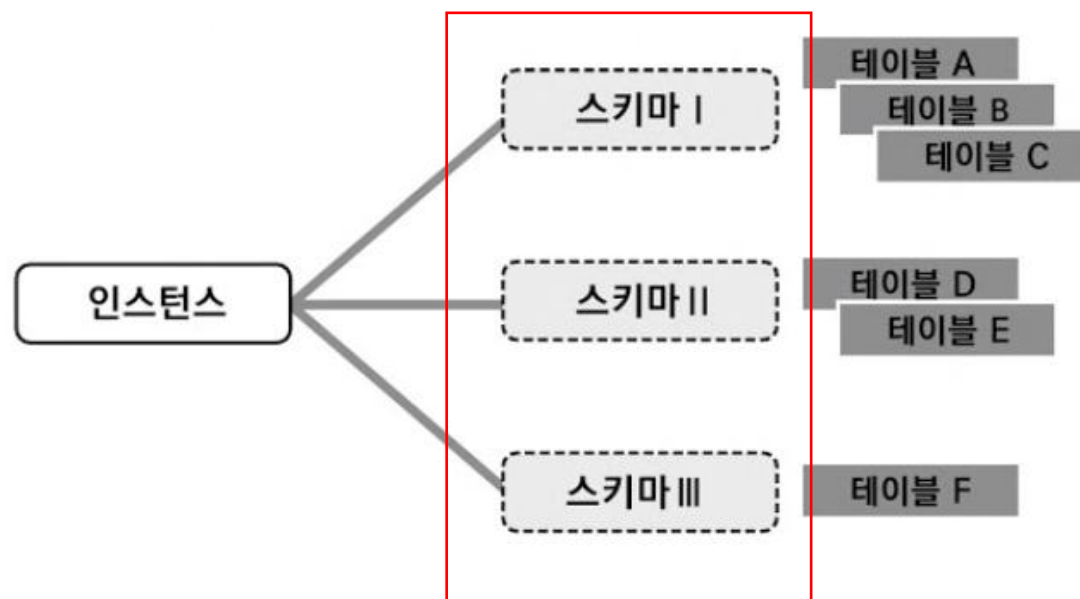
## 서버 인스턴스는 하나의 DB Server 를 나타낸다.

- DBMS를 설치되면 인스턴스가 자동 생성된다.
- 하나의 DB를 운영하기 위해 내부 Buffer / 내부 저장공간 / 관리 도구들이 동작되어야 한다.  
운영이 필요한 모든 도구들을 모아 “서버 인스턴스” 라고 부른다.



## 스키마는 Database와 동일한 뜻이다.

- 스키마 생성하는 SQL 명령어 1 : `CREATE DATABASE ssafy;`
- 스키마 생성하는 SQL 명령어 2 : `CREATE SCHEMA ssafy;`



## 데이터 타입 종류

## 숫자형

- int / float

## 문자형

- char / varchar / text

## 날짜형

- date / datetime



## char

- 고정길이 데이터 타입, 빈 공간은 띄어쓰기로 채운다.
- 예를들어 char(5) 라면, 5 Byte 를 사용하는 것이고  
만약 해당 문자열에 'BTS' 를 넣으면 공백을 포함한 'BTS ' 으로 저장 된다.

## varchar

- 가변길이 데이터 타입, 빈 공간을 채우지 않는다.
- varchar(20) 이지만, 'BTS'를 넣는다면 3 Byte만 사용한다.
- 공간을 아낄 수 있지만, 데이터 파편화로 성능 저하

## 문자열의 특징에 따라 쓰는 타입을 결정하자

- char : 길이가 일정한 문자 / 빠른 성능이 필요한, 자주 읽는 필드
- varchar : 주소 와 같이 길이 변화가 큰 문자
- text : 64KB 미만의 긴 문자열

## 대표적인 날짜형 데이터 타입

- **DATE**

- 날짜 타입 (year, month, day)
- 기본 Format은 YYYY-MM-DD

- **DATETIME**

- 날짜 + 시간을 나타냄
- YYYY-MM-DD HH:MM:SS

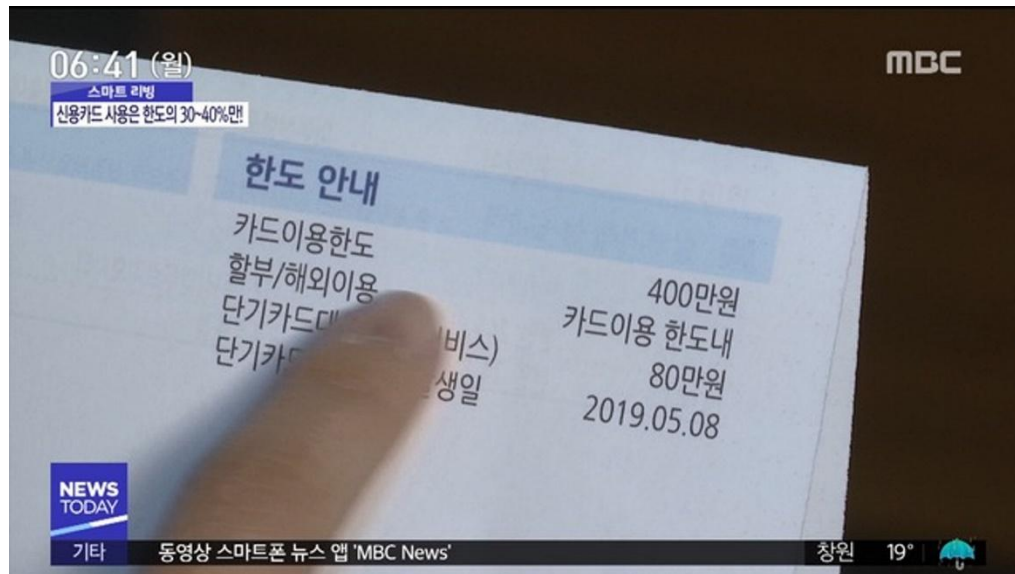
## JSON Data를 저장할 수 있는 Type

- INSERT 예시

```
mysql> INSERT INTO t1 VALUES('{"key1": "value1", "key2": "value2"}');  
Query OK, 1 row affected (0.01 sec)
```

보안을 위해서, 모든 변경 권한이 있는 root 아이디를 쓰지 말자.

- 해킹 당했을 때, 피해를 최소화 하기 위함
  - 사용자 계정 해킹 당하는 경우 : 권한 허용한도 내 해킹 피해를 입는다.
  - root 계정 해킹 당하는 경우 : 해킹 피해 MAX



무제한 한도 카드를 도난 당했을 때 피해  
VS  
카드이용한도 카드를 도난 당했을 때 피해

### root 계정 사용 하는 경우

- **평소에 사용하지 않는다.** 사용자 계정 권한이 없는 일을 할 때만 접속하여 사용한다.
- 사용자 계정 추가 / 스키마 (Database) 생성 등

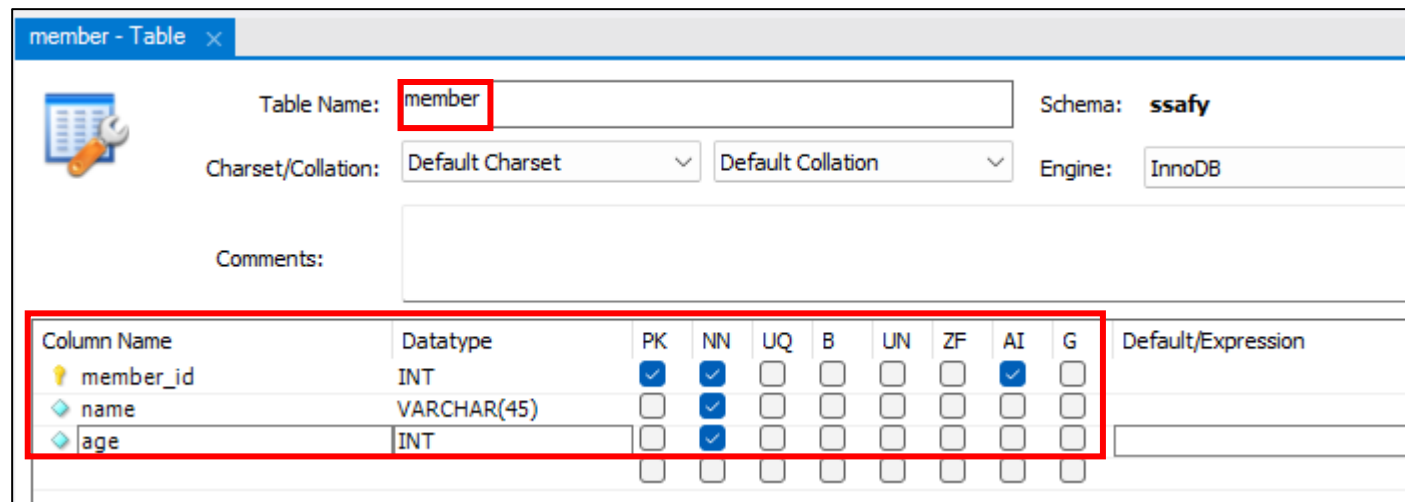
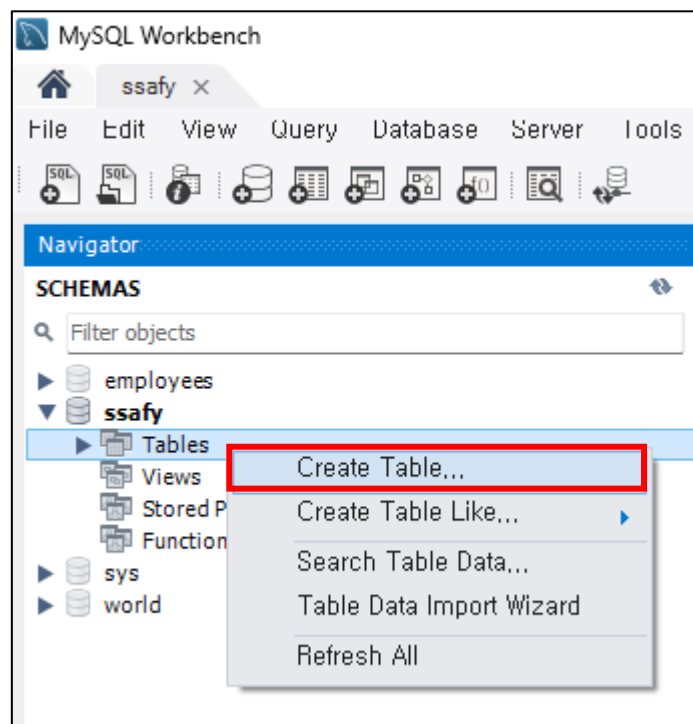
### 사용자 계정 사용 하는 경우

