

# DataBase

김지성 강사

# 데이터

세상에는 정말 많은 데이터가 존재

아이디 또는 전화번호

비밀번호

회원 정보



좋아요



구독

# 데이터

각 데이터는 관련된 회사의 데이터베이스에 저장

아이디 또는 전화번호
<input type="text"/>
비밀번호
<input type="password"/>



N



# 데이터베이스란?

---

## DataBase (DB)

여러 사람에 의해 공유되어 사용될 목적으로 통합하여 관리되는 **데이터의 집합**

자료항목의 중복을 없애고 자료를 구조화하여 저장함으로써 자료 검색과 갱신의 효율을 높인다.

현대적인 의미의 데이터베이스 개념을 확립한 사람은 당시 제너럴일렉트릭사(社)에 있던 **C.바크만**으로,  
그는 1963년 IDS(Integrated Data Store)라는 데이터베이스 관리시스템을 만들었다.

[네이버 지식백과] 데이터베이스 [data base] (두산백과)

<https://db-engines.com/en/ranking>

# 데이터가 저장된 형태

데이터 : **테이블** 단위로 저장됨


테이블 = 표

# 데이터가 저장된 형태

테이블 : 표형태로 저장된 데이터의 집합

actor_id	first_name	last_name	last_update
1	PENELOPE	GUINNESS	2006-02-15 04:34:33
2	NICK	WAHLBERG	2006-02-15 04:34:33
3	ED	CHASE	2006-02-15 04:34:33
4	JENNIFER	DAVIS	2006-02-15 04:34:33
5	JOHNNY	LOLLOBRIGIDA	2006-02-15 04:34:33
6	BETTE	NICHOLSON	2006-02-15 04:34:33
7	GRACE	MOSTEL	2006-02-15 04:34:33

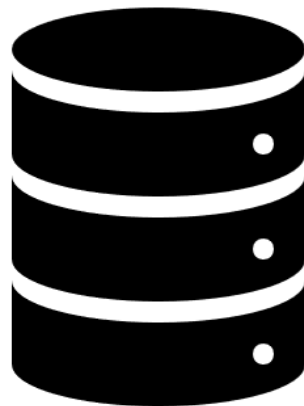
# 데이터가 저장된 형태

하나의 DB에 여러 개의 테이블 생성 가능

actor_id	first_name	last_name	last update			
1	PENELOPE	GU	city_id	city	country_id	last_update
2	NICK	W	1	A Coruña (La Coruña)	87	2006-02-15 04:45:25
3	ED	CH	2	Abha	82	2006-02-15 04:45:25
4	JENNIFER	DA	3	Abu Dhabi	101	2006-02-15 04:45:25
5	JOHNNY	LO	4	Acuña	60	2006-02-15 04:45:25
			5	Adana	97	2006-02-15 04:45:25
			6	Addis Abeba	31	2006-02-15 04:45:25
			7	Aden	107	2006-02-15 04:45:25

address_id	address	address				
1	47 MySakila Drive	ALBERTA	Alberta	300	14033335568	2014-09-25 22:30:27
2	28 MySQL Boulevard	QUEENSLAND	QLD	576	6172235589	2014-09-25 22:30:09
3	23 Workhaven Lane	NAGASAKI	Nagasaki	463	35200	28303384290
4	1411 Lilydale Drive	CALIFORNIA	California	449	17886	838635286649
5	1913 Hanoi Way	ATTIKA	Attika	38	83579	448477190408
6	1121 Loja Avenue					
7	692 Joliet Street					



DB

# 데이터가 저장된 형태

actor

actor_id	first_name	last_name	last_update
1	PENELOPE	GUINNESS	2006-02-15 04:34:33
2	NICK	WAHLBERG	2006-02-15 04:34:33
3	ED	CHASE	2006-02-15 04:34:33
4	JENNIFER	DAVIS	2006-02-15 04:34:33
5	JOHNNY	LOLLOBRIGIDA	2006-02-15 04:34:33
6	BETTE	NICHOLSON	2006-02-15 04:34:33
7	GRACE	MOSTEL	2006-02-15 04:34:33

city

city_id	city	country_id	last_update
1	A Coruña (La Coruña)	87	2006-02-15 04:45:25
2	Abha	82	2006-02-15 04:45:25
3	Abu Dhabi	101	2006-02-15 04:45:25
4	Acuña	60	2006-02-15 04:45:25
5	Adana	97	2006-02-15 04:45:25
6	Addis Abeba	31	2006-02-15 04:45:25
7	Aden	107	2006-02-15 04:45:25

address_id	address	address2	district	city_id	postal_code	phone	location	last_update
1	47 MySakila Drive	NULL	Alberta	300			BLN	2014-09-25 22:30:27
2	28 MySQL Boulevard	NULL	QLD	576			BLN	2014-09-25 22:30:09
3	23 Workhaven Lane	NULL	Alberta	300		14033335568	BLN	2014-09-25 22:30:27
4	1411 Lillydale Drive	NULL	QLD	576		6172235589	BLN	2014-09-25 22:30:09
5	1913 Hanoi Way		Nagasaki	463	35200	28303384290	BLN	2014-09-25 22:31:53
6	1121 Loja Avenue		California	449	17886	838635286649	BLN	2014-09-25 22:34:01
7	692 Joliet Street		Attika	38	83529	448477190408	BLN	2014-09-25 22:31:07

address

테이블마다 관련 정보를 저장하여 여러 개의 테이블을 만들면  
많은 데이터를 체계적으로 관리할 수 있음



## 데이터가 저장된 형태

---



데이터베이스도 여러 개 만듦으로서  
방대한 데이터도 체계적으로 관리 가능

# 테이블 구조

이름	전화번호	가입일	상품	수량
홍길동	010-3242-5931	2019-01-02	우유	1
홍길동	010-3242-5931	2019-01-03	식빵	2
이동진	010-3943-1992	2018-12-21	치즈	1
박철우	010-6123-4453	2018-12-23	소금	1
박철우	010-6123-4453	2018-12-25	우유	3

**Row (행)**  
개체 한 개

**Column (열)** 개체의 속성

5개의 **행**과 5개의 **열**을 갖는 테이블

# DBMS

김지성 강사

# DBMS란?

---

## DataBase Management System

데이터베이스 관리 시스템

# DBMS란?

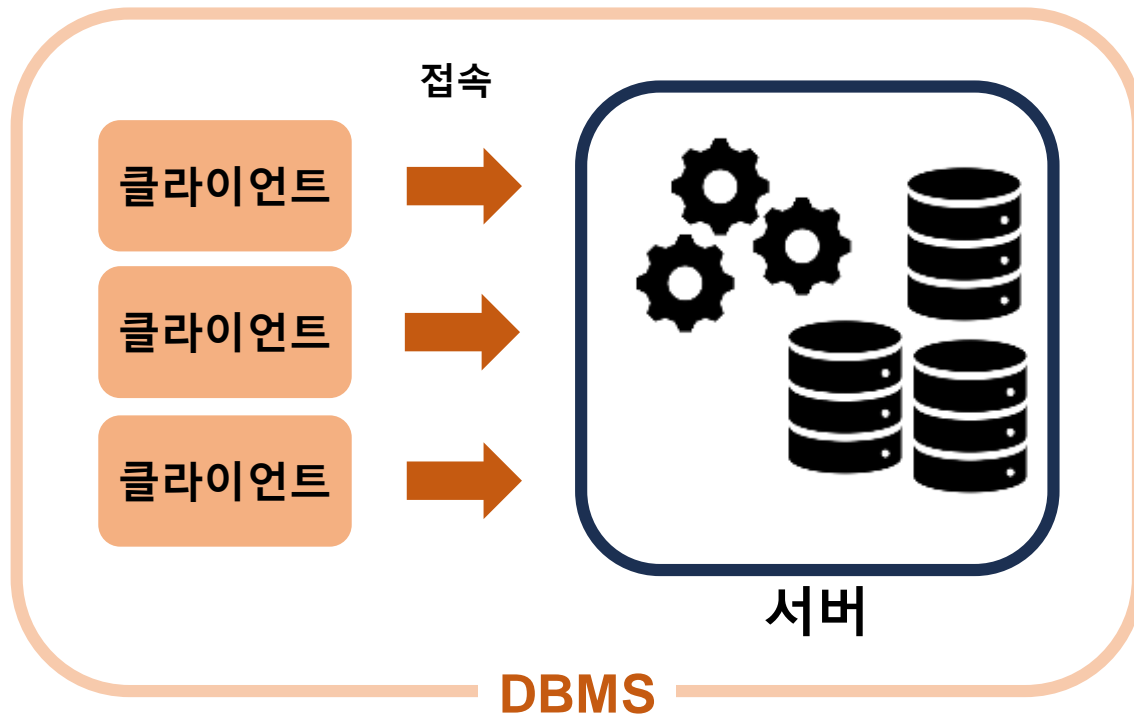


DBMS를 통해 **데이터 관리** 가능

사용자의 명령에 따라 **DB or table** 생성 및 삭제, 데이터 추가, 조회 및 삭제 가능

# DBMS 구성 요소

클라이언트를 통해 sever에 접속하는 구조



# DBMS 구성 요소

---

## ✓ client (클라이언트 프로그램)

사용자가 server에 접속해서 원하는 데이터베이스 관련 작업을 할 수 있도록 SQL을 입력할 수 있는 화면 등을 제공하는 **프로그램**

## ✓ server (서버 프로그램)

클라이언트로부터 SQL문 등을 전달 받아 데이터베이스 관련 작업을 직접 처리하는 **프로그램**

# 서버-클라이언트 구조

---

## ✓ MySQL

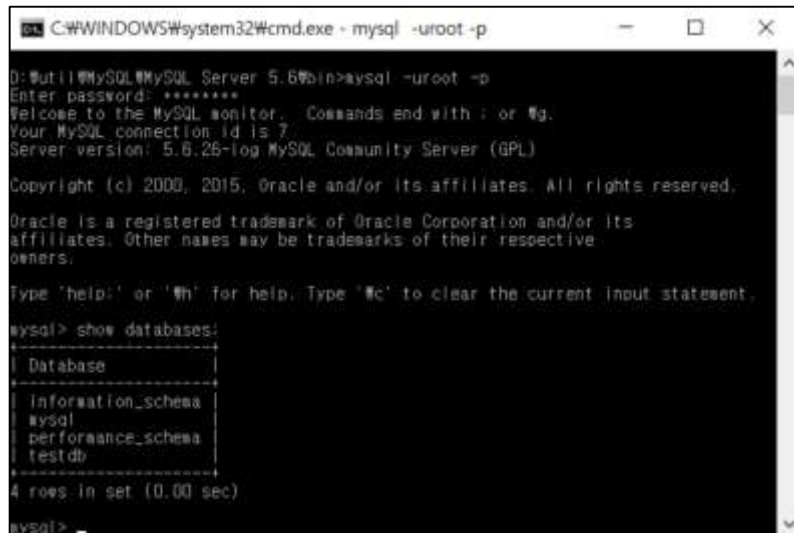
- **서버** 프로그램
  - 이름 : mysqld
  - 사용자가 클라이언트 프로그램을 통해 접속할 수 있음
- **클라이언트** 프로그램
  - 이름 : mysql
  - 주로 CLI(Command-Line Client)환경에서 사용 됨
  - GUI(Graphical User Interface) 환경 제공 프로그램 : MySQL Workbench



# 서버-클라이언트 구조

## ✓ MySQL

### CLI 환경

A screenshot of a Windows command prompt window titled 'C:\WINDOWS\system32\cmd.exe - mysql -uroot -p'. The prompt shows the user entering 'mysql -uroot -p' and then the MySQL monitor interface. The user enters a password and is greeted with 'Welcome to the MySQL monitor. Commands end with ; or \g. Your MySQL connection id is 7. Server version: 5.6.26-log MySQL Community Server (GPL)'. The user then enters 'show databases;' and the output is displayed in a table format with 4 rows in 1 set.

```
C:\WINDOWS\system32\cmd.exe - mysql -uroot -p
D:\util\MySQL\MySQL Server 5.6\bin>mysql -uroot -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7.
Server version: 5.6.26-log MySQL Community Server (GPL)

Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.

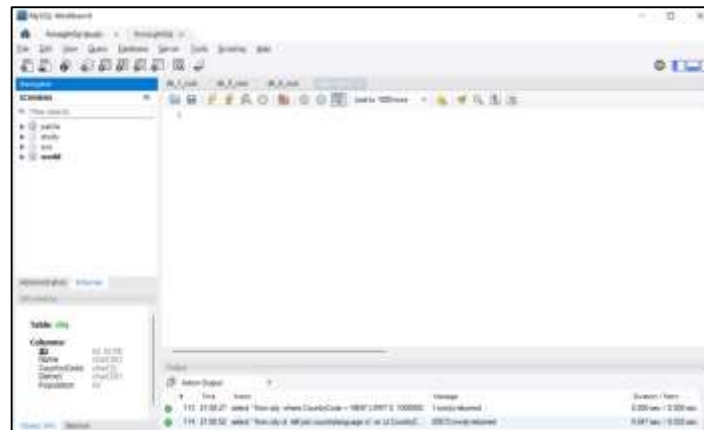
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| testdb |
+-----+
4 rows in set (0.00 sec)

mysql>
```

### GUI 환경



# DBMS를 사용한다?

---

실행되고 있는 server에 client를 이용해서 접속한 후  
원하는 명령어를 내리는 것

# DBMS 종류

---

<https://db-engines.com/en/ranking>

# DBMS 종류

---

## ✓ ORACLE

- 미국 오라클(ORACLE)사의 관계형 DBMS
- 안정성과 유지보수를 보장받을 수 있다는 장점 때문에 많이 사용됨
- 대량의 정보관리 할 때 타 DBMS보다 좋은 성능 보임
- 오라클 자체 SQL쿼리 사용

## ✓ MySQL

- 관계형 DBMS
- 오픈소스라서 무료로 사용 가능
- 기술적 한계가 있음

# DBMS 종류

---

## ✓ MariaDB

- 관계형 DBMS
- 오픈소스라서 무료로 사용 가능
- MySQL 개발진들이 오라클 정책이 추구하는 바와 맞지 않아 따로 나와서 개발한 DB
- MySQL과 거의 99% 호환성을 갖고있음
- MySQL에 비해 속도와 성능면서에서 더 향상 됨

# 관계형 DB

김지성 강사

# 관계형 DB란?

---

- RDB : Relational DataBase
- 관계형 데이터 모델에 기초를 둔 데이터베이스
- 모든 데이터를 2차원 테이블 형태로 표현한 뒤 각 테이블 간의 관계를 정의

# 관계형 DB란?

## Database : world

Table : city

ID	Name	CountryCode	District	Population
1	Kabul	AFG	Kabul	1780000
2	Qandahar	AFG	Qandahar	237500
3	Herat	AFG	Herat	186800
4	Mazar-e-Sharif	AFG	Balkh	127800
5	Amsterdam	NLD	Noord-Holland	731200
6	Rotterdam	NLD	Zuid-Holland	593321
7	Haag	NLD	Zuid-Holland	440900
8	Utrecht	NLD	Utrecht	234323
9	Eindhoven	NLD	Noord-Brabant	201843

Table : countrylanguage

CountryCode	Language	IsOfficial	Percentage
ABW	Dutch	T	5.3
ABW	English	F	9.5
ABW	Papiamentu	F	76.7
ABW	Spanish	F	7.4
AFG	Balochi	F	0.9
AFG	Dari	T	32.1
AFG	Pashto	T	52.4
AFG	Turkmenian	F	1.9

Table : country

Code	Name	Continent	Region	SurfaceArea	IndepYear
ABW	Aruba	North America	Caribbean	193.00	1986
AFG	Afghanistan	Asia	Southern and Central Asia	652090.00	1919
AGO	Angola	Africa	Central Africa	1246700.00	1975
AIA	Anguilla	North America	Caribbean	96.00	1980
ALB	Albania	Europe	Southern Europe	28748.00	1912
AND	Andorra	Europe	Southern Europe	468.00	1278
ANT	Netherlands Antilles	North America	Caribbean	800.00	1980
ARE	United Arab Emirates	Asia	Middle East	83600.00	1971
ARG	Argentina	South America	South America	2780400.00	1816
ARM	Armenia	Asia	Middle East	29800.00	1991



# SQL

---

## SQL : Structured Query Language

DBMS에서 데이터를 **관리**하고 **조작**하기 위한 표준 프로그래밍 언어

ANSI(미국 국가 표준 협회)와 ISO(국제 표준화 기구)에서 표준으로 정한 언어  
모든 DBMS가 표준 SQL을 쓰는 건 아님

# Oracle vs ANSI

---

## ✓ Oracle SQL

```
SELECT a.ename ,b.ename  
FROM scott.emp a, scott.emp b  
WHERE a.mgr (+) = b.empno
```

## ✓ ANSI SQL

```
SELECT a.ename,b.ename  
FROM scott.emp a  
LEFT OUTER JOIN scott.emp b  
ON a.mgr = b.empno
```

# SQL

명령어 종류	명령어	설명
데이터 조작어 (DML: Data Manipulation Language)	SELECT	데이터베이스에 있는 데이터를 조회하거나 검색
	INSERT	데이터베이스에 데이터를 입력
	UPDATE	데이터베이스에 데이터를 수정
	DELETE	데이터베이스에 데이터를 삭제
데이터 정의어 (DDL: Data Definition Language)	CREATE	데이터베이스 구조생성 (ex 테이블, 데이터베이스)
	ALTER	데이터베이스 구조변경
	DROP	데이터베이스 구조삭제
	RENAME	데이터베이스 이름변경
데이터 제어어 (DCL: Data Control Language)	GRANT	데이터베이스에 접근하는 권한 생성
	REVOKE	데이터베이스에 접근하는 권한 삭제
트랜잭션 제어어 (TCL: Transaction Control Language)	COMMIT	논리적인 작업단위를 묶어서 작업단위(트랜잭션) 별로 제어하는 명령어
	ROLLBACK	

# NoSQL

---

- **NoSQL : Not Only SQL**
- 다양한 데이터 모델과 쿼리 방식 지원
- 정의된 스키마가 없음 (데이터의 구조가 자유로움)
- 다양한 형식의 데이터를 유연하게 처리 가능
- 문서지향, 키-값, 열 지향(열 단위로 데이터 저장), 그래프 등의 데이터 모델 지원

# NoSQL

---

## ✓ MongoDB

- 문서지향 데이터베이스
- Binary JSON을 사용하여 BSON 형식의 문서로 저장 됨
- 각 문서는 키-값 쌍으로 구성 됨

## ✓ Redis

- 인메모리 저장 : 데이터를 메모리에 저장
- 데이터 캐시와 세션 저장소로 사용 됨

# ERD

김지성 강사

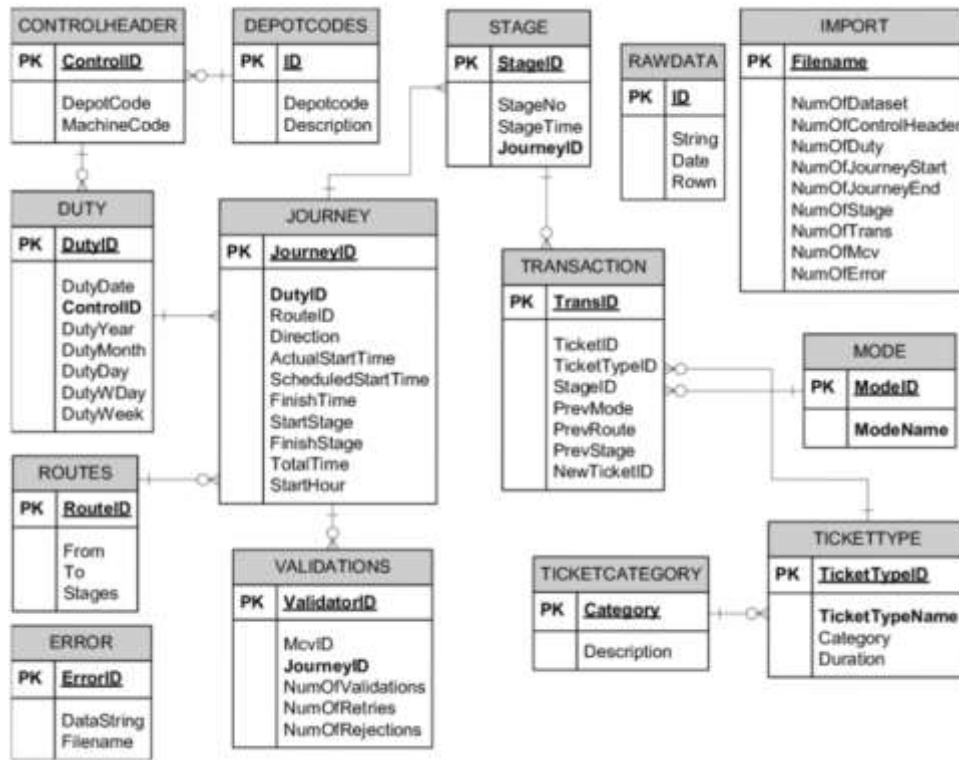
# ERD란?

## ✓ ERD (Entity Relationship Diagram)

시스템에 어떤 엔터티들이 존재하며  
그들 간에 어떤 관계가 있는지를 나타내는 다이어그램



# ERD란?





# ERD 작성 이유

---

데이터베이스의 설계의 핵심을 시각적으로 표현하고  
**데이터 구조를 명확하게** 이해하고 정의하기 위함

# ERD

- ✓ 엔터티(Entity) : 데이터베이스 테이블
- ✓ 인스턴스(Instance) : 데이터베이스에 저장된 내용의 집합
- ✓ 속성(Attribute) : 인스턴스의 구성 요소로서 더 이상 분리되지 않는 단위

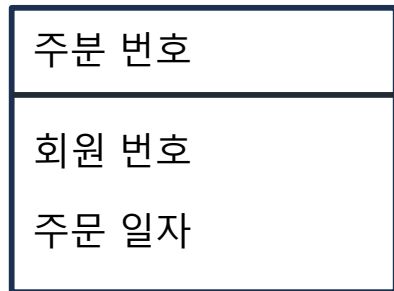


# Entity(엔터티)란?

식별 가능한 객체

업무에서 쓰이는 데이터를 용도별로 분류한 그룹

주문



주문 상품



# 인스턴스란?

상품 엔터티에 '새우깡'이라는 상품과 '자갈치'라는 상품이 있다면  
각각은 상품 엔터티의 **인스턴스**가 됨

## 상품

인스턴스 (= 행 row)

상품 코드	상품명	카테고리
100001	새우깡	과자
100002	자갈치	과자
100003	코카콜라	음료
100004	서울우유	유제품

# 속성이란?

각 엔터티는 자신을 더 상세히 나타내기 위해 **속성(Attribute)**를 갖게 됨  
하나의 **인스턴스**를 구체적으로 나타내주는 데이터

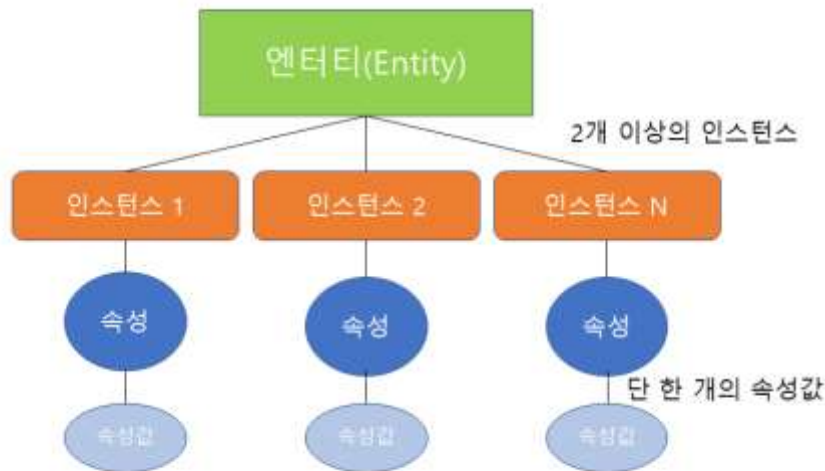
상품

속성 (=칼럼 column)

상품 코드	상품명	카테고리
100001	새우깡	과자
100002	자갈치	과자
100003	코카콜라	음료
100004	서울우유	유제품

# 엔티티, 인스턴스, 속성, 속성 값

- 한 개의 엔티티는 두 개 이상의 인스턴스를 갖는다
- 한 개의 인스턴스는 두 개 이상의 속성을 갖는다
- 한 개의 속성은 하나의 속성값을 갖는다



# 엔터티, 인스턴스, 속성, 속성 값

만약 하나의 속성이 여러 개의 속성값을 갖는 경우  
별도의 엔터티로 분리하는 것이 바람직함

이름	생년월일	직업
이지은	19930516	배우, 가수, 작곡가
...	...	...



이름	생년월일
이지은	19930516

이름	직업
이지은	배우
이지은	가수
이지은	작곡가

# 엔터티, 인스턴스, 속성, 속성 값

중복되는 데이터가 많으면  
엔터티를 분리 해주는 것이 바람직

주문번호	회원번호	주문일자	상품번호	주문수량
10001	1992	240818	2424	3
10001	1992	240818	2425	2
10001	1992	240818	2427	5



# 식별자

인스턴스가 중복되거나 식별이 모호하면 설계가 잘못된 것  
각 인스턴스를 식별할 수 있는 식별자가 필요

세 명의 아이디가 동일하면  
각 회원을 구분할 수 없음

회원아이디	상품번호	주문일자
test1	2424	240818
test1	2427	240620
test1	2450	240717

# 식별자

인스턴스가 중복되거나 식별이 모호하면 설계가 잘못된 것  
각 인스턴스를 식별할 수 있는 식별자가 필요

식별자를 통해  
같은 아이디의 회원들을  
각각 구분할 수 있도록 함

회원번호	회원아이디	상품번호	주문일자
1001	test1	2424	240818
1002	test1	2427	240620
1003	test1	2450	240717

# 식별자 종류

- **PK** (Primary Key) : 엔터티의 인스턴스들을 식별할 수 있는 속성
- **FK** (Foreign Key) : 다른 엔터티의 속성에서 가져온 속성
- **일반 속성** : PK, FK를 제외한 나머지 속성

**PK**

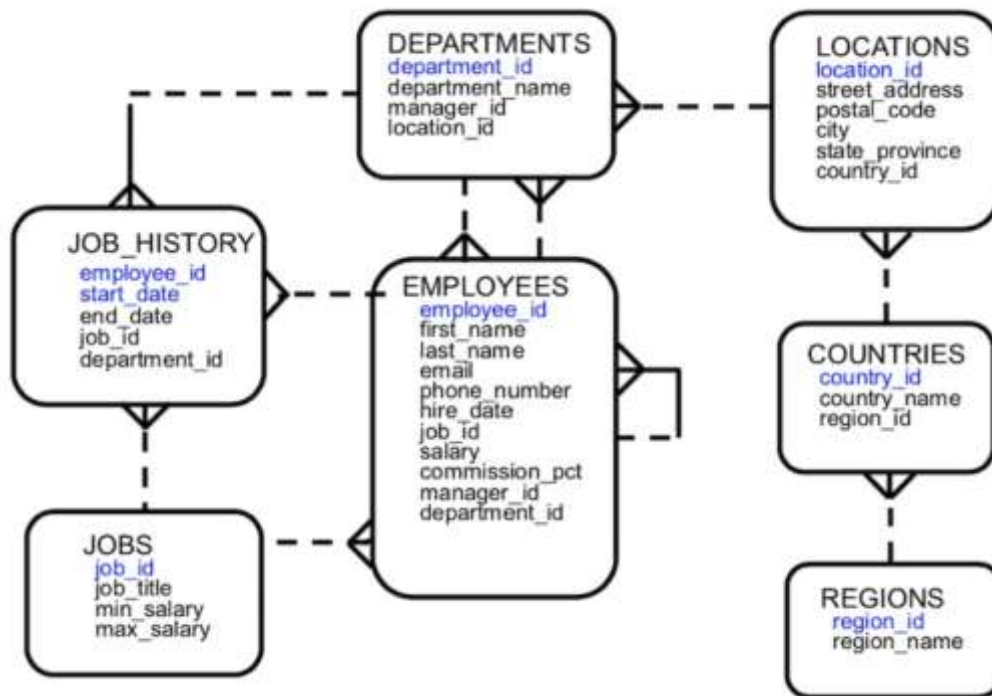
회원번호	회원아이디	상품번호	주문일자
1001	test1	2424	240818
1002	test1	2427	240620
1003	test1	2450	240717

# 정규화

---

관계형 데이터베이스의 설계에서 데이터 중복을 줄이고  
데이터 무결성을 개선하기 위해 데이터를 정규형에 맞도록  
구조화 하는 프로세스

# ERD 예시



# DBMS 설치

김지성 강사

# Mysql 설치

<https://downloads.mysql.com/archives/installer/>

MySQL Product Archives
MySQL Installer (Archived Versions)

Please note that these are old versions. New releases will have recent bug fixes and features! To download the latest release of MySQL Installer, please visit [MySQL Downloads](#).

Product Version: 8.0.37
Operating System: Microsoft Windows

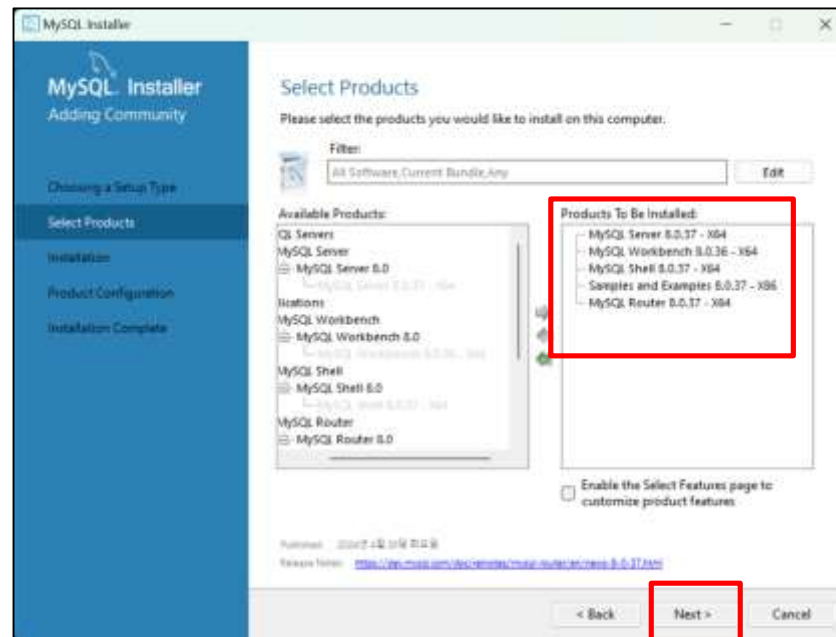
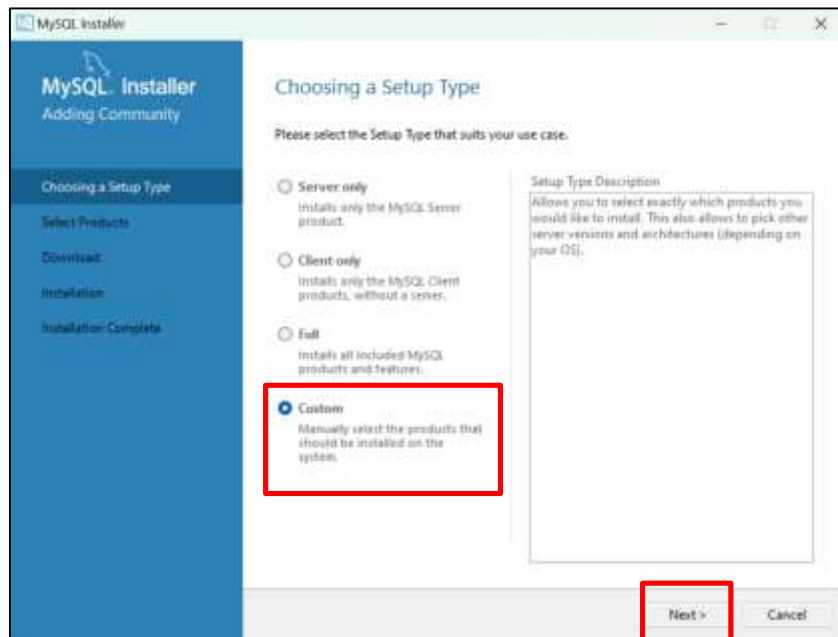
Windows (x86, 32-bit), MSI Installer	Apr 15, 2024	2.1M	<a href="#">Download</a>
Windows (x86, 32-bit), MSI Installer	Apr 15, 2024	296.1M	<a href="#">Download</a>

We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the packages you download.

MySQL open source software is provided under the GPL License.