- 허용된 스키마 (Database)만 사용 할 권한이 부여된다.
- 허용된 스키마에 **Table 추가 후 SQL 명령어 사용 가능**
- 스키마 편집 / 제거 불가

## 테이블 생성

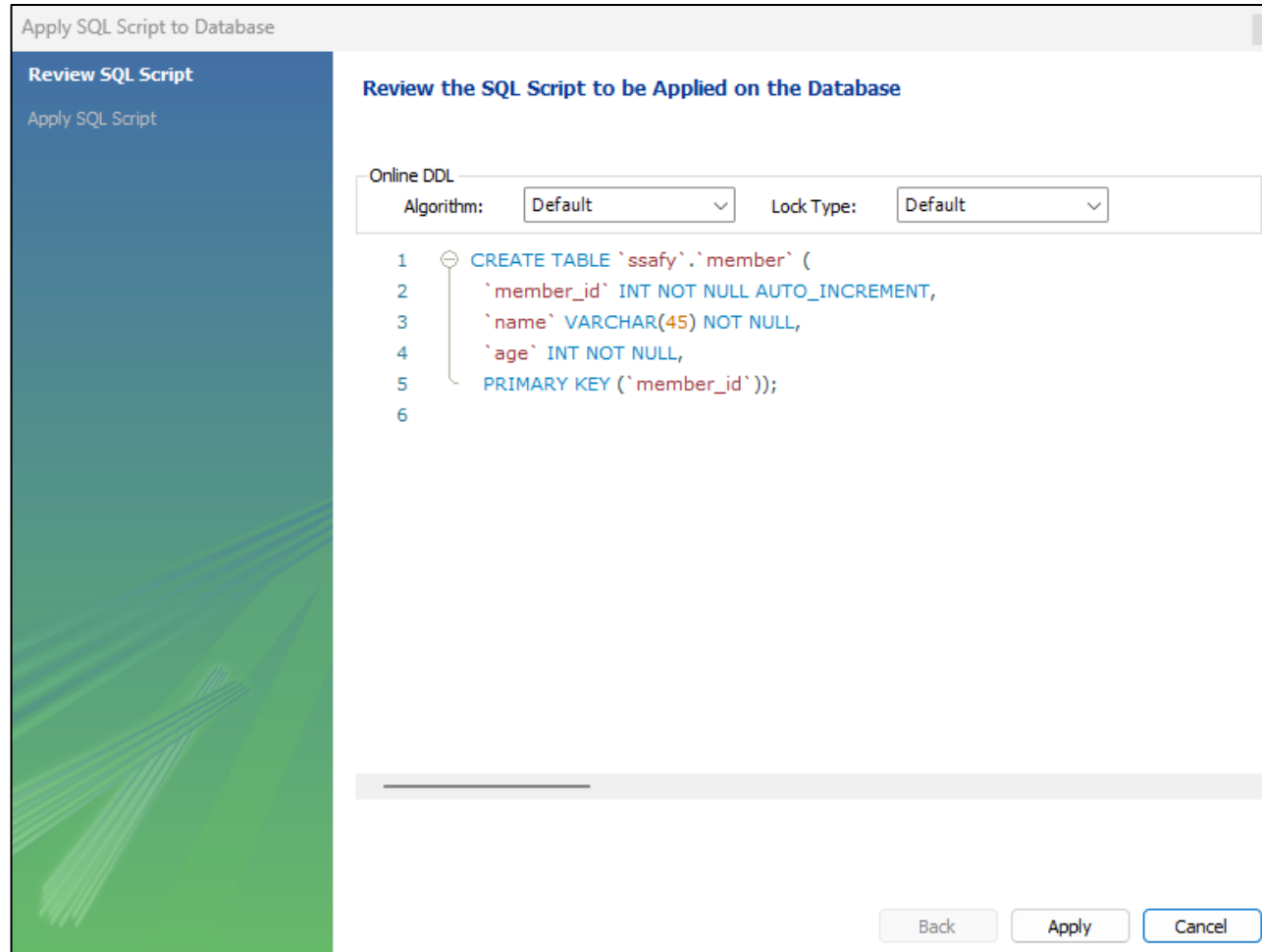
## member 테이블을 생성하자

- PK : Primary Key (주키)
- NN : Not Null
- AI : Auto Increment



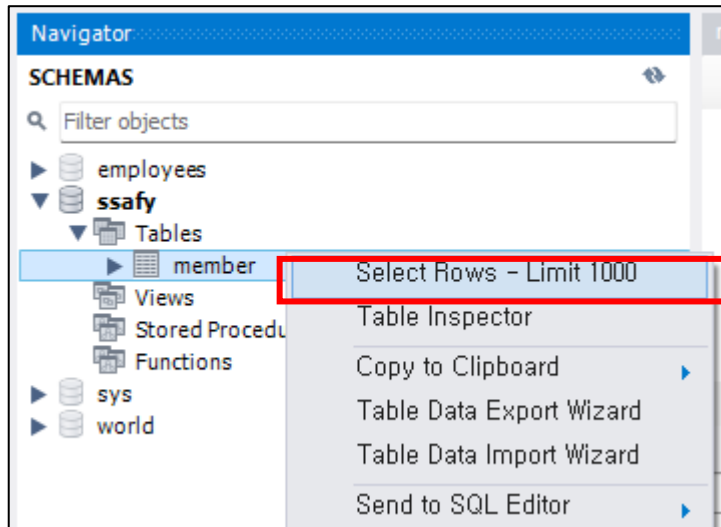


## CREATE 쿼리문 확인

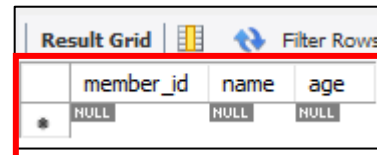


## Navigator 에서 생성된 member 테이블을 확인 해보자

- Select Rows 클릭
- 생성된 Table 확인 가능



자동으로 작성된 쿼리문 확인



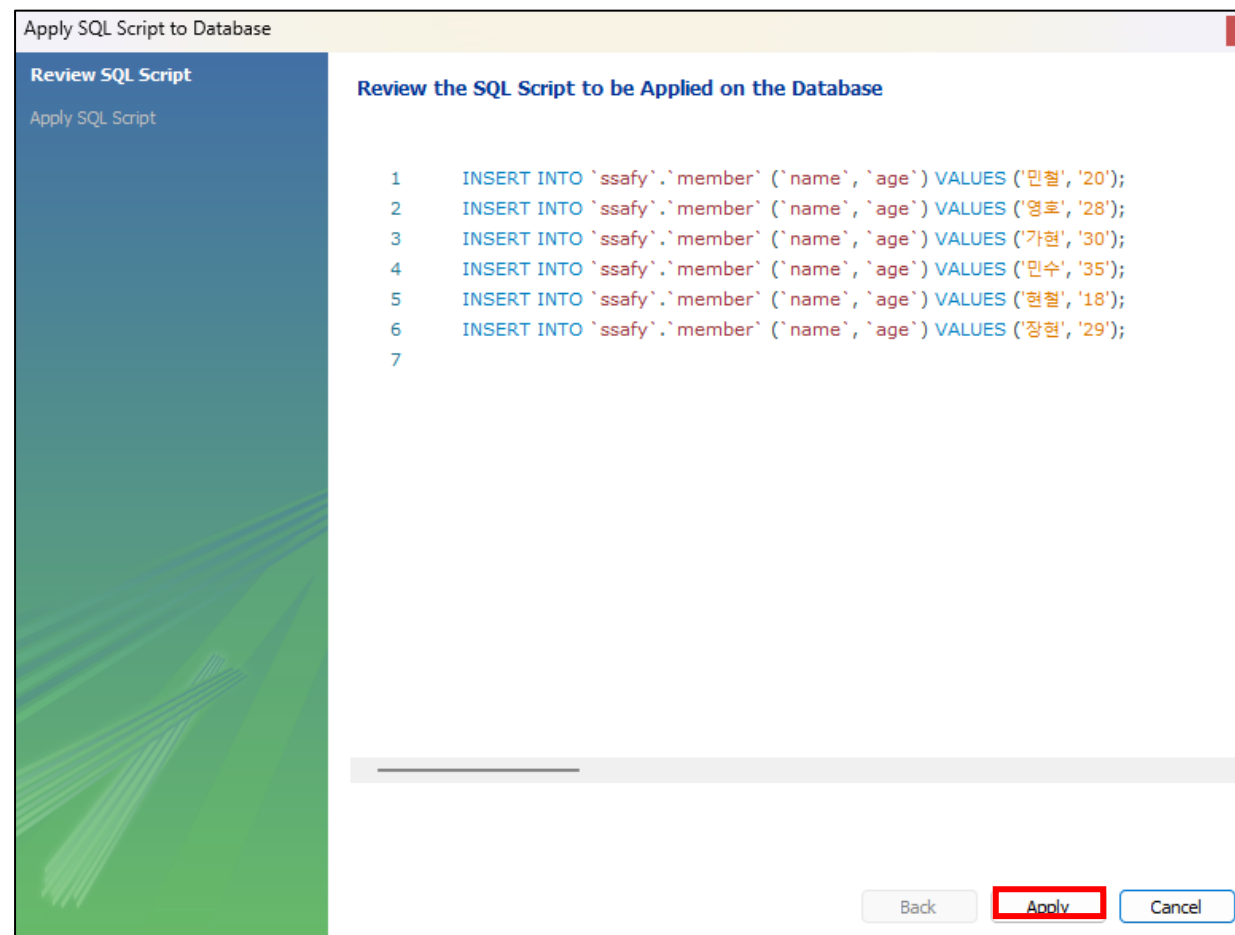
아직 값을 넣지않은 상태

## 샘플 데이터를 입력 후 Apply 클릭

- member\_id 는 Auto Increment 이므로 입력 안함
- 자동으로 생성된 INSERT INTO 쿼리문 확인