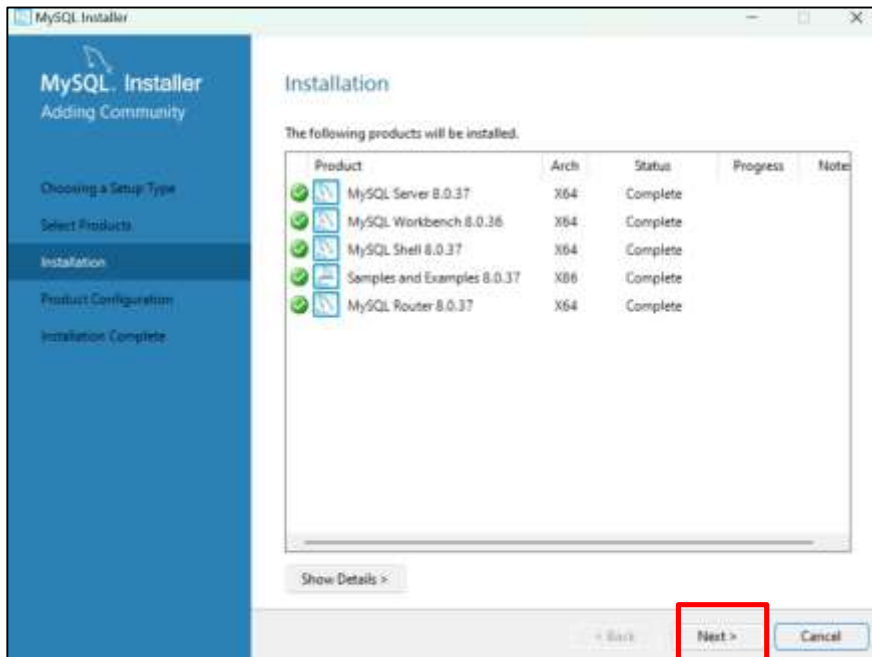
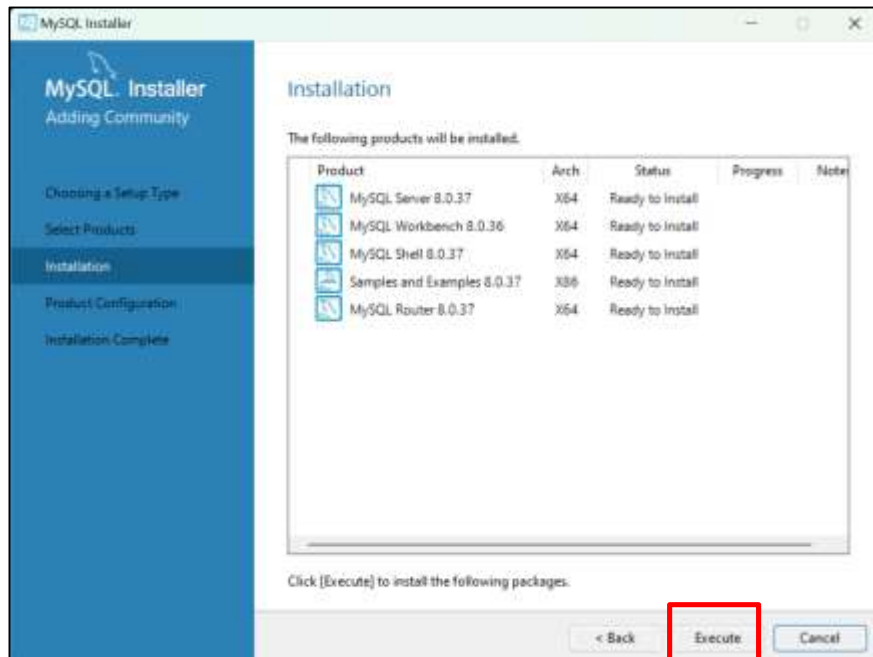
# Mysql 설치

<https://downloads.mysql.com/archives/installer/>

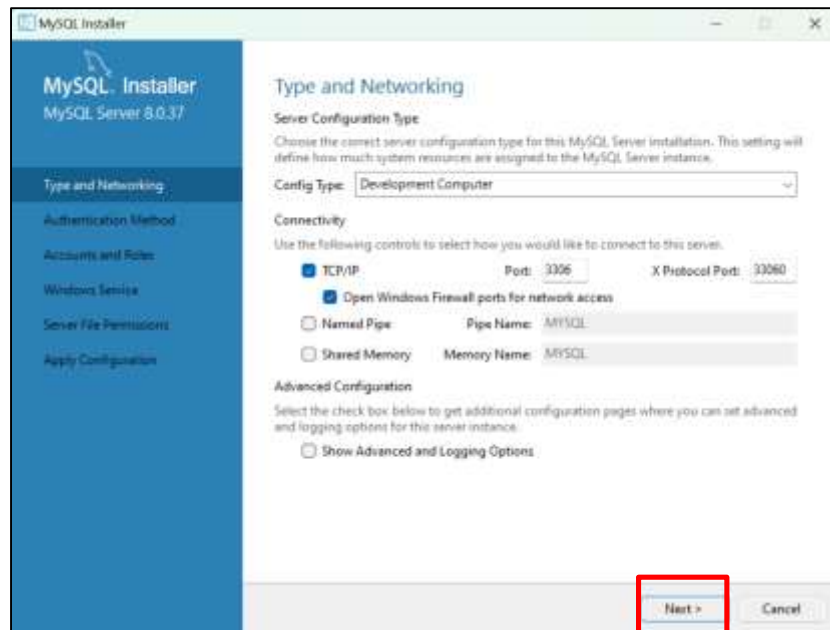
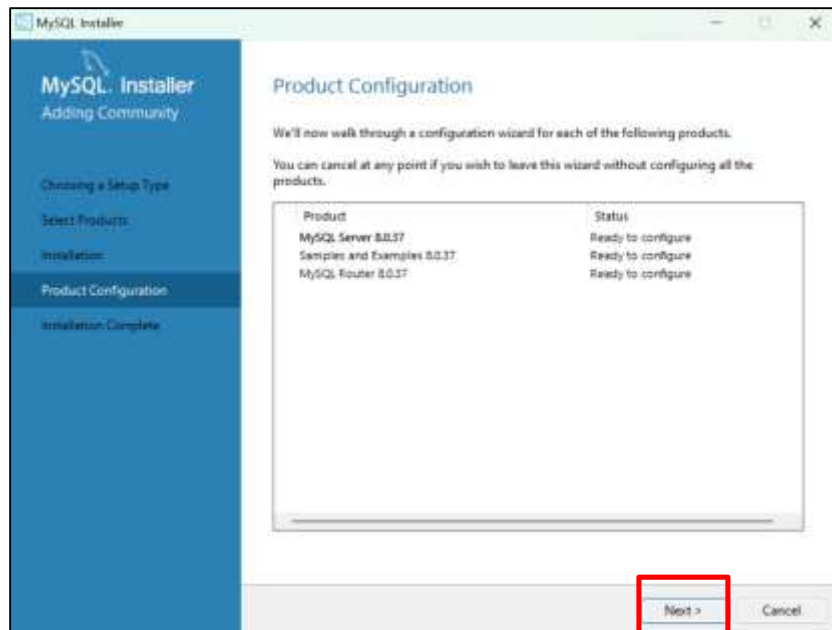




# Mysql 설치



# Mysql 설치



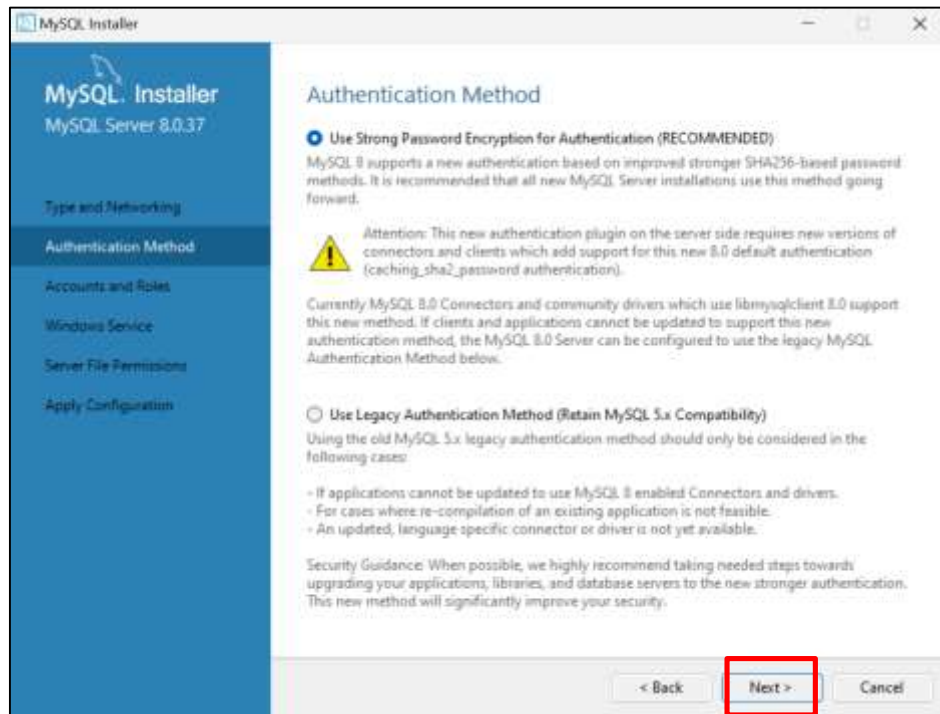
인증방식 설정 – 기본값으로 설정하기

# Mysql 설치

---

- **TCP/IP** : 네트워크 연결 방식
- **Port** : mysql 서버가 사용하는 기본 포트
- **X protocol port** : mysql 8.0 이상에서 사용하는 포트, 서버가 X protocol 통해 클라이언트와 통신, 새로운 어플리케이션 개발 위해 JSON, CRUD, API, SQL 등의 기능 제공
- **Open windows Firewall ports for network access** : mysql 서버가 네트워크를 통해 접근할 수 있도록
- 윈도우 방화벽 설정을 자동으로 구성 -> 클라이언트가 방화벽으로 인한 접속 문제 없이 서버에 접속할 수 있음
- **Named Pipe** : windows에서 로컬 프로세스간 통신을 위해 사용되는 방식, 주로 동일한 물리적 서버 내에서 서버와 클라이언트가 통신할 때 사용
- **Shared Name** : 또 다른 로컬 통신 방식, 동일한 서버 내의 mysql 서버와 클라이언트 간의 빠른 통신 위해 사용됨

# Mysql 설치



- 대부분의 DBMS는 서버-클라이언트 구조
- 클라이언트는 서버에 접속하기 위해 DB관련 작업을 해줘야 함
- 이 때 서버에 접속하기 위해 사용자 계정이 필요
- Mysql 설치 시 가장 처음에 root라는 사용자가 기본으로 준비 돼있음
- 이 root 계정으로 접속하게 될텐데 이 때 비번이 필요
- 비번은 재설정이 어려울 수 있으니 여러 번 확인 후 잘 기억해두기

# Mysql 설치

MySQL Installer  
MySQL Server 8.0.37

Accounts and Roles

Root Account Password  
Enter the password for the root account. Please remember to store this password in a secure place.

MySQL Root Password:

Repeat Password:

Password strength: Medium

MySQL User Accounts  
Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.

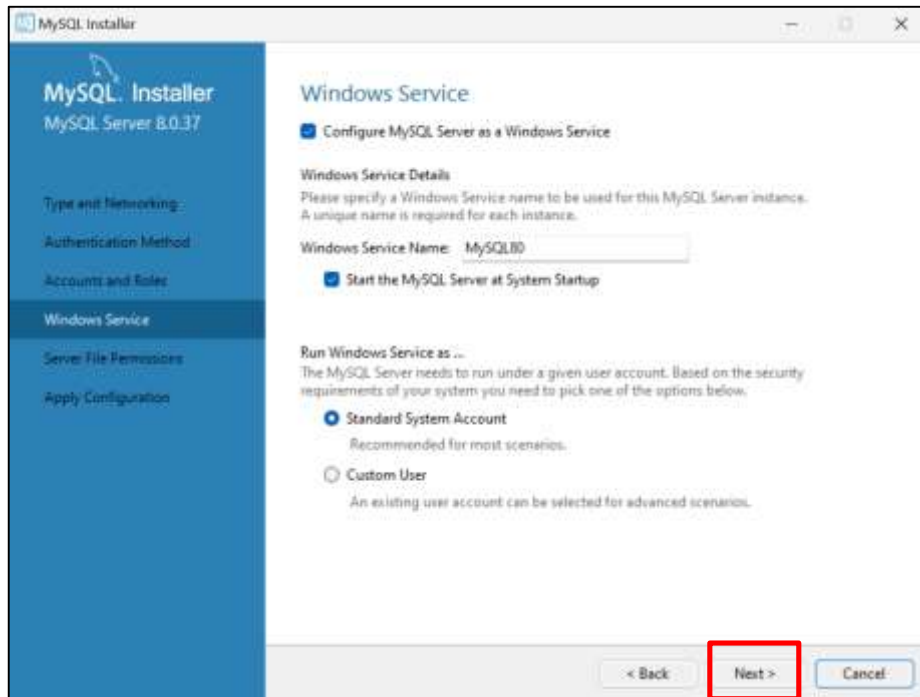
MySQL User Name	Host	User Role

Buttons: Add User, Edit User, Delete

Navigation: < Back, **Next >**, Cancel

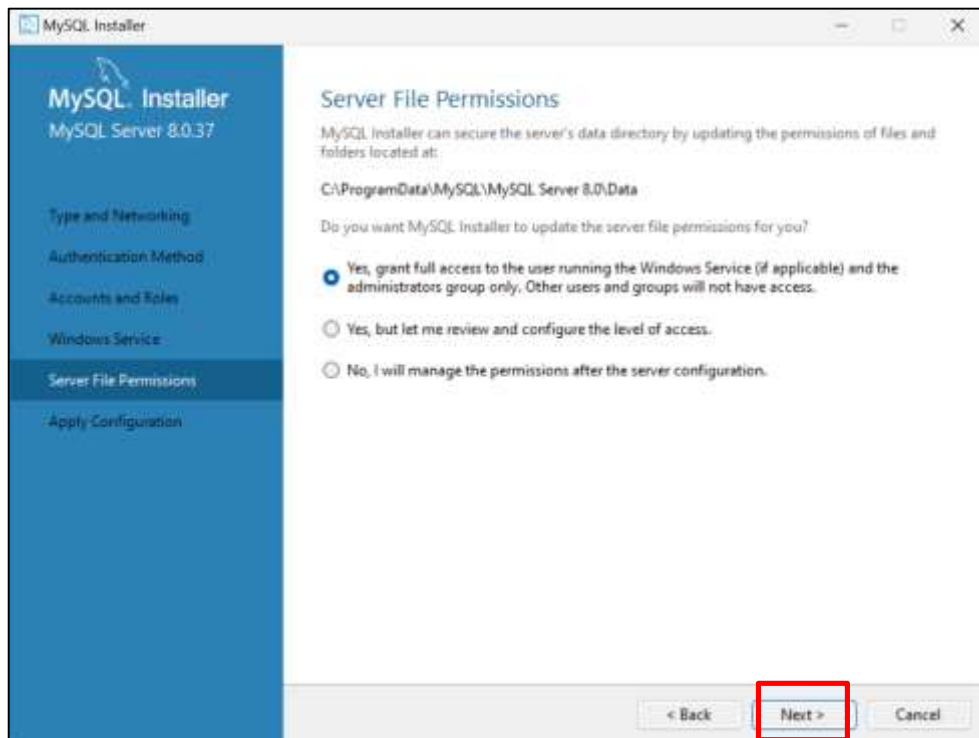
**각자 비밀번호 설정 후 꼭 기억하기!**

# Mysql 설치



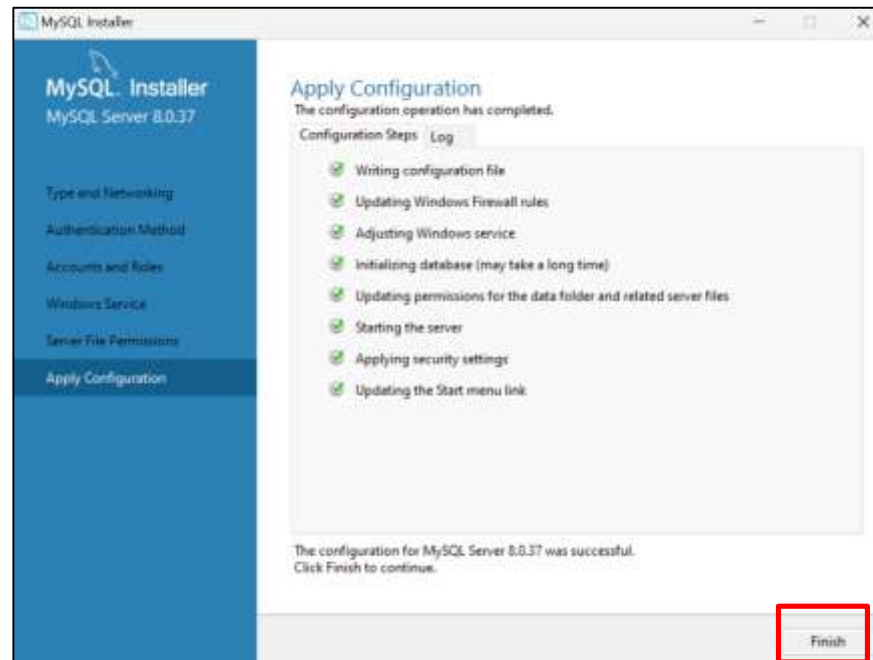
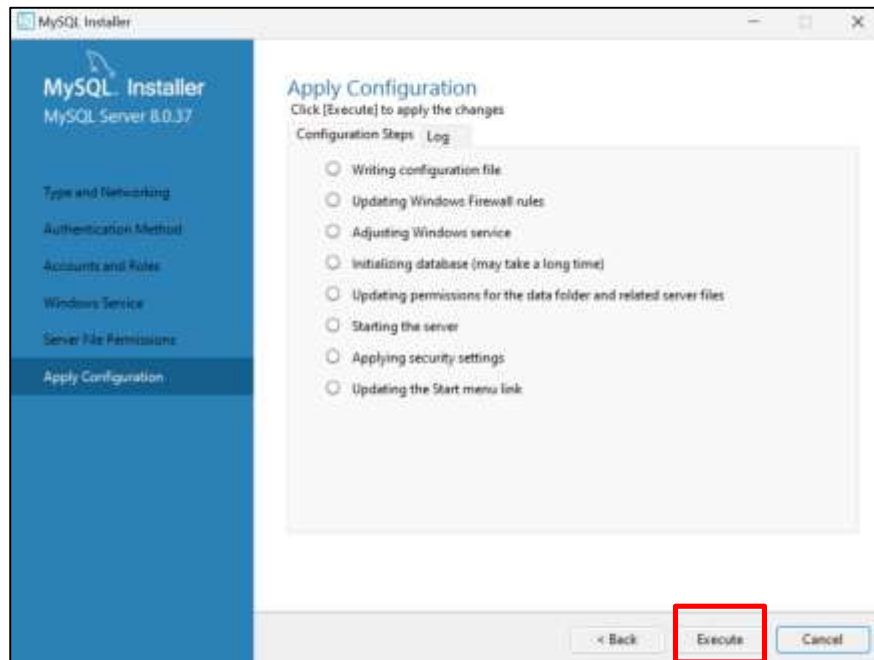
Workbench 실행 할 때 마다 서버가 자동으로  
실행 될건지 여부 확인