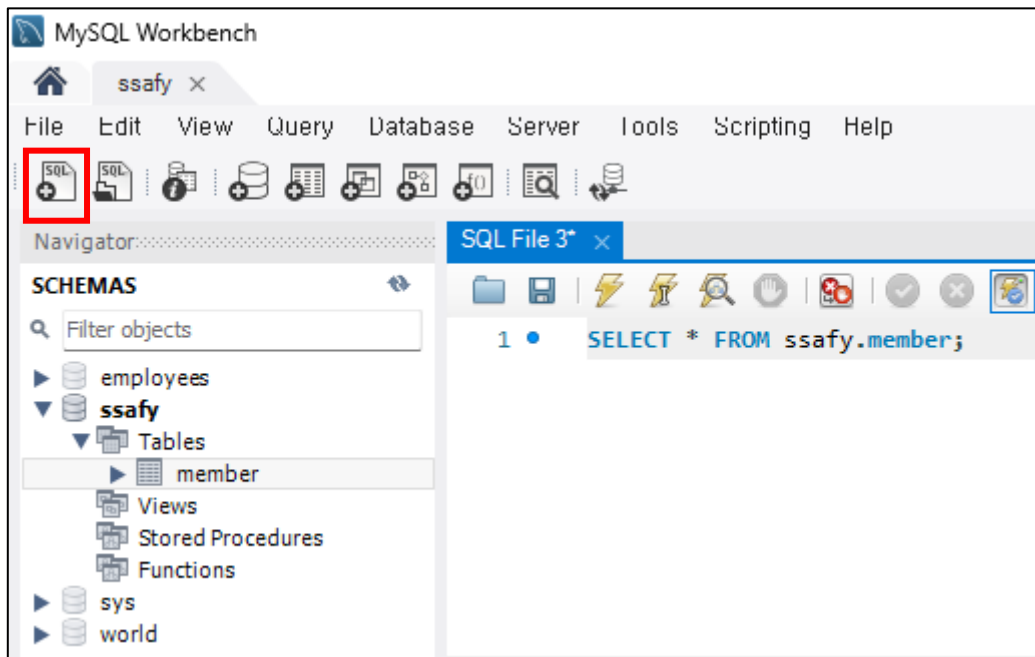
Result Grid			
member_id	name	age	
NULL	민철	20	
NULL	영호	28	
NULL	가현	30	
NULL	민수	35	
NULL	현철	18	
NULL	장현	29	
NULL	NULL		

member 1 x **Apply** Revert



## SQL File을 하나 생성하기

- `select * from ssafy.member` 입력
- execute 단축키 : **Ctrl + Enter**



Result Grid			
Filter Rows:			
	member_id	name	age
▶	1	민철	20
	2	영호	28
	3	가현	30
	4	민수	35
	5	현철	18
	6	장현	29
*	NULL	NULL	NULL

# Day1-2. SELECT 기초

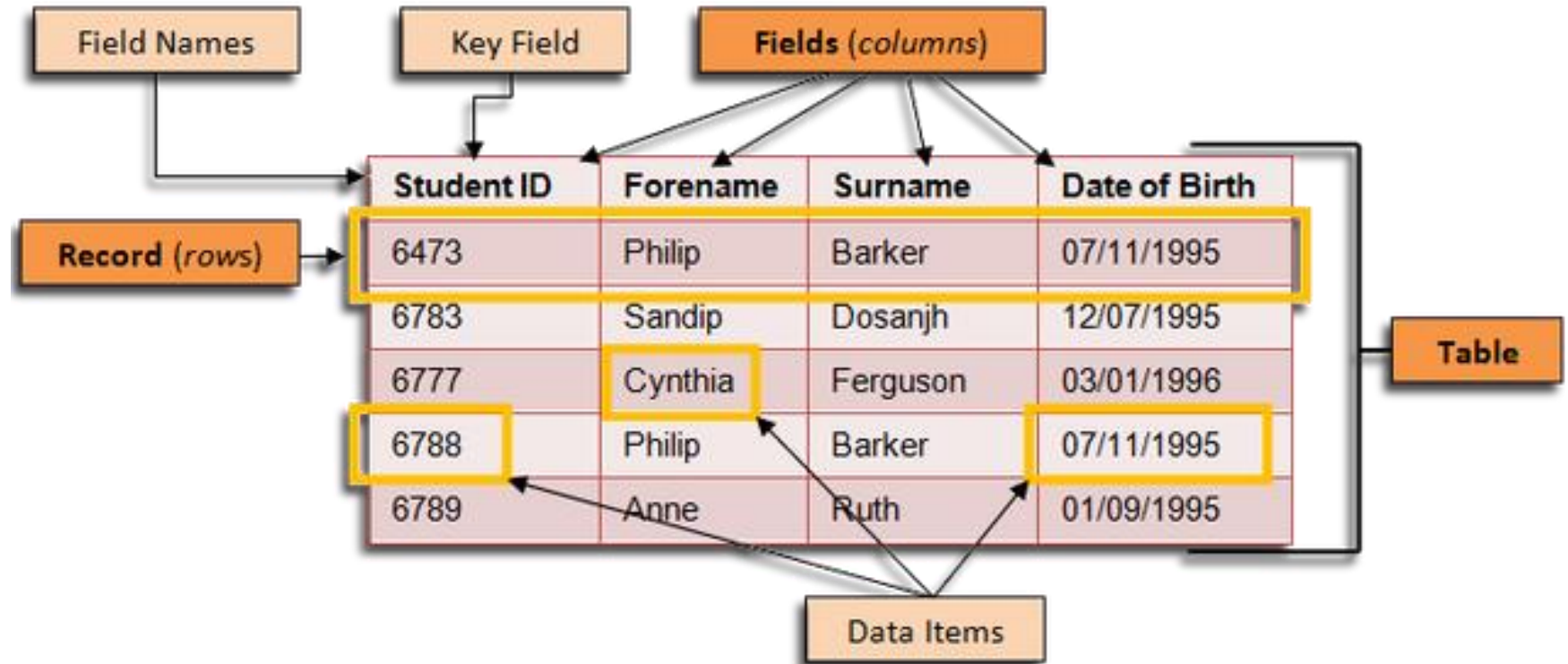
# 챕터의 포인트

- SELECT 기초
- DB Backup

# SELECT 기초

## 세 가지 용어들을 암기하자

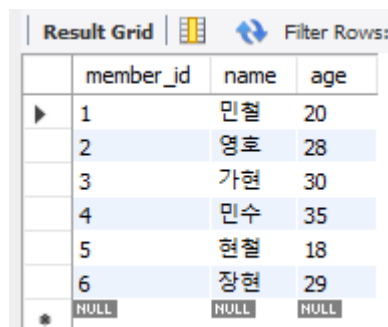
- Field (columns)
- Record (rows)
- Items