# Mysql 설치



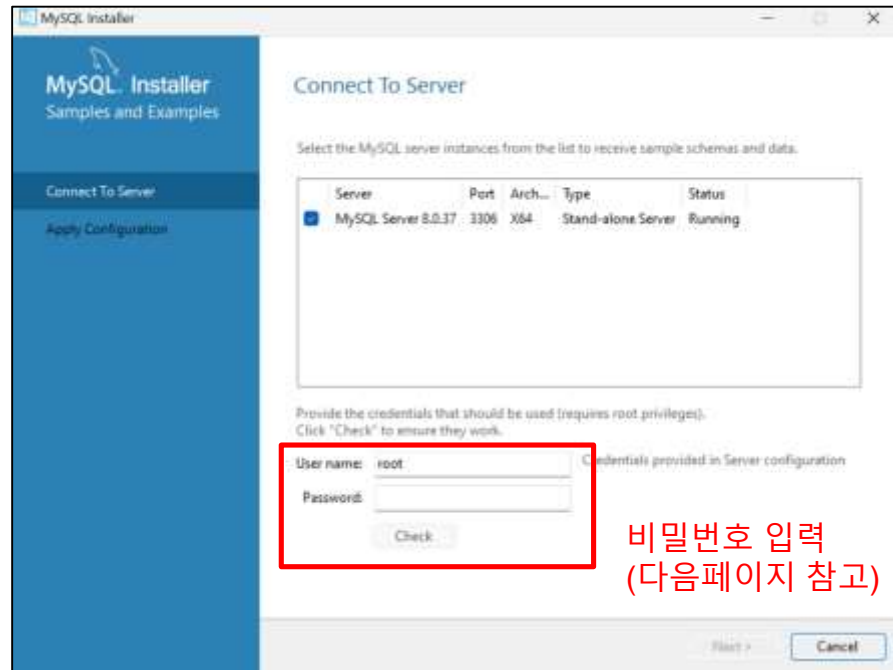
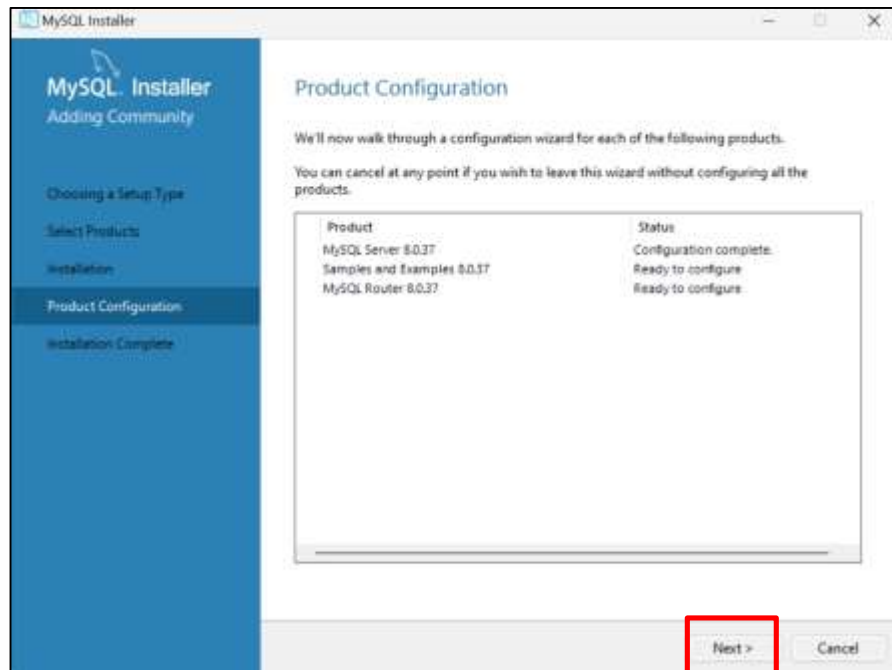
서버 파일에 대한 권한 설정

# Mysql 설치

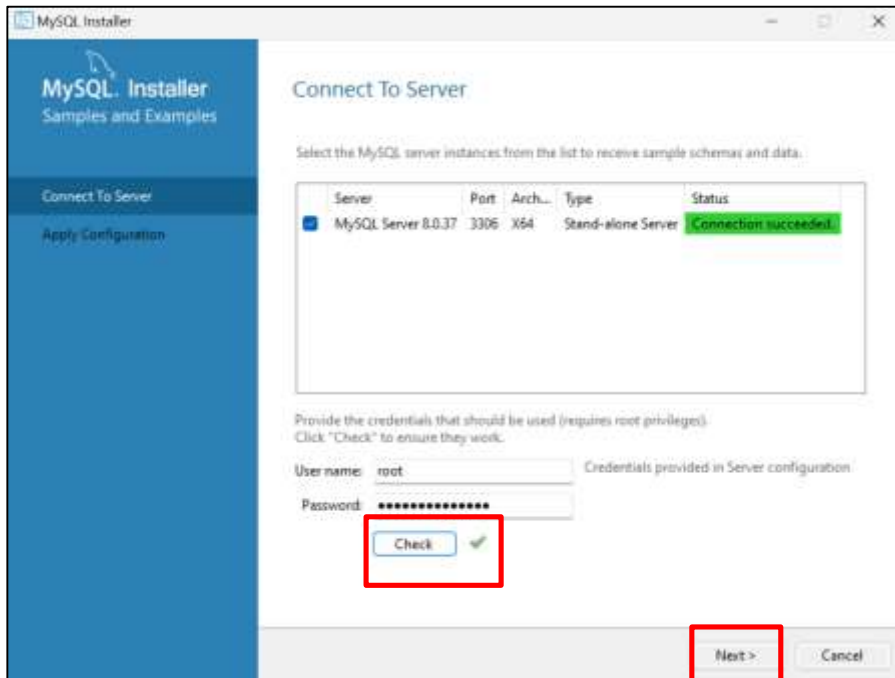
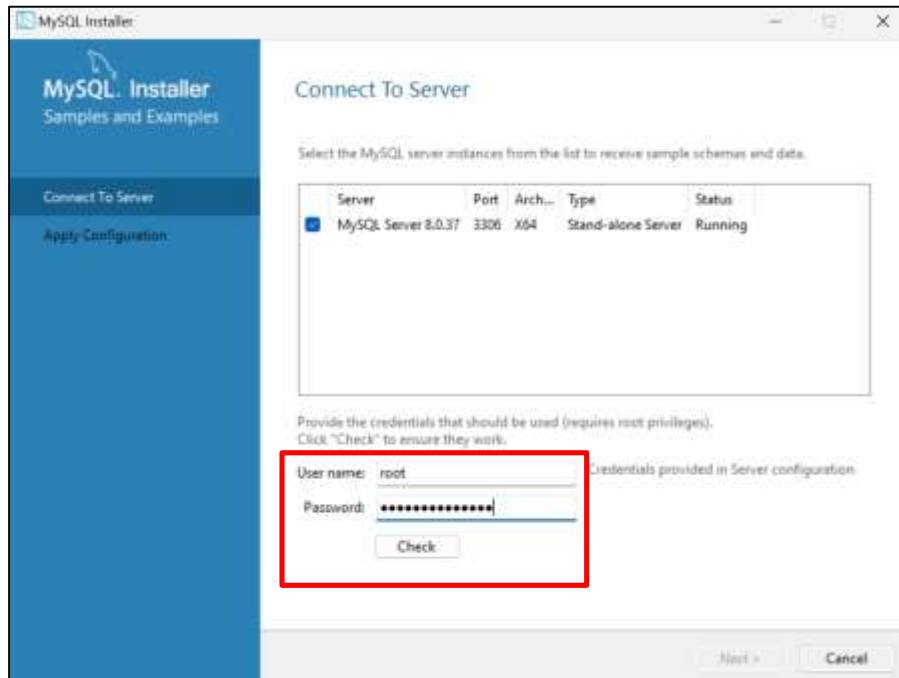




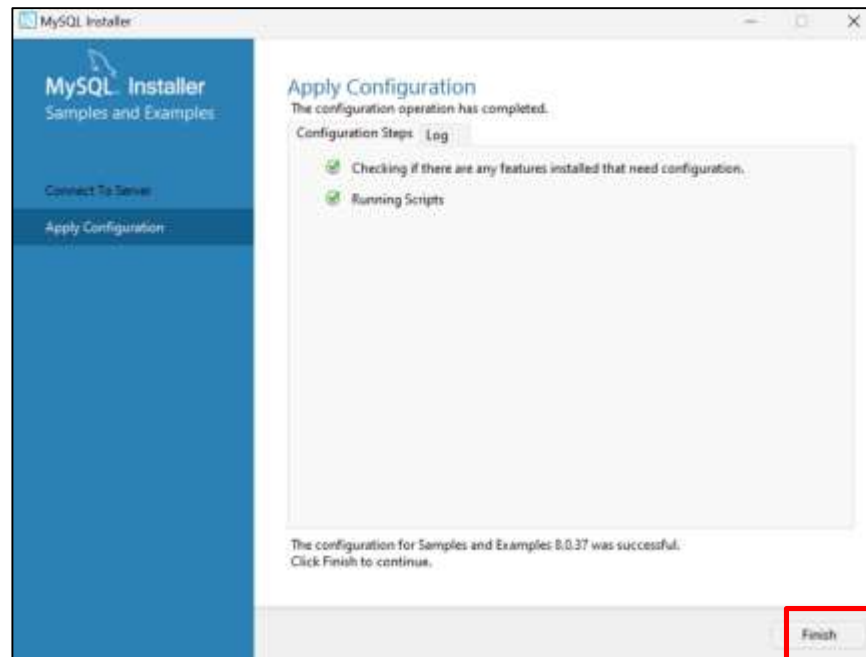
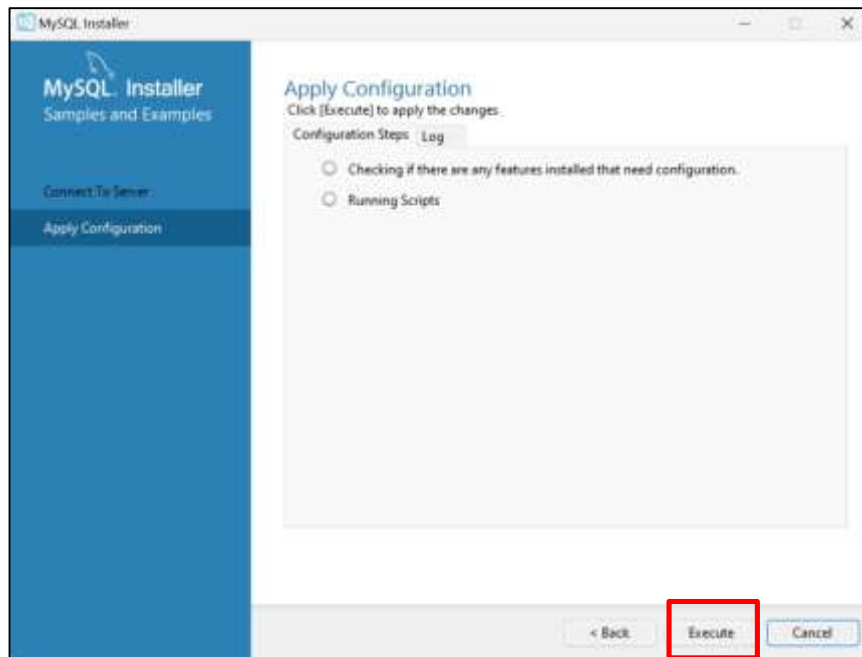
# Mysql 설치



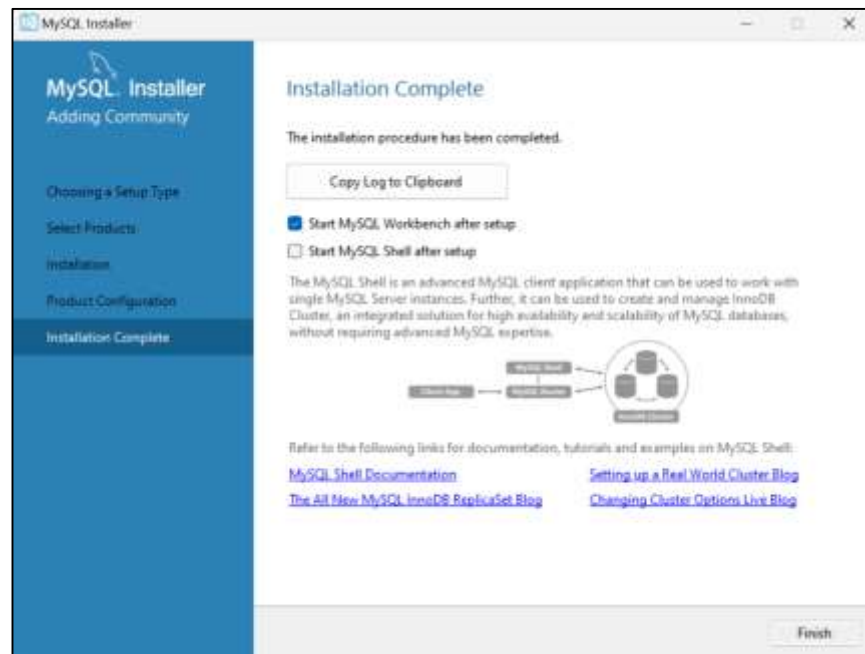
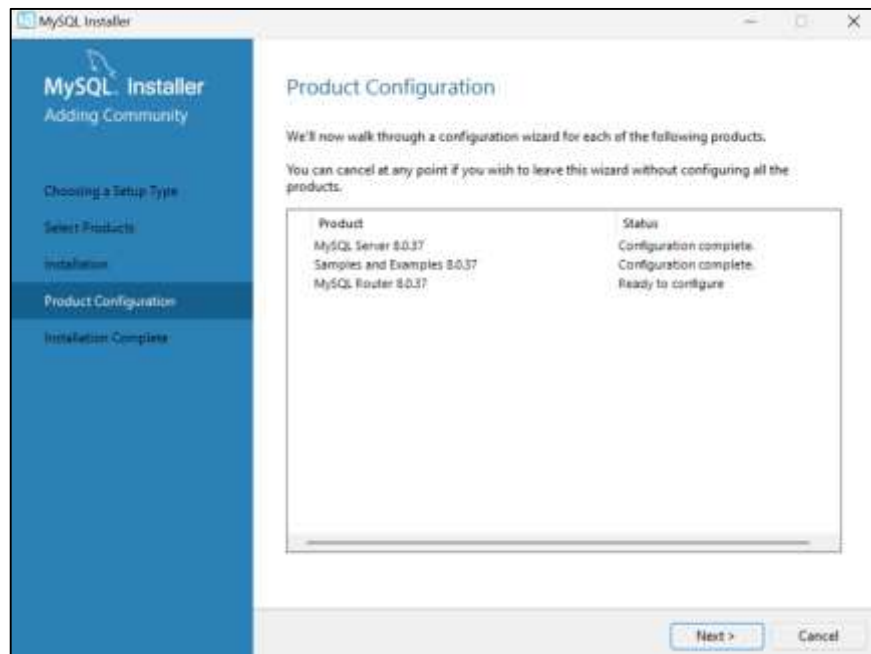
# Mysql 설치



# Mysql 설치



# Mysql 설치



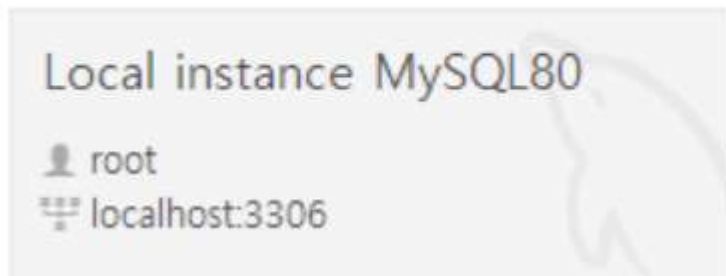
# MySQL 설치 완료



# Mysql 서버 접속

**MySQL Connections** ⊕ ⊖

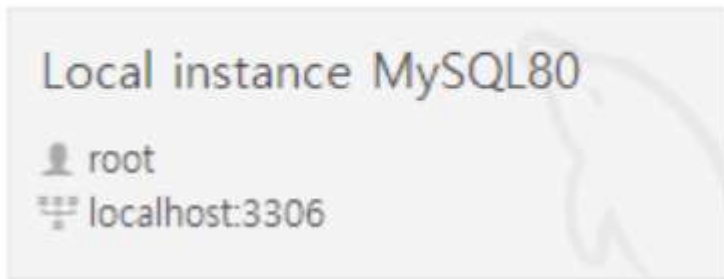
서버로의 접속을 관리



# Mysql 서버 접속

MySQL Connections  

클릭 -> 하나의 접속을 설정하는 창이 뜬다



# Mysql 서버 접속

Setup New Connection

Connection Name: fininsightSql **접속 이름 설정** Type a name for the connection

Connection Method: Standard (TCP/IP) Method to use to connect to the RDBMS

Parameters SSL Advanced

**Hostname:** 127.0.0.1 **어디서 db에 접속하는지** Port: 3306 Name or IP address of the server host - and TCP/IP port.

Username: root Name of the user to connect with.

Password: Store in Vault ... Clear The user's password. Will be requested later if it's not set.

Default Schema: The schema to use as default schema. Leave blank to select it later.

Configure Server Management... Test Connection Cancel **OK**



# Mysql 서버 접속

---

## ✓ 호스트

- 네트워크에서의 호스트는 인터넷에 연결된 특정 컴퓨터나 장치를 의미
- DB 시스템에서는 어디서 DB에 접속하는지를 의미
  - localhost : 서버가 있는 컴퓨터에서만(자기자신만) 접속 가능

속 가능

- 특정 ip : 해당 ip에서만 접속 가능
- % : 어떤 장치 에서든지 접속 가능

# Mysql 서버 접속

## MySQL Connections ⊕ ⊖

Local instance MySQL80

root

localhost:3306

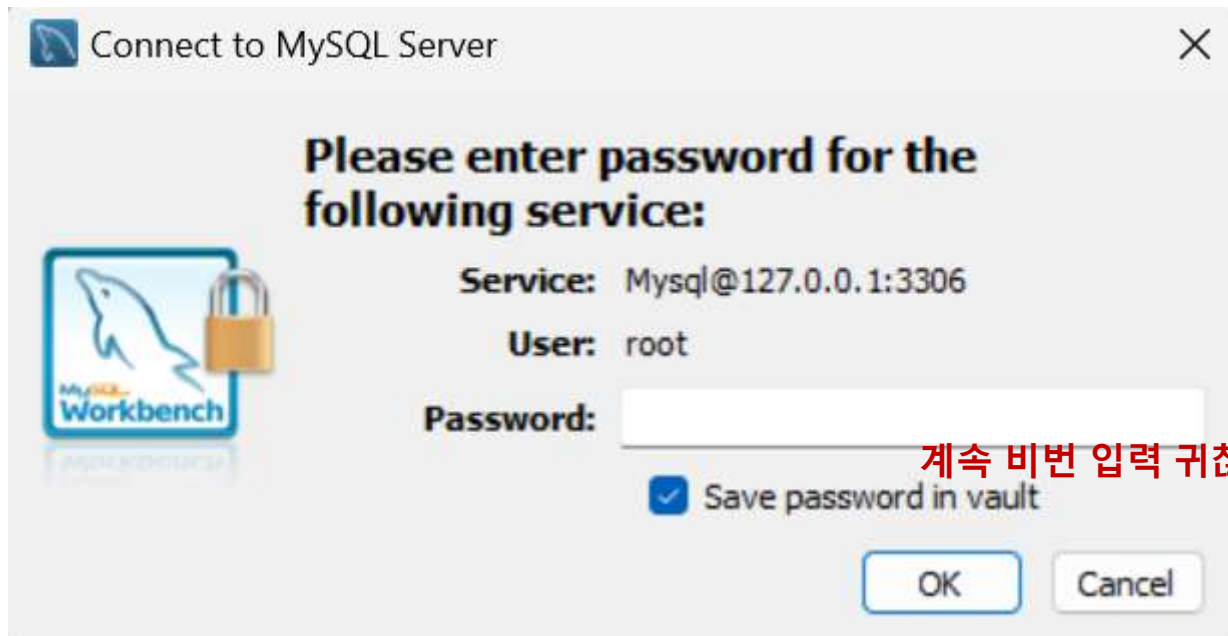
fininsightSql

root

127.0.0.1:3306

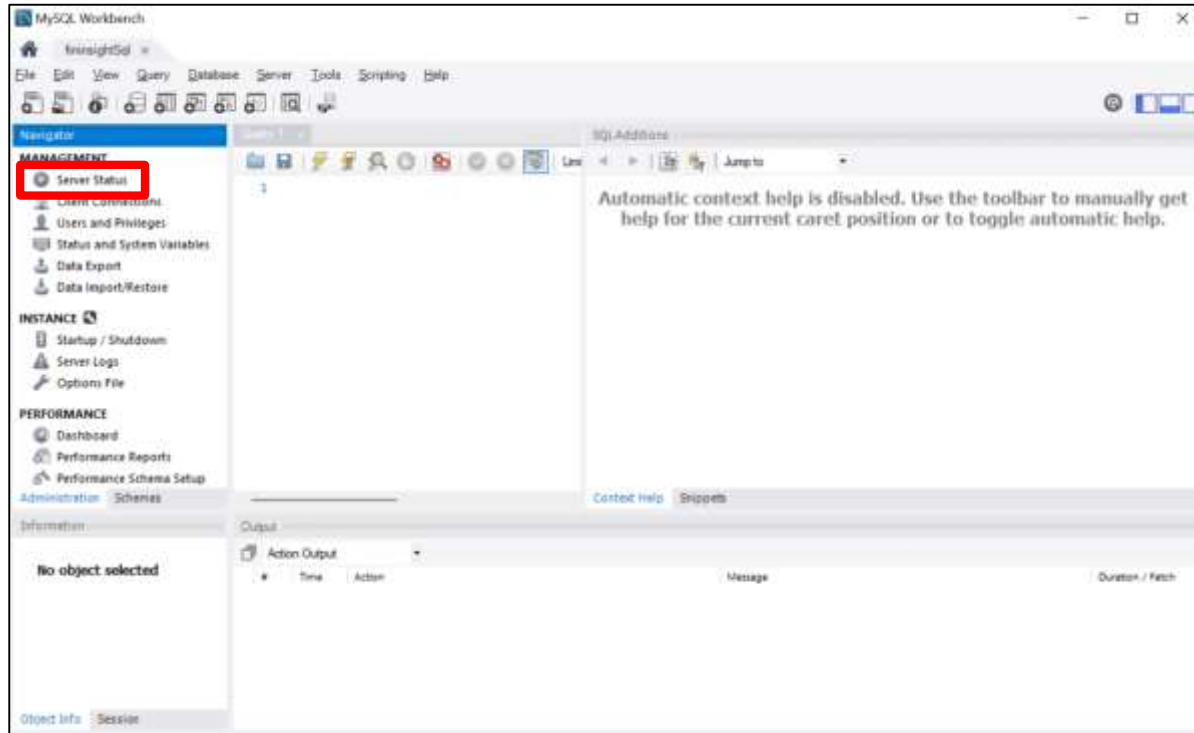
새로 커넥션 생성된 거 확인

# Mysql 서버 접속

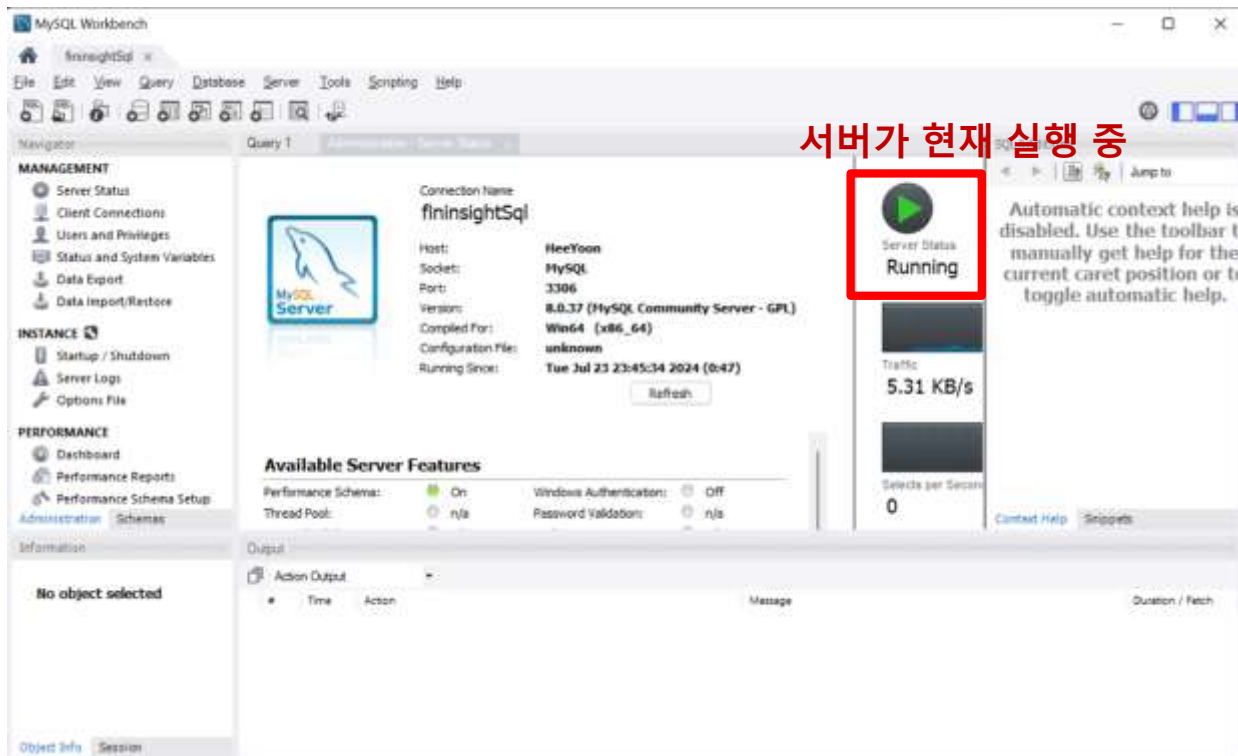


계속 비번 입력 귀찮으면 체크

# Mysql 서버 접속



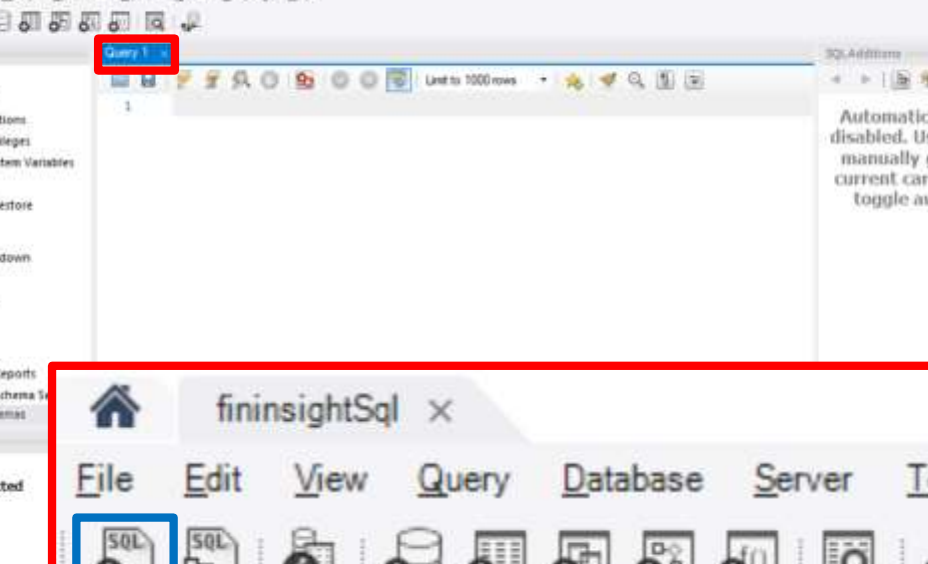
# Mysql 서버 접속



서버가 현재 실행 중

© 2007 The Authors  
Journal compilation © 2007 Blackwell Publishing Ltd

## 하는 쿼리창 추가 생성 가능



The screenshot displays two windows from Microsoft SQL Server. The top window is SQL Server Enterprise Manager, showing the 'Query 1' window highlighted. The bottom window is SQL Server Enterprise Studio, showing the 'Query' menu and the 'SQL' icon highlighted.