## SELECT 필드명 FROM 테이블

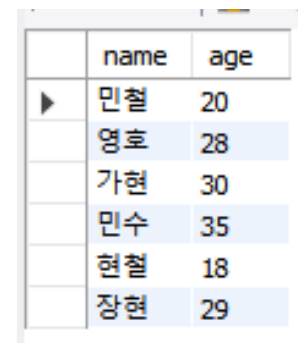
- 모든 필드를 출력하고 싶을 때는 \* 을 쓰면 된다.  
SELECT \* FROM TABLE



Result Grid

	member_id	name	age
▶	1	민철	20
	2	영호	28
	3	가현	30
	4	민수	35
	5	현철	18
	6	장현	29
*	NULL	NULL	NULL

SELECT \* FROM ssafy.member;

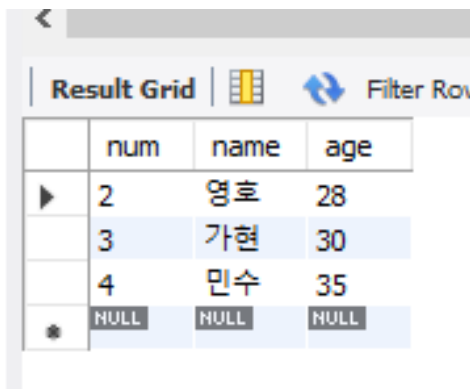


	name	age
▶	민철	20
	영호	28
	가현	30
	민수	35
	현철	18
	장현	29

SELECT name, age from ssafy.member;

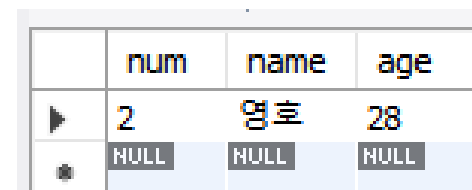
## WHERE 절 사용하기

- **SELECT \* FROM TABLE WHERE 조건1 AND 조건2 AND 조건3 ...**
- 특정 조건에 해당하는 레코드만 출력할 때 사용
  - 특정 필드만 출력 : **SELECT [어떤 필드?] FROM TABLE**
  - 특정 레코드만 출력 : **SELECT \* FROM TABLE WHERE [어떤 레코드?]**



	num	name	age
▶	2	영호	28
	3	가현	30
	4	민수	35
•	NULL	NULL	NULL

```
SELECT * FROM ssafy.member  
WHERE member_id >= 2 and member_id <= 4;
```



	num	name	age
▶	2	영호	28
•	NULL	NULL	NULL

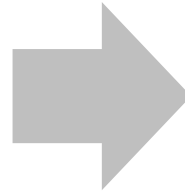
```
SELECT * FROM ssafy.member  
WHERE name='영호';
```

## alias 지정하여 필드 이름 변경

- SELECT 필드명 AS [별칭] FROM TABLE

Result Grid	
	name
▶	민철
	영호
	가현
	민수
	현철
	장현

SELECT name FROM ssafy.member;

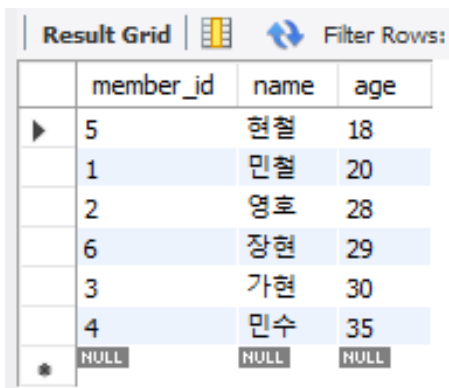


	이름
▶	민철
	영호
	가현
	민수
	현철
	장현

SELECT name as '이름' FROM ssafy.member;

## 특정 필드를 정렬한다

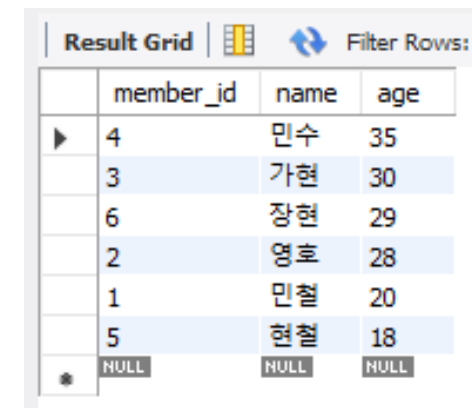
- 오름차순 : `SELECT * FROM TABLE ORDER BY [필드명]`
- 내림차순 : `SELECT * FROM TABLE ORDER BY [필드명] DESC;`



Result Grid | Filter Rows:

	member_id	name	age
▶	5	현철	18
	1	민철	20
	2	영호	28
	6	장현	29
	3	가현	30
	4	민수	35
*	NULL	NULL	NULL

`SELECT * FROM ssafy.member ORDER BY age;`



Result Grid | Filter Rows:

	member_id	name	age
▶	4	민수	35
	3	가현	30
	6	장현	29
	2	영호	28
	1	민철	20
	5	현철	18
*	NULL	NULL	NULL

`SELECT * FROM ssafy.member ORDER BY age DESC;`

원하는 데이터 개수만큼만 가져온다.

- LIMIT 가져올 개수 / LIMIT 건너뛸 개수, 가져올 개수

```
SELECT * FROM ssafy.member LIMIT 3;  
SELECT * FROM ssafy.member LIMIT 3, 3;
```

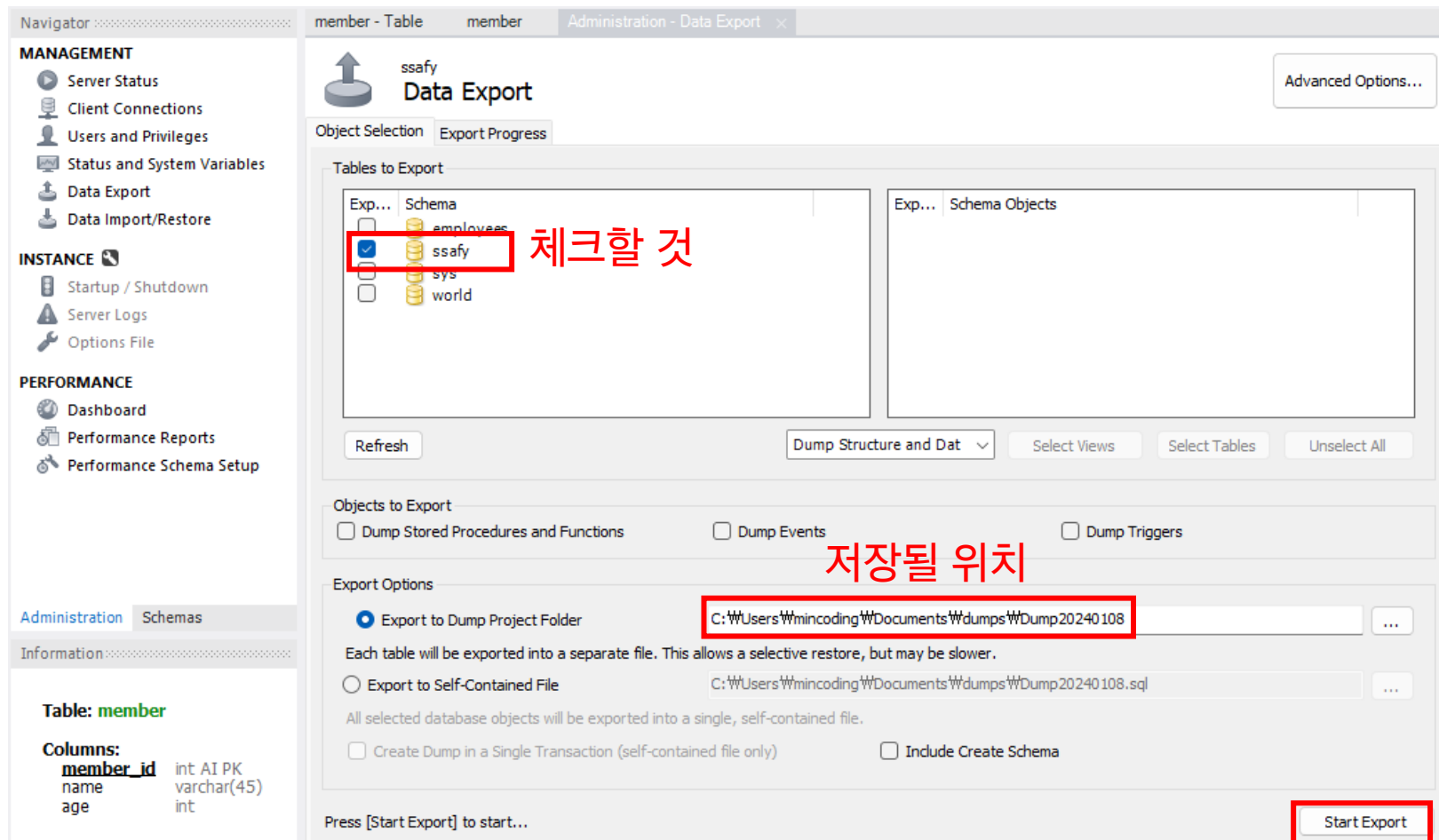
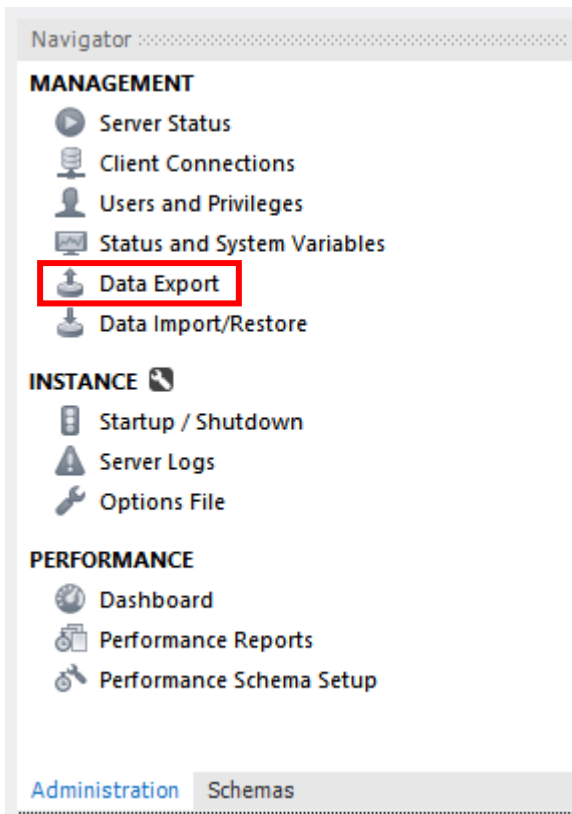
	member_id	name	age
▶	1	민철	20
	2	영호	28
	3	가현	30
✱	NULL	NULL	NULL