# 데이터베이스 생성

김지성 강사

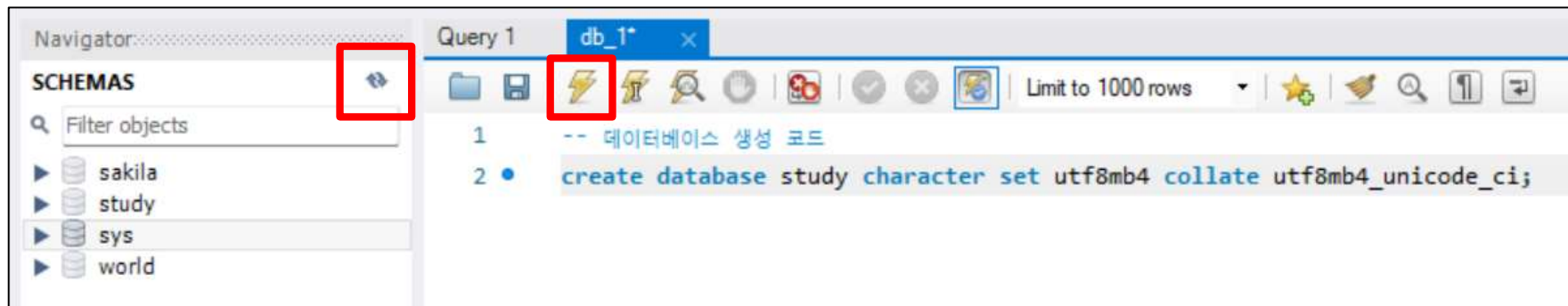
# 데이터베이스 생성

---

```
CREATE DATABASE DB명 CHARACTER SET utf8mb4 COLLATE utf8mb4_Unicode_ci;
```



# 데이터베이스 생성 실습



데이터베이스 생성 명령어 실행 후 스키마를 새로고침 하주세요

# 사용자 생성

- ✓ 관리자 계정 외 사용자 생성

**CREATE USER '사용자 명' @ '호스트 명' IDENTIFIED BY '사용자 패스워드' ;**



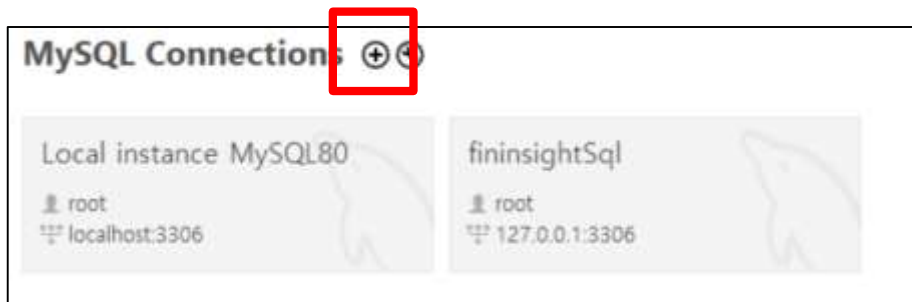
- **IP (아이피)** : 해당 IP에서만 접속 가능
- **localhost** : 로컬(서버가 있는 컴퓨터)에서만 접속 가능
- **%** : 외부 어디서나 접속 가능

# 사용자 생성 실습

## 1. 사용자 생성 명령어 실행

```
4      -- 사용자 생성 (사용자 명 : study, 호스트 : 외부 접속 가능, 비밀번호 : study)
5 •    create user 'study'@'%' identified by 'study';
```

## 2. 커넥션 생성



# 사용자 생성 실습

## 3. Connection Name, Username란에 아래와 같이 작성후 Test Connection 해보기

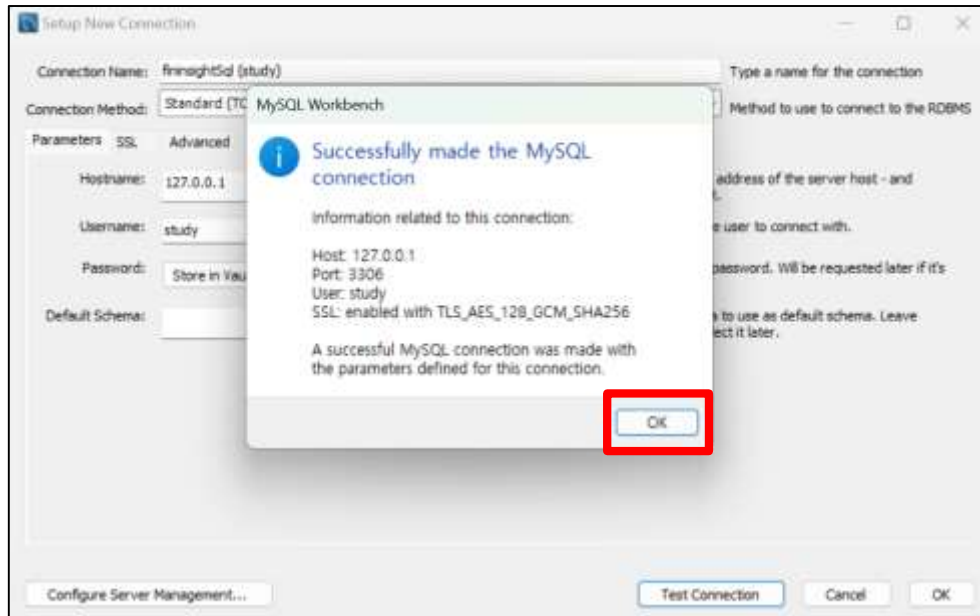
The screenshot shows the 'Setup New Connection' dialog box with the following fields and values:

- Connection Name:** fninsightSql (study)
- Connection Method:** Standard (TCP/IP)
- Parameters:**
  - Hostname:** 127.0.0.1
  - Port:** 3306
  - Username:** study
  - Password:** (with 'Store in Vault...' and 'Clear' buttons)
  - Default Schema:** (empty)

The 'Test Connection' button at the bottom right is highlighted with a red box.

# 사용자 생성 실습

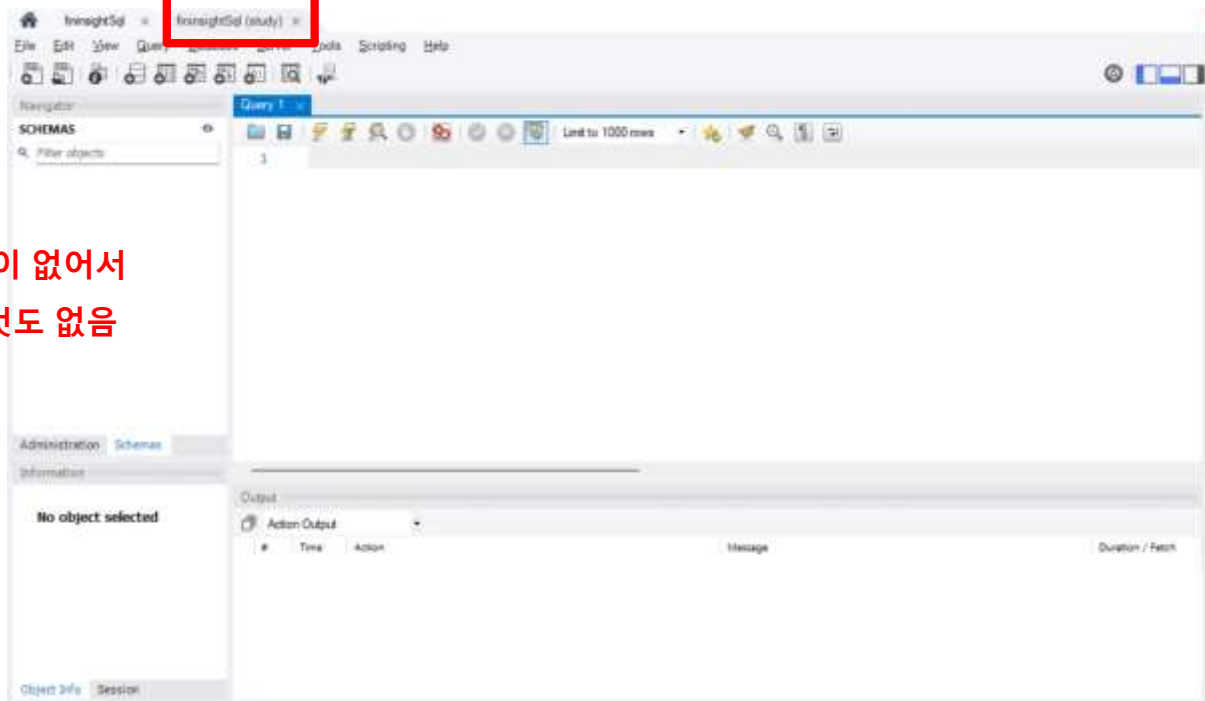
## 4. 아래와 같이 창이 뜨면 OK



# 사용자 생성 실습

## 4. study커넥션으로 서버에 접속해보기

아직 아무 권한이 없어서  
스키마에 아무것도 없음



## 권한 부여

**GRANT** [권한부여1, 권한부여2, ...] **ON** [DB명].[TABLE명] **TO** [유저명]@[호스트주소];

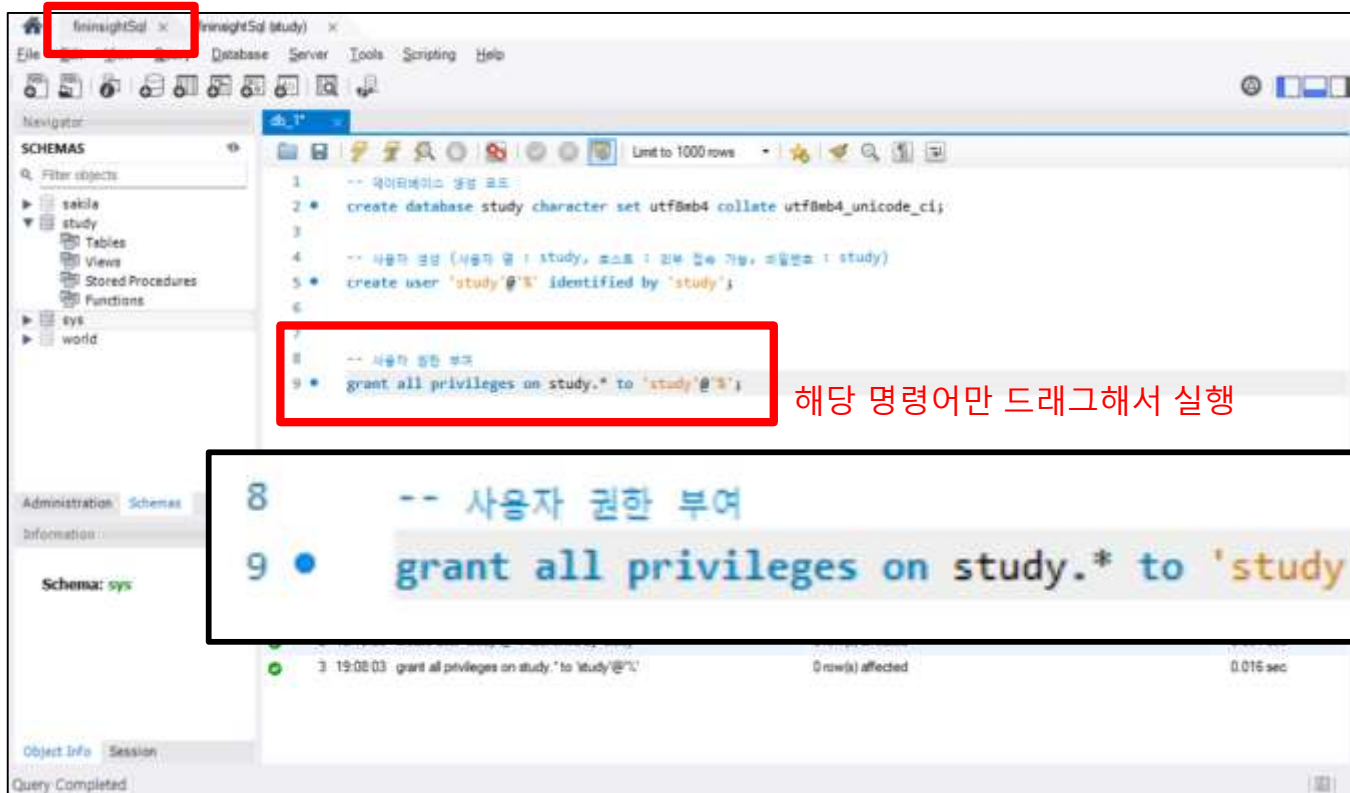
DB뿐만 아니라 테이블 마다 권한을 달리 줄 수 있음

**GRANT** SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX, ALTER  
**ON** DB명.\* **TO** '사용자명'@'localhost';

**GRANT** ALL PRIVILEGES **ON** DB명.\* **TO** '사용자명'@'localhost';

# 권한 부여 실습

fininsightSql커넥션으로 와서 study에 권한 부여하기



The screenshot shows the fininsightSql IDE interface. The 'fininsightSql' window is active, displaying a SQL script. The script includes comments in Korean and SQL commands to create a database named 'study', create a user named 'study' with password 'study', and grant all privileges on the 'study' database to the 'study' user. The line `grant all privileges on study.* to 'study'@'%';` is highlighted with a red box. A callout box points to this line with the text '해당 명령어만 드래그해서 실행' (Drag and execute only this command). Below the script, the execution results show that the command was successful, affecting 0 rows in 0.016 seconds.

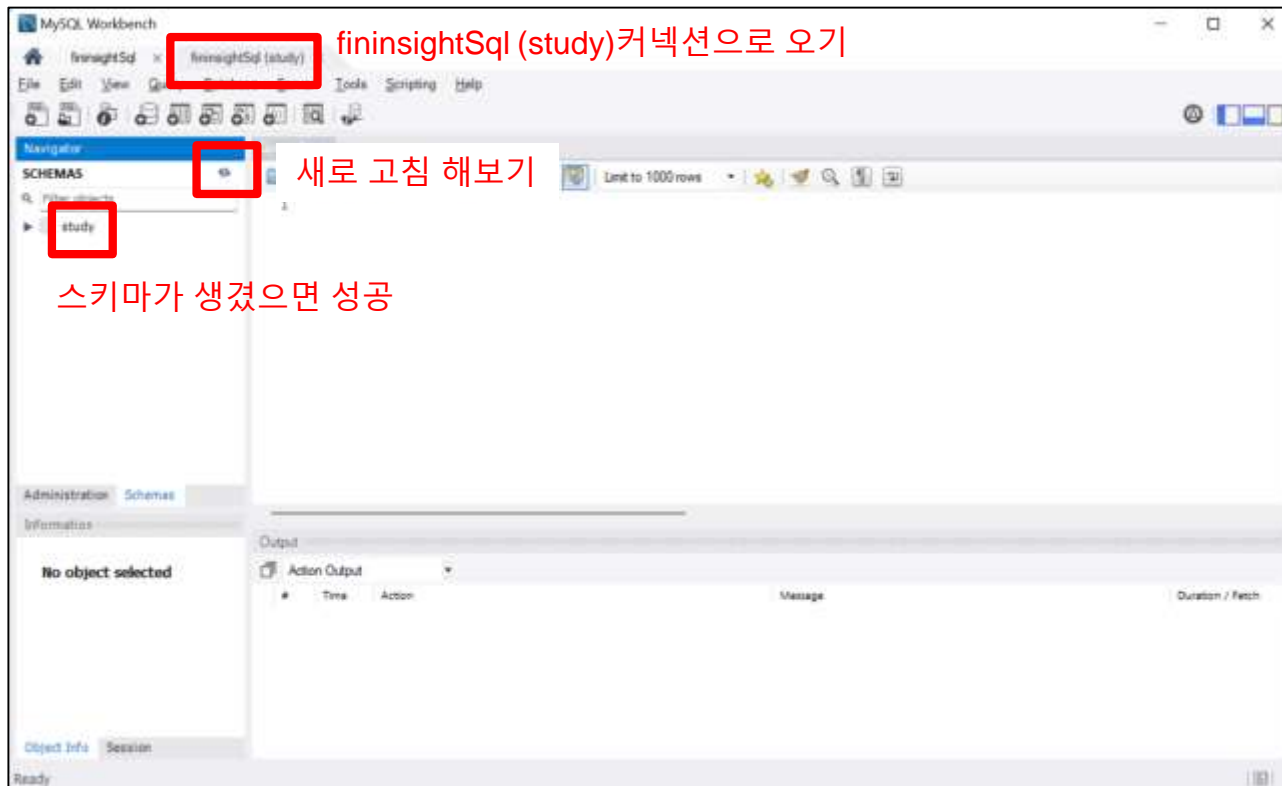
```
-- 데이터베이스 생성 코드
1 create database study character set utf8mb4 collate utf8mb4_unicode_ci;
2
3
4 -- 사용자 생성 (사용자명 : study, 비밀번호 : 권부 학습 가능, 비밀번호 : study)
5 create user 'study'@'%' identified by 'study';
6
7
8 -- 사용자 권한 부여
9 grant all privileges on study.* to 'study'@'%';
```

8 -- 사용자 권한 부여  
9 grant all privileges on study.\* to 'study'@'%';

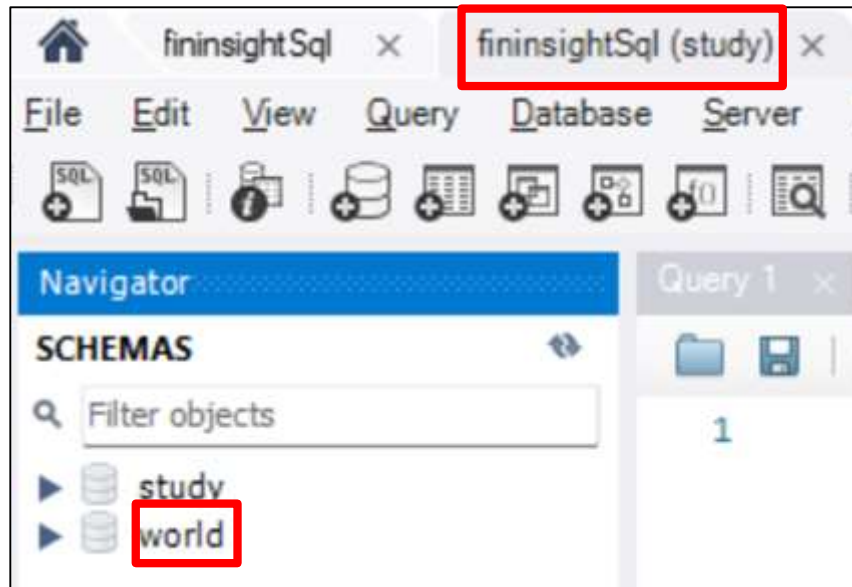
3 19:08:03 grant all privileges on study.\* to 'study'@'%' 0 row(s) affected 0.016 sec



# 권한 부여 실습



# 권한 부여 실습



Study계정에 world 데이터베이스의  
모든 테이블에 모든 권한을 부여해보세요

## 권한 관련 명령어

---

- ✓ 계정 삭제

**DROP USER** '유저명'@'호스트 주소'

- ✓ 계정 권한 조회

**SHOW GRANTS FOR** '유저명'@'호스트 주소'

- ✓ 계정 권한 삭제

**REVOKE** 권한1, 권한2, ... **ON** DB명.테이블명 **FROM** '유저명'@'호스트주소'

- ✓ 계정 모든 권한 삭제

**REVOKE** ALL **ON** \*.\* **FROM** '유저명'@'호스트주소'

# 권한 관련 명령어

---

✓ 유저 및 권한 조회

```
SELECT GRANTEE, PRIVILEGE_TYPE, IS_GRANTABLE  
FROM INFORMATION_SCHEMA.USER_PRIVILEGES;
```

✓ 실습

1. study계정의 권한 조회
2. world db에 대한 모든 권한 삭제
3. study에 다시 world db에 대한 모든 권한 부여

# 데이터베이스 테이블 생성

김지성 강사

# CREATE TABLE

## ✓ Database내 테이블 생성

**CREATE TABLE** *table\_name*

(

*column1*

*datatype,*

*column2*

*datatype,*

*column3*

*datatype,*

....

);



**CREATE TABLE** Persons (

PersonID int,

LastName

varchar(255),

FirstName

varchar(255),

Address varchar(255),

City varchar(255)

);

# CREATE TABLE

## ✓ Database내 테이블 생성

```
CREATE TABLE table_name (  
    column1 datatype NOT NULL  
    AUTO_INCREMENT,  
    column2 datatype,  
    column3 datatype,  
    PRIMARY KEY(column1)  
);
```

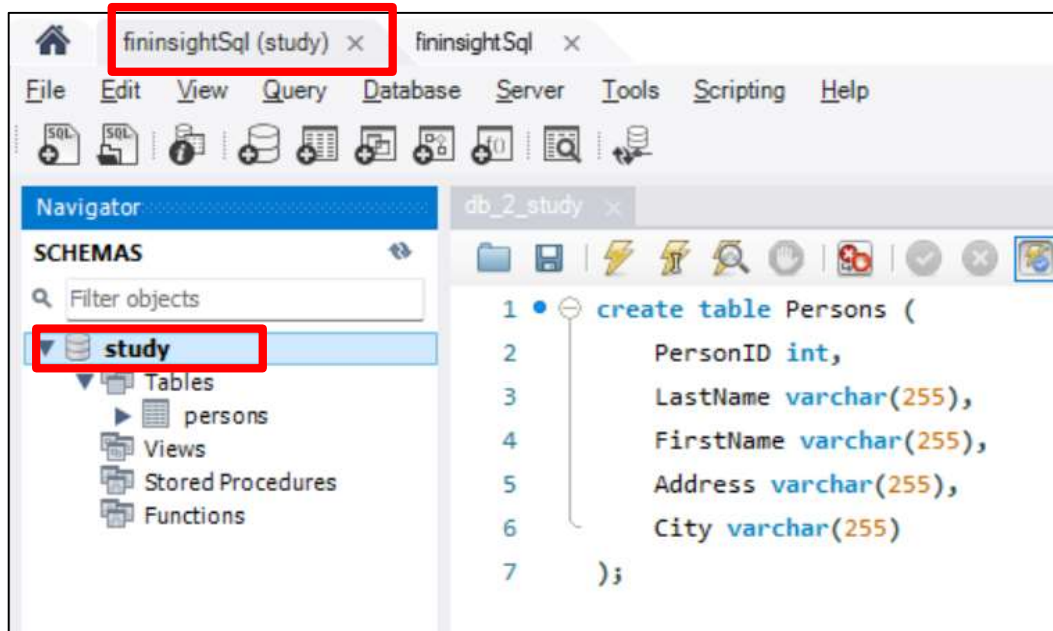
```
CREATE TABLE Persons (  
    SEQ int NOT NULL  
    AUTO_INCREMENT,  
    LastName varchar(255),  
    FirstName varchar(255),  
    Address varchar(255),  
    City varchar(255),  
    PRIMARY KEY(SEQ)  
);
```

- NOT NULL : 빈 값(null)은 올 수 없음
- AUTO\_INCREMENT : primary key 칼럼에서 사용, int형 일 때만 사용 가능, 자동으로 숫자가 0부터 순차적으로 생성 됨
- PRIMARY KEY(column1) : 해당 칼럼을 고유값 / 유일한 값으로 지정, 데이터가 많을 경우 원하는 데이터를 찾기 수월해짐

# CREATE TABLE 실습1

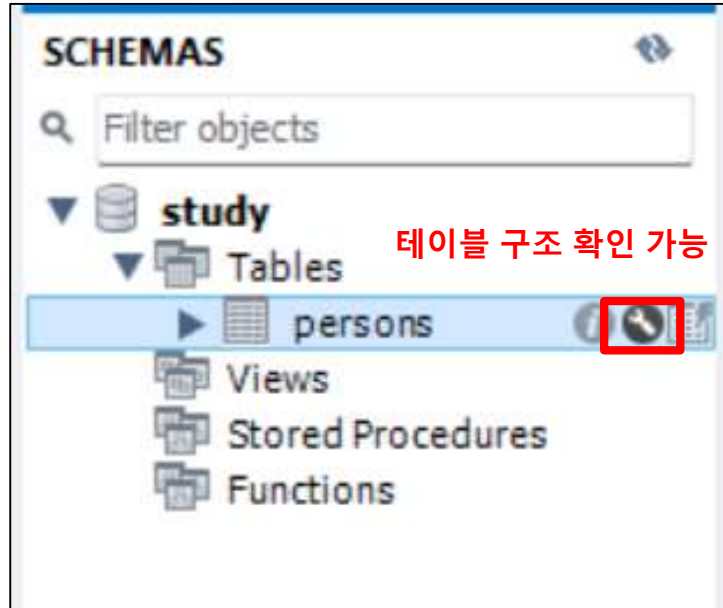
fininsightSql (study)에서 진행하기

Table study 더블클릭한 후  
쿼리 실행하기





# CREATE TABLE 실습1



Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
PersonID	INT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
LastName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
FirstName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
Address	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
City	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

Column Name:  Data Type:   
Charset/Collation:  Default:   
Comments:   
Storage: ☐ Virtual ☐ Stored  
☐ Primary Key ☐ Not Null ☐ Unique  
☐ Binary ☐ Unsigned ☐ Zero Fill  
☐ Auto Increment ☐ Generated

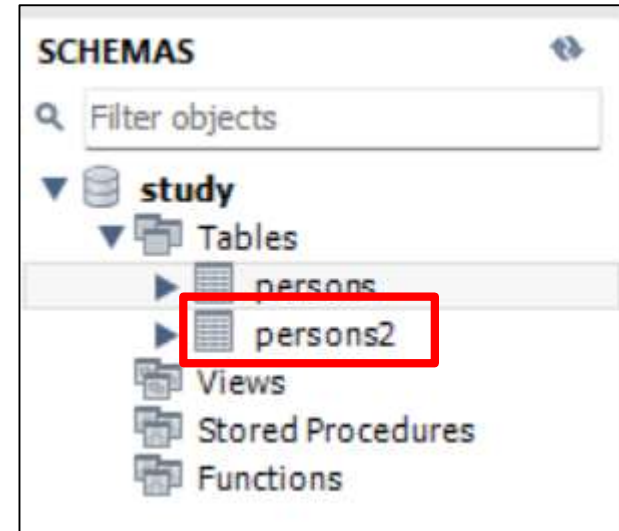
Columns | Indexes | Foreign Keys | Triggers | Partitioning | Options

Apply Revert

# CREATE TABLE 실습2

명령어 실행 후 스키마 새로고침 해보기

```
-- 실습2
create table Persons2(
    SEQ int not null auto_increment,
    LastName varchar(255),
    FristName varchar(255),
    Address varchar(255),
    City varchar(255),
    primary key(SEQ)
```



# CREATE TABLE 실습2

Table Name: persons2 Schema: study

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
SEQ	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
LastName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
FirstName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
Address	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
City	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

Column Name: Data Type: Charset/Collation: Default: Storage: ☐ Virtual ☐ Stored ☐ Primary Key ☐ Not Null ☐ Unique ☐ Binary ☐ Unsigned ☐ Zero Fill ☐ Auto Increment ☐ Generated

Columns | Indexes | Foreign Keys | Triggers | Partitioning | Options

Apply Revert

SEQ 칼럼에 pk와 not null처리가  
돼있는 거 확인하기

# 데이터 타입 (숫자형)

TYPE	사용되는 바이트	최소값 (signed/unsigned)	최대값 (signed/unsigned)
TINYINT	1	-128 0	127 255
SMALLINT	2	-32768 0	32767 65535
MEDIUMINT	3	-8388608 0	8388607 16777215
INT or INTEGER	4	-2147483648 0	2147483647 4294967295
BIGINT	8	-9223372036854775808 0	9223372036854775807 18446744073709551615
FLOAT	4	-3.40E+45 (no unsigned)	3.40E+45 (no unsigned)
DUBLE or REAL	8	-1.7976E+320 (no unsigned)	1.7976E+320 (no unsigned)

# 데이터 타입 (문자형)

TYPE	표현범위(문자 개수)	설명
CHAR(N)	정확히 n ( $\leq 255$ )	무조건 크기만큼 메모리사용
VARCHAR(N)	최대 n ( $\leq 65535$ )	안에 데이터만큼 메모리사용
TINYTEXT(N)	최대 n ( $\leq 255$ )	문자열
TEXT(N)	최대 n ( $\leq 65535$ )	문자열
MEDIUMTEXT(N)	최대 n ( $\leq 16777215$ )	문자열
LONGTEXT(N)	최대 n ( $\leq 4294967295$ )	문자열

# 데이터 타입 (바이너리)

파일을 저장할 수 있는 데이터 타입

TYPE	표현범위 (바이트)	예제
TINYBLOB(N) (Binary Large Object)	최대 n ( $\leq 255$ ) (255바이트 까지의 바이너리 저장)	바이너리
BLOB(N)	최대 n ( $\leq 65535$ )	바이너리
MEDIUMBLOB(N)	최대 n ( $\leq 16777215$ )	바이너리
LOB(N)	최대 n ( $\leq 4294967295$ )	바이너리

# 데이터 타입 (날짜형)

TYPE	표현범위	예제
DATE	1000-01-01 ~ 9999-12-31 (3바이트의 저장공간 사용)	YYYY-MM-DD
DATETIME	1000-01-01 00:00:00 ~ 9999-12-31 23:59:59	YYYY-MM-DD HH:MM:SS 문자로저장
TIMESTAMP	1970-01-01 ~ 2037-01-01 임의시 간	YYYY-MM-DD HH:MM:SS 숫자로 저장, 타임존
TIME	-838:59:59 ~ 838:59:59	
YEAR	901~2155	

# DROP TABLE

---

- ✓ 데이터베이스 내 테이블 삭제

**DROP TABLE** *table\_name*;

- ✓ 테이블 내 모든 내용 삭제

**TRUNCUATE TABLE** *table\_name*;



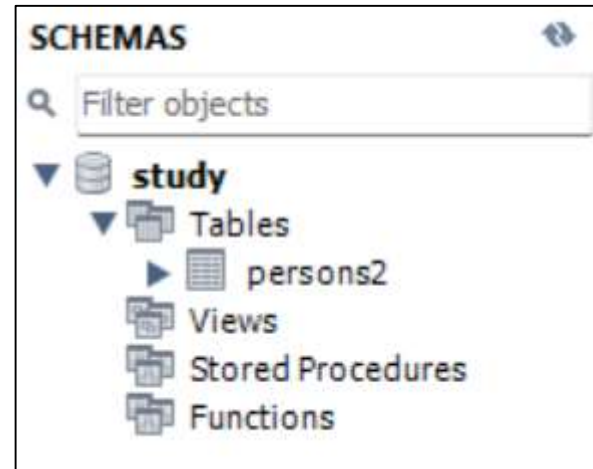
# DROP TABLE 실습

✓ 데이터베이스 내 테이블 삭제

**DROP TABLE** *table\_name*;

```
22      -- 테이블 삭제 실습
23 ●    drop table Persons;
```

Persons테이블 삭제



# ALTER TABLE

---

- ✓ 데이터베이스 내 테이블 수정 (칼럼 추가)

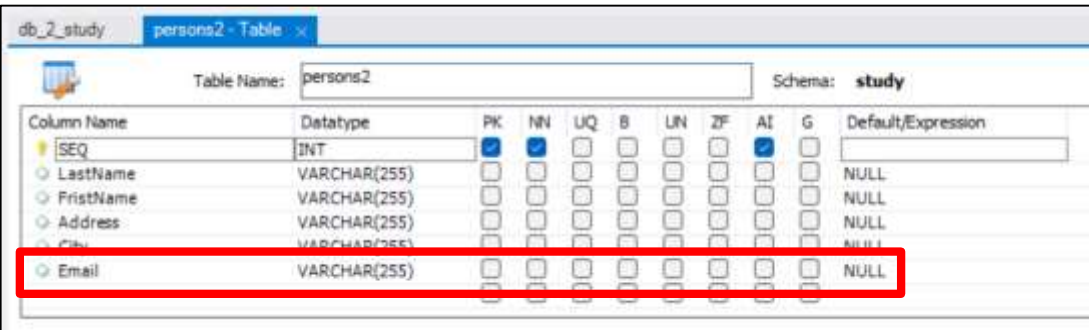
**ALTER TABLE** *table\_name* **ADD** *column\_name datatype*;

# ALTER TABLE 실습

- ✓ 데이터베이스 내 테이블 수정 (칼럼 추가) 실습

**ALTER TABLE** customers **ADD Email** varchar(255);

```
25      -- 테이블 수정 실습
26 •    alter table persons2 add Email varchar(255);
```



db\_2\_study / persons2 - Table

Table Name: persons2 Schema: study

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
SEQ	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
LastName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
FirstName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
Address	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
City	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
Email	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

# Workbench로 테이블 수정해보기1

✓ 테이블 이름 변경

db\_2\_study   persons2 - Table ×   persons2에서 person으로 변경하기

Table Name:    Schema: **study**

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
SEQ	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
LastName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
FristName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
Address	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
City	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
Email	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

# Workbench로 테이블 수정해보기2

✓ Email칼럼 없애길

db\_2\_study persons2 - Table x

Table Name:  Schema: **study**

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
SEQ	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
LastName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
FristName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
Address	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
City	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
Email	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

우클릭 후 cut 하기

# Workbench로 테이블 수정해보기2

✓ Apply하기

db\_2\_study person - Table

Table Name: person Schema: study

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
SEQ	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
LastName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
FristName	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
Address	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
City	VARCHAR(255)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

Column Name: Data Type: Charset/Collation: Default Collation: Default: Comments: Storage: ☐ Virtual ☐ Stored ☐ Primary Key ☐ Not Null ☐ Unique ☐ Binary ☐ Unsigned ☐ Zero Fill ☐ Auto Increment ☐ Generated

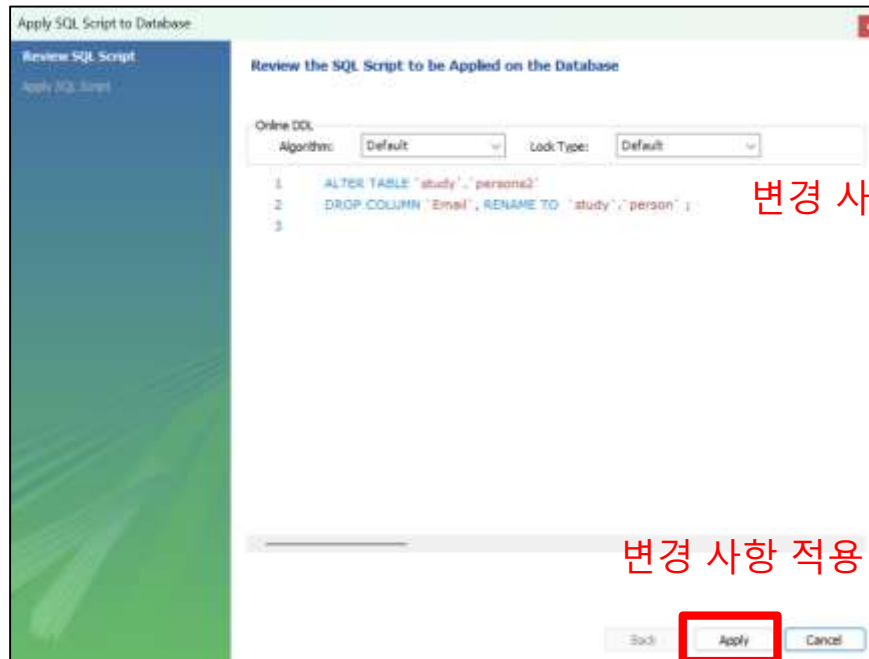
Columns Indexes Foreign Keys Triggers Partitioning Options