	member_id	name	age
▶	4	민수	35
	5	현철	18
	6	장현	29
✱	NULL	NULL	NULL

# DB Backup

## Data Export

- 현재 Data를 외부 파일로 추출한다.



## 데이터 변경

- 데이터를 변경한 후, Restore 해보자
  - Member Table에서 레코드 추가 및 제거
  - 새로운 Table 추가하기

Result Grid

Filter Rows:

Edit:

	member_id	name	age
	1	민철	20
	2	영호	28
	3		
	4		
	5		
	6		
	NULL		

Open Value in Editor

Set Field to NULL

Mark Field Value as a Function/Literal

Delete Row(s)

TEST - Table

Table Name: TEST Schema: ssafy

Charset/Collation: Default Charset Default Collation Engine: InnoDB

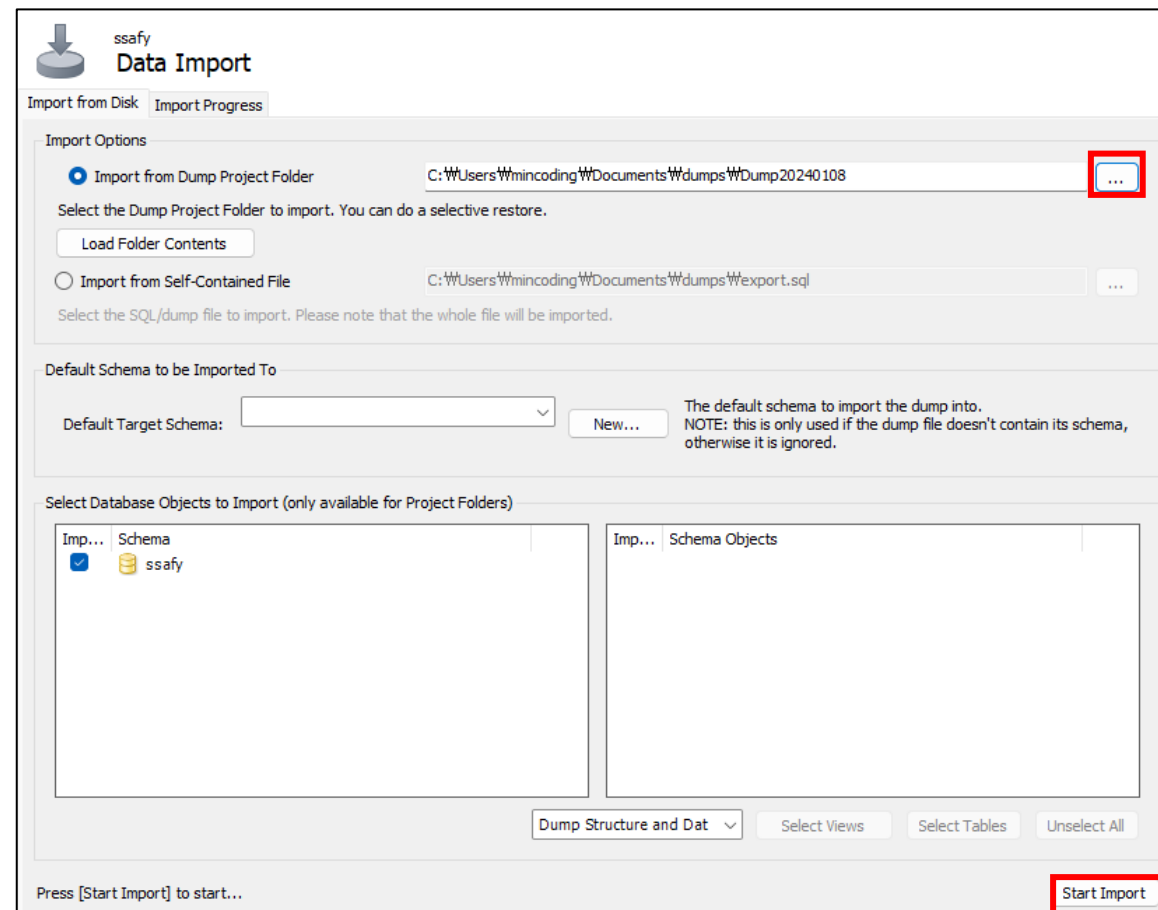
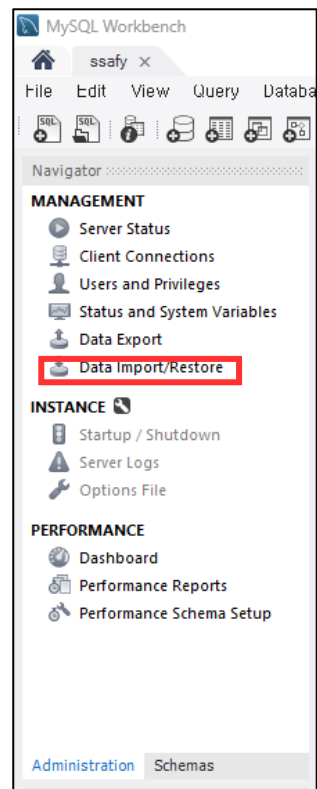
Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
idTEST	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
TESTcol	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
TESTcol1	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	



## 데이터 복구 하기

- 제거된 데이터는 복구가 된다.
- 복구 전 추가된 Table은 삭제하지 않는다.



# Day1-3. CRUD와 Console

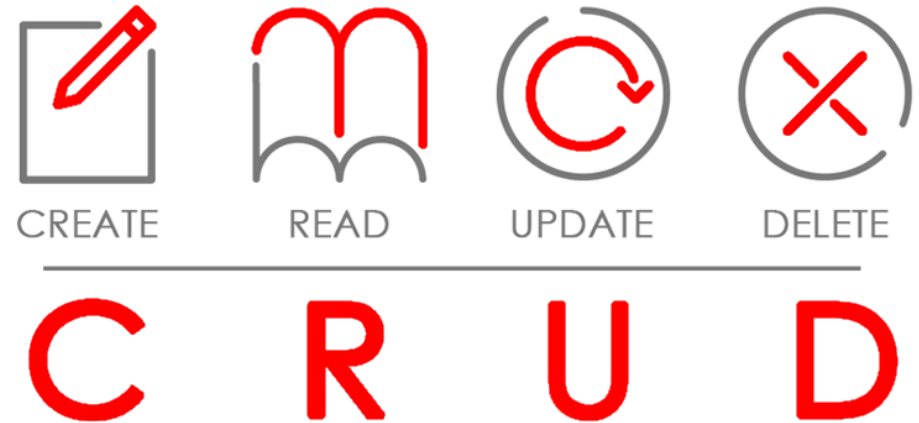
# 챕터의 포인트

- CRUD
- MySQL Console

**CRUD**

# CRUD : Create + Read + Update + Delete

- 데이터를 다루는 Software의 기본적인 인터페이스를 뜻함
- Database에서 CRUD (암기 추천)
  - C : INSERT INTO ~ VALUES
  - R : SELECT
  - U : UPDATE ~ SET WHERE
  - D : DELETE FROM ~ WHERE



웹서비스에서 해당 Query를 쓰는 이유에 대해 생각해보자.

- 로그인 : SELECT
- 회원가입 : SELECT, INSERT, UPDATE
- 회원탈퇴 : DELETE
- 게시판은 CRUD 전체 사용

## INSERT INTO

- INSERT INTO 테이블명 (‘컬럼명’, ‘컬럼명’, ‘컬럼명’) VALUES (값1, 값2, 값3)
- 값 ('박중박', 50세)를 추가하자.
- 값 ('이순신', 20세)을 추가하자.
- PK 는 기입하지 않음

```
INSERT INTO ssafy.member (‘name’, ‘age’) value('박중박', 50);
```

```
INSERT INTO ssafy.member (‘name’, ‘age’) value('이순신', 20);
```

	member_id	name	age
▶	1	민철	20
	2	영호	28
	3	가현	30
	4	민수	35
	5	현철	18
	6	장현	29



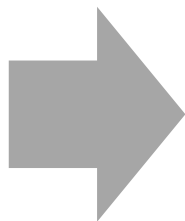
	member_id	name	age
▶	1	민철	20
	2	영호	28
	3	가현	30
	4	민수	35
	5	현철	18
	6	장현	29
	7	박중박	50
	8	이순신	20

## UPDATE 테이블 SET 필드=값 WHERE 레코드 조건

- 수정, 삭제 시 WHERE 은 필수 기입!
- '이순신' 나이를 320세로 수정하자.
- '박종박'의 이름을 '박재훈' 으로 수정하자.

```
UPDATE ssafy.member SET age=320 WHERE member_id=8;  
UPDATE ssafy.member SET name='박재훈' WHERE member_id=7;  
SELECT * FROM ssafy.member;
```

	member_id	name	age
▶	1	민철	20
	2	영호	28
	3	가현	30
	4	민수	35
	5	현철	18
	6	장현	29
	7	박종박	50
	8	이순신	20
*	NULL	NULL	NULL



	member_id	name	age
▶	1	민철	20
	2	영호	28
	3	가현	30
	4	민수	35
	5	현철	18
	6	장현	29
	7	박재훈	50
	8	이순신	320
*	NULL	NULL	NULL

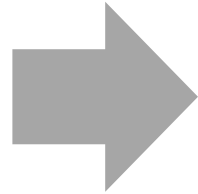


## DELETE FROM 테이블 WHERE 레코드 조건

- 7번 레코드 삭제
- 8번 레코드 삭제

```
DELETE FROM ssafy.member WHERE member_id=7;  
DELETE FROM ssafy.member WHERE member_id=8;  
SELECT * FROM ssafy.member;
```

	member_id	name	age
▶	1	민철	20
	2	영호	28
	3	가현	30
	4	민수	35
	5	현철	18
	6	장현	29
	7	박재훈	50
	8	이순신	320
*	NULL	NULL	NULL



	member_id	name	age
▶	1	민철	20
	2	영호	28
	3	가현	30
	4	민수	35
	5	현철	18
	6	장현	29
*	NULL	NULL	NULL

## CRUD 명령어 Format을 암기 해두자

- INSERT INTO 테이블 (`컬럼명`, ...) VALUES(값, ...)
- UPDATE 테이블 SET 필드 WHERE
- DELETE FROM 테이블 WHERE

# MySQL Console

## GUI 환경 : MySQL Workbench

- 대량 데이터에도, 구조를 쉽게 파악할 수 있다.
- 사용하기 편리한 장점

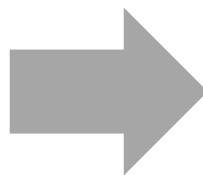
## CLI 환경 : MySQL Console

- Linux + DB Server + SSH 에서 DB 초기설정에 사용된다.
- Workbench 없이 간단한 데이터 조회 시에도 사용

## SHOW DATABASES / USE [DB명]

- 현 계정이 접근 권한이 있는 Database 목록을 확인한다.
- 사용할 Databases 선택 명령어 : use [DB이름]

```
mysql> SHOW DATABASES;  
+-----+  
| Database  
+-----+  
| employees  
| information_schema  
| mysql  
| performance_schema  
| ssafy  
| sys  
| world  
+-----+  
7 rows in set (0.00 sec)
```



```
mysql> USE ssafy;  
Reading table info  
You can turn off  
Database changed
```

## SHOW TABLES;

- 테이블 목록 확인

## DESC [테이블명]

- DESCRIBE [테이블명]
- 테이블 구조 확인

```
mysql> SHOW TABLES;
```

Tables_in_ssafy
TEST
member

```
2 rows in set (0.00 sec)
```

```
mysql> DESC member;
```

Field	Type	Null	Key	Default	Extra
member_id	int	NO	PRI	NULL	auto_increment
name	varchar(45)	NO		NULL	
age	int	NO		NULL	

```
3 rows in set (0.00 sec)
```

# Day1-4. Select - 2

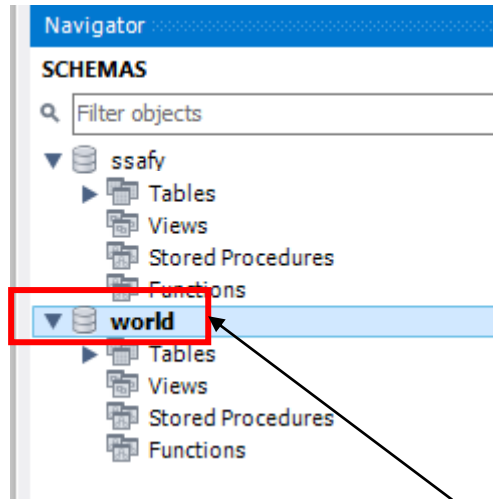
# 챕터의 포인트

- 샘플 스키마 준비
- DB KEY
- JOIN
- LIMIT / BETWEEN / IN / LIKE
- GROUP BY / HAVING

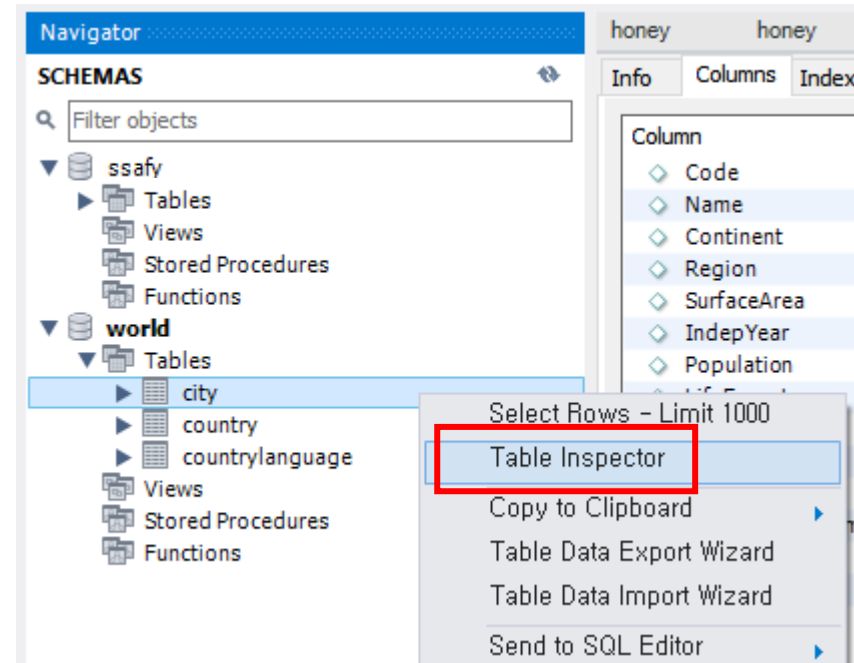


## 샘플 스키마 준비

## 연습용 스키마 world 구조 확인하기



더블 클릭 시  
자동으로 use world;  
명령어가 수행된다.



Columns 구조를 확인할 수 있다.

world.city x

InfoColumnsIndexesTriggersForeign keysPartitionsGrantsDDL

Column	Type	Default Value	Nullable	Character Set	Collation	Privileges
ID	int		NO			select,insert,update,references
Name	char(35)		NO	latin1	latin1_swedish_ci	select,insert,update,references
CountryCode	char(3)		NO	latin1	latin1_swedish_ci	select,insert,update,references
District	char(20)		NO	latin1	latin1_swedish_ci	select,insert,update,references
Population	int	0	NO			select,insert,update,references

```
mysql> DESC city;
```