변경 사항 적용

Apply Revert

# Workbench로 테이블 수정해보기2

## ✓ Apply하기



변경 사항 확인 꼭 하기

변경 사항 적용

# INSERT INTO

---

- ✓ 컬럼 지정해서 데이터 입력

```
INSERT INTO table_name (column1, column2, column3 ...)  
VALUES (value1, value2, value3...);
```

- ✓ 모든 컬럼에 데이터 입력

```
INSERT INTO table_name  
VALUES (value1, value2, value3...);
```

- ✓ 다른 테이블의 값을 그대로 넣고싶은 경우

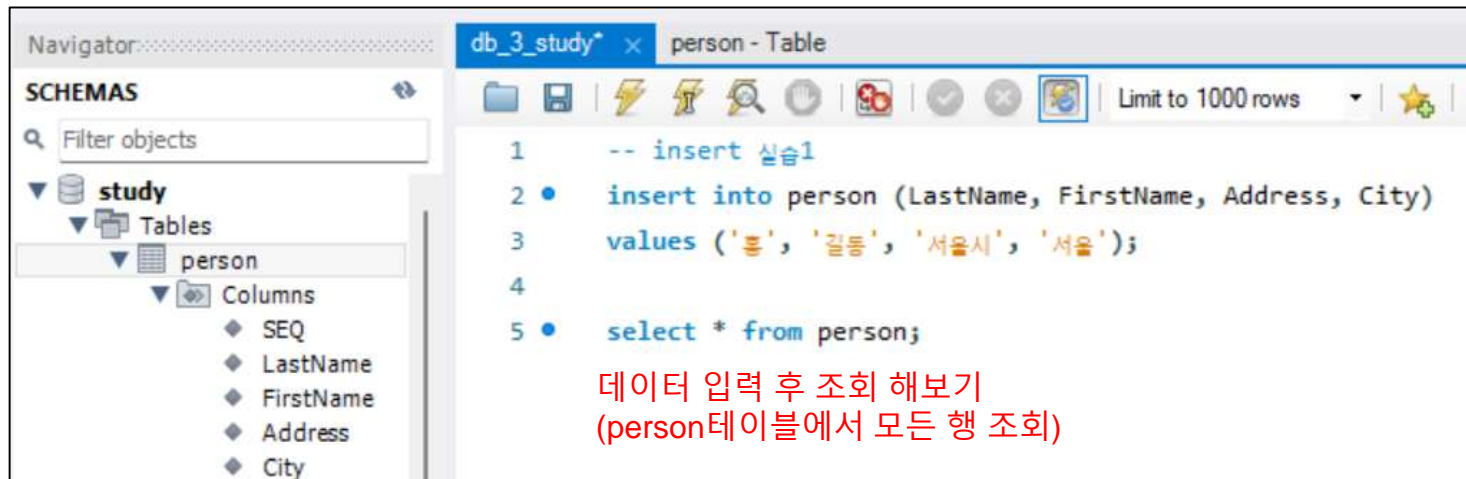
```
INSERT INTO table2 (column1, column2, column3, ..)  
SELECT column1, column2, column3, ...  
FROM table1  
WHERE condition;
```



# INSERT INTO 실습1

- ✓ 컬럼 지정해서 데이터 입력

**INSERT INTO** *table\_name* (*column1*, *column2*, *column3* ...)  
**VALUES** (*value1*, *value2*, *value3*...);



The screenshot shows a database management tool interface. On the left, the 'Navigator' pane displays the 'study' schema with a table named 'person'. The table's columns are listed: SEQ, LastName, FirstName, Address, and City. The main pane shows a SQL script for 'db\_3\_study' with the following content:

```
1  -- insert 실습1
2  • insert into person (LastName, FirstName, Address, City)
3  values ('홍', '길동', '서울시', '서울');
4
5  • select * from person;
```

Below the script, red text indicates the next step: "데이터 입력 후 조회 해보기 (person테이블에서 모든 행 조회)".

# INSERT INTO 실습1

✓ 결과

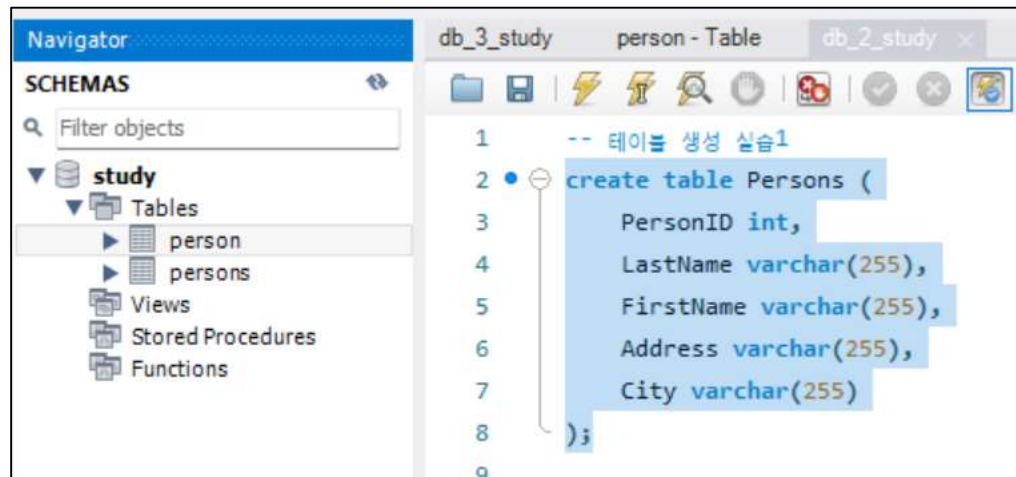
	SEQ	LastName	FirstName	Address	City
▶	1	홍	길동	서울시	서울
★	NULL	NULL	NULL	NULL	NULL

# INSERT INTO 실습2

- ✓ 모든 컬럼에 데이터 입력

**INSERT INTO** *table\_name* **VALUES** (*value1, value2, value3...*);

1. db\_2\_study.sql에서 persons테이블 다시 create해주기



# INSERT INTO 실습2

## 2. PersonID칼럼 데이터 추가 후 모든 칼럼에 데이터 입력해주기

```
8      -- insert 실습2 (persons테이블 다시 생성 후 모든 컬럼에 데이터 삽입)
9      • insert into persons
10     values (0, '홍', '길동', '서울시', '서울');
11
12     • select * from persons;
```

Result Grid			Filter Rows: <input type="text"/>	Export:	Wrap Cell Content:
	PersonID	LastName	FirstName	Address	City
▶	0	홍	길동	서울시	서울

# INSERT INTO 실습3

- ✓ 다른 테이블의 값을 그대로 넣고싶은 경우

```
INSERT INTO table2 (column1, column2, column3, ..)  
SELECT column1, column2, column3, ...  
FROM table1  
WHERE condition;
```

```
17 • insert into person (LastName, FirstName, Address, City)  
18   select LastName, FirstName, Address, City from persons;  
19  
20 • select * from person;
```

---

	SEQ	LastName	FirstName	Address	City
▶	1	홍	길동	서울시	서울
	2	홍	길동	서울시	서울
*	NULL	NULL	NULL	NULL	NULL

# DELETE

---

- ✓ 조건에 맞는 데이터 삭제

**DELETE FROM** *table\_name* **WHERE** *condition*;

# DELETE 실습

- ✓ 조건에 맞는 데이터 삭제

**DELETE FROM** *table\_name* **WHERE** *condition*;

```
25 • select * from person where SEQ = 2;  
26 • delete from person where SEQ = 2;
```

Delete전엔 **select문으로** 해당 데이터  
**확인 및 조회** 후, 데이터가 존재하면  
Delete문 실행 해주기

결과

✓	17 01:50:46	delete from person where SEQ = 2	1 row(s) affected	0.016 sec
---	-------------	----------------------------------	-------------------	-----------

Delete실행 후 해당 명령문은 **주석처리** 해주기

# UPDATE

---

- ✓ 조건에 맞는 데이터 수정

**UPDATE** *table\_name*

**SET** *column1 = value1, column2 = value2, ...*

**WHERE** *condition;*



# UPDATE 실습

- ✓ 조건에 맞는 데이터 수정

**UPDATE** table\_name **SET** column1 = value1, column2 = value2, ... **WHERE** condition;

```
32 • update person
33   set address = '서울시 양천구', city='서울시'
34   where SEQ=1;
35
36 • select * from person;
```

Result Grid | Filter Rows: | Edit:

	SEQ	LastName	FirstName	Address	City
▶	1	홍	길동	서울시 양천구	서울시
•	NULL	NULL	NULL	NULL	NULL

# 전체 실습

---

- ✓ study db에 두 개의 테이블을 생성 후 각각 데이터 입력하기

## 테이블명 : Students

number: 0, name : 홍길동, age : 30, address : 인천광역시

number: 1, 이연걸, age : 60, address : 서울특별시

number: 2, 이몽룡, age : 42, address : 대전광역시

number: 3, 성춘향, age : 27, address : 경기도

## 테이블명 : scores

number : 0, math : 90, english : 80, science : 50

number : 1, math : 69, english : 76, science : 65

number : 2, math : 98, english : 87, science : 97

number : 3, math : 87, english : 67, science : 79

# 데이터 조화

김지성 강사

# SELECT

---

- ✓ 컬럼 지정하여 조회

```
SELECT column1, column2, column3 ...
```

```
FROM table_name;
```

- ✓ 전체 컬럼 조회

```
SELECT * FROM table_name;
```

- ✓ 지정 컬럼의 유일한 값 조회

```
SELECT DISTINCT column1, column2, column3 ...
```

```
FROM table_name;
```

# WHERE

---

- ✓ 조건에 맞는 데이터 조회

```
SELECT column1, column2, column3 ...
```

```
FROM table_name
```

```
WHERE condition;
```

- ✓ 패턴(조건)에 맞는 데이터 조회

```
SELECT column1, column2, column3 ...
```

```
FROM table_name
```

```
WHERE condition LIKE pattern;
```

# AND, OR, NOT

---

- ✓ AND 조건에 맞는 데이터 조회

```
SELECT column1, column2, column3 ...  
FROM table_name  
WHERE condition AND condition2 AND condition3 ...;
```

- ✓ OR 조건에 맞는 데이터 조회

```
SELECT column1, column2, column3 ...  
FROM table_name  
WHERE condition OR condition2 OR condition3 ...;
```

- ✓ NOT 조건에 맞는 데이터 조회

```
SELECT column1, column2, column3 ...  
FROM table_name  
WHERE NOT condition ;
```

# IN

---

- ✓ IN 조건에 맞는 데이터 조회

```
SELECT column_name(s)  
FROM table_name  
WHERE column_name IN (value1, value2, ...);
```

- ✓ IN 조건에 맞는 데이터 조회

```
SELECT column_name(s)  
FROM table_name  
WHERE column_name IN (SELECT STATEMENT);
```

# NULL

✓ 다른 자료형과는 별도로 처리

✓ 컬럼 데이터가 NULL인 데이터 조회

```
SELECT column_name(s)
```

```
FROM table_name
```

```
WHERE column_name IS NULL ;
```

 **= NULL 안 됨! IS NULL로 실행해야 함**

✓ 컬럼 데이터가 NULL이 아닌 데이터 조회

```
SELECT column_name(s)
```

```
FROM table_name
```

```
WHERE column_name IS NOT NULL ;
```



# ORDER BY

---

- ✓ 조회 시 정렬 적용 (오름차순 / 내림차순)

- ✓ 지정 컬럼을 기준으로 오름/내림차순 조회

```
SELECT column1, column2, ...
```

```
FROM table_name
```

```
ORDER BY column1, column2, ... ASC|DESC ;
```

- ✓ 컬럼 별로 오름/내림차순 조회

```
SELECT column1, column2, ...
```

```
FROM table_name
```

```
ORDER BY column1 ASC|DESC , column2 ASC|DESC, ... ;
```

# COUNT, AVG, SUM

---

- ✓ 데이터 갯수 조회

```
SELECT COUNT (column_name)
FROM table_name
WHERE condition;
```

- ✓ 데이터 평균 값 계산

```
SELECT AVG (column_name)
FROM table_name
WHERE condition;
```

- ✓ 데이터 합산 값 계산

```
SELECT SUM (column_name)
FROM table_name
WHERE condition;
```

# MIN, MAX

---

✓ 최대값 조회

```
SELECT MIN(column_name)
FROM table_name
WHERE condition;
```

✓ 최소값 조회

```
SELECT MAX(column_name)
FROM table_name
WHERE condition;
```

# GROUP BY

---

✓ 열 기준 데이터 크루핑

```
SELECT column_name(s)  
FROM table_name  
WHERE condition  
GROUP BY column_name(s)  
ORDER BY column_name(s);
```

# JOIN

이름	전화번호
홍길동	010-3242-5931
이동진	010-3943-1992
박철우	010-6123-4453

이름	구입일	상품	수량
홍길동	2019-01-02	우유	1
홍길동	2019-01-03	식빵	2
이동진	2018-12-21	치즈	1
박철우	2018-12-23	소금	1
박철우	2018-12-25	우유	3



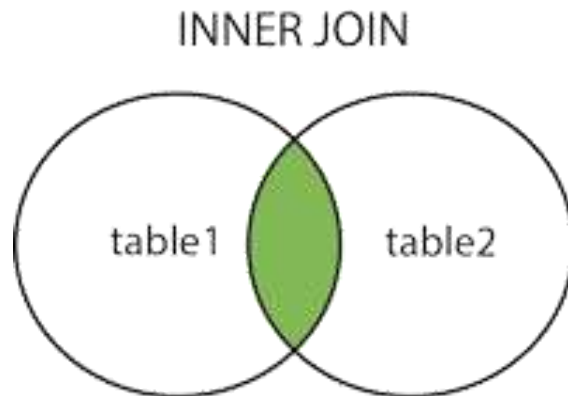
이름	전화번호	구입일	상품	수량
홍길동	010-3242-5931	2019-01-02	우유	1
홍길동	010-3242-5931	2019-01-03	식빵	2
이동진	010-3943-1992	2018-12-21	치즈	1
박철우	010-6123-4453	2018-12-23	소금	1
박철우	010-6123-4453	2018-12-25	우유	3

# INNER JOIN

---

✓ 두 테이블 간 INNER JOIN

```
SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name;
```



# LEFT JOIN

---

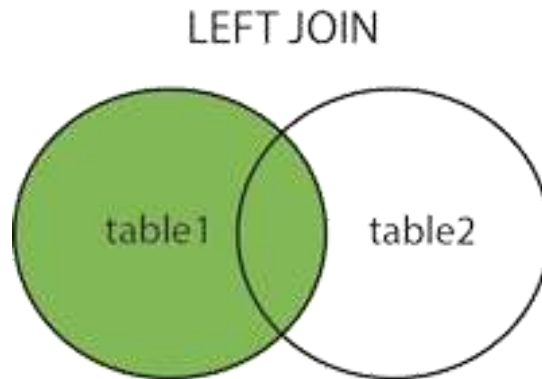
- ✓ 왼쪽에 위치한 테이블을 기준으로 데이터 조회  
(오른쪽 테이블에 NULL값이 있어도 조회 됨)

**SELECT** *column\_name(s)*

**FROM** *table1*

**LEFT JOIN** *table2*

**ON** *table1.column\_name = table2.column\_name;*

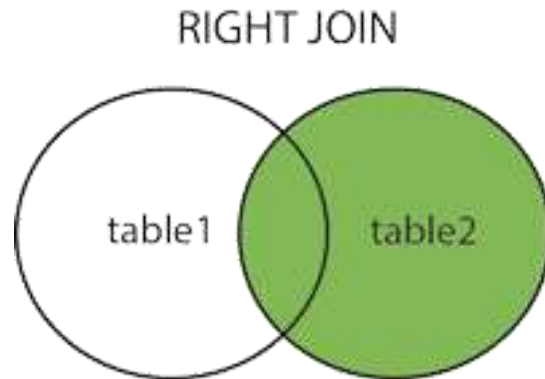


# RIGHT JOIN

---

- ✓ 오른쪽에 위치한 테이블을 기준으로 데이터 조회  
(왼쪽 테이블에 NULL값이 있어도 조회 됨)

```
SELECT column_name(s)
FROM table1
RIGHT JOIN table2
ON table1.column_name = table2.column_name;
```





# UNION

- ✓ 두 테이블 간 NULL값이 있어도 모두 조회

```
SELECT column_name(s)
FROM table1
LEFT JOIN table2
ON table1.column_name = table2.column_name
UNION
SELECT column_name(s)
FROM table1
RIGHT JOIN table2
ON table1.column_name = table2.column_name
```

Mysql에서는 FULL OUTER JOIN 대신  
**UNION**이 쓰임

FULL OUTER JOIN