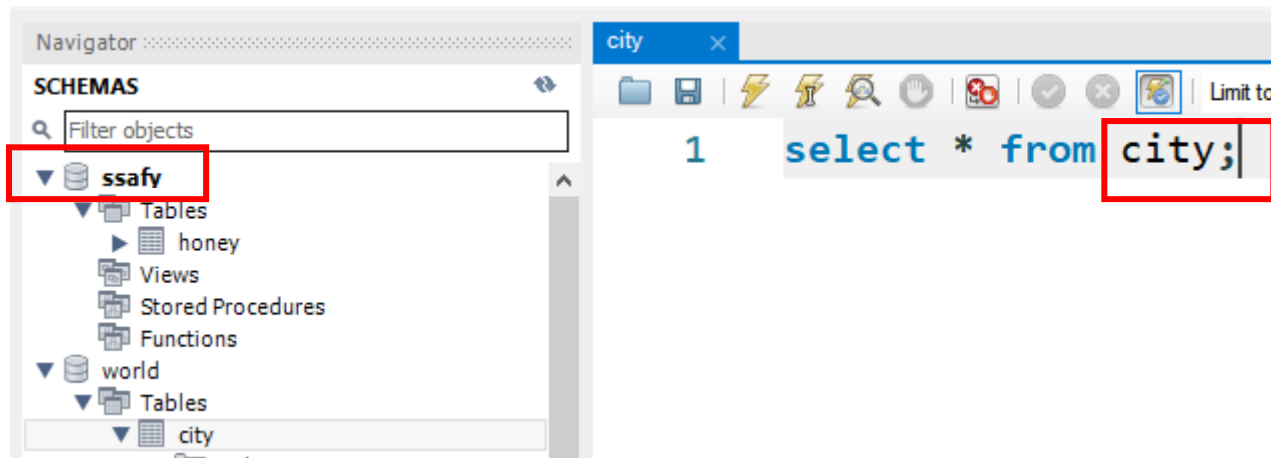
```
mysql> DESC city;
```

Field	Type	Null	Key	Default	Extra
ID	int	NO	PRI	NULL	auto_increment
Name	char(35)	NO			
CountryCode	char(3)	NO	MUL		
District	char(20)	NO			
Population	int	NO		0	

```
5 rows in set (0.00 sec)
```

world 스키마의 city table을 query 하기 위해서 USE 필수

- 아래 그림과 같이 world가 굵은 글씨가 아닌 상태에서, 에러가 발생한다.
- “world” 에 더블클릭 해주면 된다.



**DB KEY**

## PK(Primary Key)

- 한 테이블 내에서 중복되지 않는 값만 가질수 있는 키
- 유일한 값
- 중복 허용 X
- NULL 값 X

menus			
id	name	description	image_src
1	카페라떼	라떼는.. 말이야	public/image1.jpg
2	카푸치노	언빌리 "버블"	public/image2.jpg
3	아메리카노	아메아메 좋아	public/image3.jpg
4	복숭아 아이스	상큼한 복숭아!	public/image4.jpg

## FK (Foreign Key)

- 현재 테이블의 컬럼이면서, 다른 테이블의 기본 키(또는 유니크 키)를 참조하는 키이다.
- 기본적으로는 다른 테이블의 기본 키를 참조한다.
- NULL, 중복 허용 OK

menus			
id	name	description	image_src
1	카페라떼	라떼는.. 말이야	public/image1.jpg
2	카푸치노	언빌리 "버블"	public/image2.jpg
3	아메리카노	아메아메 좋아	public/image3.jpg
4	복숭아 아이스	상큼한 복숭아!	public/image4.jpg

orders			
id	quantity	request_detail	menus_id
1	1	얼음 많이많이 주세요.	1
2	3	시럽 넣지 말아주세요.	2
3	1	우유 많이 주세요.	2
4	2	복숭아 좋아요.	4

**JOIN**



## 관계형 데이터 베이스의 꽃

- 데이터베이스를 연결하여 데이터를 검색하는 방법
- PK, FK로 두 테이블을 연결한다.

## 왜 테이블을 둘로 나뉘었을까?

- 다음 상황을 가정해보자.
- 만약 100,000 개의 데이터를 저장한다면, 중복이 지나치게 많아 매우 비효율적
  - DB 용량 증가, 속도 저하
  - 비용 증가: Oracle 등의 유료 데이터베이스를 사용한다면, 훨씬 많은 비용 지불
  - 만약, menu\_description 을 바꾸고 싶다면? 100,000 개의 데이터에서 찾아 전부 바꿔야한다

id	name	description	quantity	request_detail
1	카페라떼	라떼는.. 말이야	1	얼음 많이많이 주세요.
2	카페라떼	라떼는.. 말이야	3	시럽 넣지 말아주세요.
3	아메리카노	아메아메 좋아	1	
4	복숭아 아이스티	상큼한 복숭아!	2	

## 관계형 데이터베이스 (Relational Database)

- 여러 테이블의 "관계" 로 이루어진다.
- 핵심은 FK ( Foreign Key, 외래키 ) 와 JOIN
- 중복 제거
  - 만약 100,000 개의 데이터를 저장하더라도,
    - DB 용량을 훨씬 효율적으로 사용
    - 만약, menu\_description 을 바꾸고 싶다면? menus 테이블에서 "단 하나만" 변경하면 끝

menus			
id	name	description	image_src
1	카페라떼	라떼는.. 말이야	public/image1.jpg
2	카푸치노	언빌리 "버블"	public/image2.jpg
3	아메리카노	아메아메 좋아	public/image3.jpg
4	복숭아 아이스	상큼한 복숭아!	public/image4.jpg

orders			
id	quantity	request_detail	menus_id
1	1	얼음 많이많이 주세요.	1
2	3	시럽 넣지 말아주세요.	2
3	1	우유 많이 주세요.	2
4	2	복숭아 좋아요.	4

## JOIN 문법을 사용하는 이유

- orders 에서, 주문마다 menus\_id 보유
- 즉, id 만 알고 있다면 menus 에 접근해서
  - 메뉴 이름, 메뉴 설명, 메뉴 이미지에 대한 정보를 가져올 수 있음
  - 이를 위해 JOIN 문법 사용

menus			
id	name	description	image_src
1	카페라떼	라떼는.. 말이야	public/image1.jpg
2	카푸치노	언빌리 "버블"	public/image2.jpg
3	아메리카노	아메아메 좋아	public/image3.jpg
4	복숭아 아이스	상큼한 복숭아!	public/image4.jpg

orders			
id	quantity	request_detail	menus_id
1	1	얼음 많이많이 주세요.	1
2	3	시럽 넣지 말아주세요.	2
3	1	우유 많이 주세요.	2
4	2	복숭아 좋아요.	4



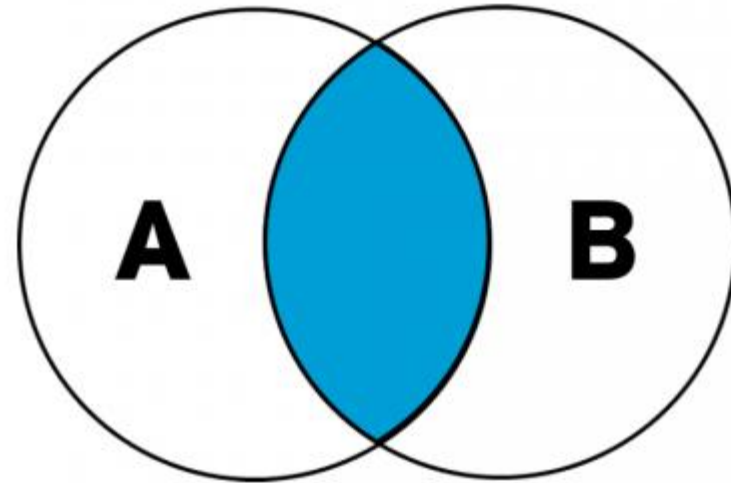
## JOIN

- SELECT <열 목록>  
FROM <첫번째 테이블>  
    <조인의 종류> <두번째 테이블>  
ON <조인될 조건>  
[ WHERE 검색조건 ]

## INNER JOIN

- `SELECT *`  
`FROM A a`  
`INNER JOIN B b`  
`ON a.id = b.id`

## INNER JOIN



## 각 사원들의 사원번호, first\_name, 현재 받고있는 급여액(salary) 가져오기

```
SELECT employees.emp_no, first_name, salary  
FROM employees  
INNER JOIN salaries  
ON employees.emp_no = salaries.emp_no;
```

	emp_no	first_name	salary
▶	10001	Georgi	60117
	10001	Georgi	62102
	10001	Georgi	66074
	10001	Georgi	66596
	10001	Georgi	66961
	10001	Georgi	71046
	10001	Georgi	74333
	10001	Georgi	75286
	10001	Georgi	75994
	10001	Georgi	76884
	10001	Georgi	80013
	10001	Georgi	81025
	10001	Georgi	81097
	10001	Georgi	84917

**LIMIT/ BETWEEN / IN / LIKE**



## 특정 개수만 보여줌

- 빠른 Query 성능을 위해 사용 함
- 웹 개발에서, 최근 게시물에 자주 사용함

```
SELECT * FROM city LIMIT 3;
```

	ID	Name	CountryCode	District	Population
▶	1	Kabul	AFG	Kabul	1780000
	2	Qandahar	AFG	Qandahar	237500
	3	Herat	AFG	Herat	186800
*	NULL	NULL	NULL	NULL	NULL

## 구간 사이 데이터 추출

- WHERE 절 다음 입력한다.
- `population >= 500 AND population <= 900` 과 동일하다.

```
SELECT * FROM city  
WHERE population BETWEEN 500 AND 900;
```

	ID	Name	CountryCode	District	Population
▶	62	The Valley	AIA	—	595
	1791	Flying Fish Cove	CXR	—	700
	2316	Bantam	CCK	Home Island	503
	2728	Yaren	NRU	—	559
	2805	Alofi	NIU	—	682
	2806	Kingston	NFK	—	800
*	NULL	NULL	NULL	NULL	NULL

## 후보들 중 해당하는 row 찾기

- WHERE 절 다음 입력한다.
- 필드명 IN ( 후보 Item1, 후보 Item2, 후보 Item3 .. )

```
SELECT * FROM city  
WHERE name IN('Seoul', 'Sydney', 'Oxford');
```

	ID	Name	CountryCode	District	Population
▶	130	Sydney	AUS	New South Wales	3276207
	498	Oxford	GBR	England	144000
	2331	Seoul	KOR	Seoul	9981619
*	NULL	NULL	NULL	NULL	NULL

## 문자열 검색

- WHERE 절 다음에 사용
- % : 다중 문자를 의미한다. (wildcard 문자)
- \_ : 한 글자를 의미한다.

```
SELECT * FROM city  
WHERE name LIKE 'New%';
```

	ID	Name	CountryCode	District	Population
▶	137	Newcastle	AUS	New South Wales	270324
	482	Newcastle upon Tyne	GBR	England	189150
	502	Newport	GBR	Wales	139000
	734	Newcastle	ZAF	KwaZulu-Natal	222993
	1106	New Bombay	IND	Maharashtra	307297
	1109	New Delhi	IND	Delhi	301297
	3793	New York	USA	New York	8008278
	3823	New Orleans	USA	Louisiana	484674

```
SELECT * FROM city  
WHERE countrycode LIKE 'K_R';
```

	ID	Name	CountryCode	District	Population
▶	2255	Bikenibeu	KIR	South Tarawa	5055
	2256	Bairiki	KIR	South Tarawa	2226
	2331	Seoul	KOR	Seoul	9981619
	2332	Pusan	KOR	Pusan	3804522
	2333	Inchon	KOR	Inchon	2559424
	2334	Taegu	KOR	Taegu	2548568
	2335	Taejon	KOR	Taejon	1425835
	2336	Kwangju	KOR	Kwangju	1368341

**GROUP BY / HAVING**

## 그룹을 지어 데이터를 묶는다.

- 집계 함수를 사용하기 위해 묶는다.
  - SUM() / AVG()
  - MIN() / MAX()
  - COUNT()

```
SELECT countrycode, sum(population) AS '총인구'  
FROM city  
GROUP BY countrycode;
```

	countrycode	총인구
▶	ABW	29034
	AFG	2332100
	AGO	2561600
	AIA	1556
	ALB	270000
	AND	21189
	ANT	2345
	ARE	1728336
	ARG	19996563
	ARM	1633100
	ASM	7523

```
SELECT countrycode, count(population) AS '도시수'  
FROM city  
GROUP BY countrycode;
```

	countrycode	도시수
▶	ABW	1
	AFG	4
	AGO	5
	AIA	2
	ALB	1
	AND	1
	ANT	1
	ARE	5
	ARG	57
	ARM	3

## GROUP BY을 썼을 때 쓸 수 있는 조건 절

- 집계함수에 대해 조건을 걸 수 있음

```
SELECT countrycode, count(population) AS '도시수' FROM city  
GROUP BY countrycode  
HAVING count(population) > 200;
```

	CountryCode	도시수
▶	BRA	250
	CHN	363
	IND	341
	JPN	248
	USA	274

# Day1-5. SQL 도전미션



# 챕터의 포인트

- 미션 목표
- 도전 1. ssafy.member
- 도전 2. world
- 도전 3. W3Schools TrySQL

## 미션목표

- 총 세 가지 도전 미션이 제공 됨

1. SQL 연습 - “world” Database 의 다량의 Table Query
2. SQL 연습 - “employees” Database 의 다량의 Table Query

## 도전1. world

## 다음 미션을 순서대로 수행하자

1. 나라이름이 United로 시작하는 국가 조회하기

	Name
▶	United Arab Emirates
	United Kingdom
	United States Minor Outlying Islands
	United States

## 다음 미션을 순서대로 수행하자

1. 아시아에 속한 국가(Country Table)들의 이름을 조회 하기

Result Grid			 Filter Row
	Name		
▶	Afghanistan		
	United Arab Emirates		
	Armenia		
	Azerbaijan		
	Bangladesh		
	Bahrain		
	Brunei		
	Bhutan		
	China		
	Cyprus		
	Georgia		
	Hong Kong		
	Indonesia		
	India		
	Iran		
	Iraq		
	Israel		
	Jordan		
	Japan		
	Kazakstan		
	Kyrgyzstan		
	Cambodia		
	South Korea		
	Kuwait		
	Laos		
	Lebanon		
	Sri Lanka		
	Macao		

### 다음 미션을 순서대로 수행하자

1. 대륙 별 평균 인구 구하기
2. 영어 사용비율(Percentage)이 90% 이상인 국가 찾기 (국가 이름 필수)

	Continent	AvgPopulation
▶	Asia	72647562.7451
	South America	24698571.4286
	Europe	15871186.9565
	Africa	13525431.0345
	North America	13053864.8649
	Oceania	1085755.3571
	Antarctica	0.0000

	Name	Percentage
▶	Australia	81.2
	Bermuda	100.0
	United Kingdom	97.3
	Gibraltar	88.9
	Ireland	98.4
	New Zealand	87.0
	Trinidad and Tobago	93.5
	United States	86.2
	Virgin Islands, U.S.	81.7

## 다음 미션을 순서대로 수행하자

1. 각 도시(City Table)의 인구가 100만 이상일 때, 해당 도시의 국가와 도시 이름을 조회하기

	Country	City
▶	Afghanistan	Kabul
	Algeria	Alger
	Angola	Luanda
	Argentina	Buenos Aires
	Argentina	La Matanza
	Argentina	Córdoba
	Armenia	Yerevan
	Australia	Sydney
	Australia	Melbourne
	Australia	Brisbane
	Australia	Perth
	Azerbaijan	Baki
	Bangladesh	Dhaka
	Bangladesh	Chittagong
	Brazil	São Paulo
	Brazil	Rio de Janeiro
	Brazil	Salvador
	Brazil	Belo Horizonte
	Brazil	Fortaleza
	Brazil	Brasília
	Brazil	Curitiba
	Brazil	Recife
	Brazil	Porto Alegre



## 도전2. employees

### 다음 미션을 순서대로 진행하자

- 급여가 40,000 미만인 직원 조회

	first_name	last_name	salary
▶	Shahaf	Famili	39935
	Divier	Reistad	39520
	Pradeep	Makrucki	39765
	Florian	Syrotiuk	39507
	Basil	Tramer	39735
	Udi	Jansch	39551
	Hironoby	Sidou	39567
	Hironoby	Sidou	39724
	Abdulah	Thibadeau	39564
	Abdulah	Thibadeau	39855
	Yuping	Alpin	39836
	Yuping	Alpin	39940
	Yuping	Alpin	39661
	Jackson	Kakkad	39636
	Tze	Nourani	39812
	Basem	Teitelbaum	39973
	Shaunak	Cullers	39871
	Shaunak	Cullers	39539
	Shaunak	Cullers	39838
	Hercules	Benzmuller	39682
	Hercules	Benzmuller	39265
	Masali	Murrill	39773
	Dietrich	DuCasse	39994

### 다음 미션을 순서대로 진행하자

- 입사일별로 직원수를 출력 하시오

	hire_date	num_employees
▶	1986-06-26	83
	1985-11-21	119
	1986-08-28	78
	1986-12-01	94
	1989-09-12	78
	1989-06-02	75
	1989-02-10	89
	1994-09-15	42
	1985-02-18	112
	1989-08-24	65
	1990-01-22	75
	1992-12-18	52
	1985-10-20	99
	1987-03-11	91
	1987-07-02	93
	1995-01-27	36
	1993-08-03	41
	1987-04-03	97
	1999-04-30	5
	1991-01-26	72
	1988-02-10	91
	1995-08-22	32

### 다음 미션을 순서대로 진행하자

- 'Sales' 부서에 소속된 직원들의 이름과 입사일(hire\_date)을 출력 하시오
  - 힌트 - join 2번

	first_name	last_name	hire_date
▶	Bezalel	Simmel	1985-11-21
	Kazuhito	Cappelletti	1995-01-27
	Bader	Swan	1988-09-21
	Uri	Lenart	1989-11-12
	Yinghua	Dredge	1990-12-25
	Sanjiv	Zschoche	1986-02-04
	Breannda	Billingsley	1987-11-02
	Tse	Herber	1985-09-17
	Charlene	Brattka	1987-08-07
	Xinglin	Eugenio	1986-09-08
	Jungsoon	Syrzycki	1988-09-02
	Sudharsan	Flasterstein	1986-08-12
	Sailaja	Desikan	1996-11-05
	Hilari	Morton	1986-07-15
	Valter	Sullins	1988-10-18
	Perla	Heyers	1992-12-28
	Dunn	Rara	1994-03-27

# 내일 방송에서 만나요!

삼성청년SW·AI아카데미

과제1(93p)

도전 1-1. world

과제2(94p)

도전 1-2. world

과제3(95p)

도전 1-3. world

과제4(96p)

도전 1-4. world

과제5(98p)

도전 2-1. employees

과제6(99p)

도전 2-2. employees

과제7(100p)

도전 2-3. employees