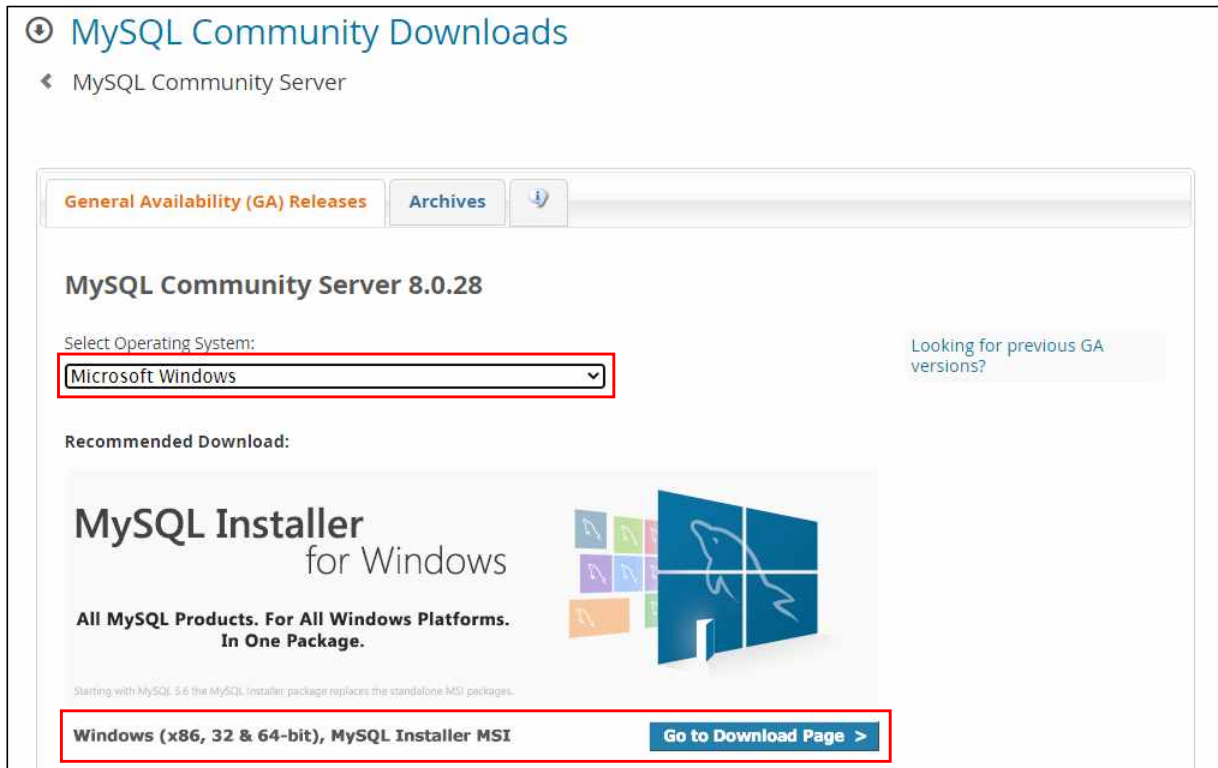


1. Windows에 MySQL 설치

1. MySQL 다운로드

1) [링크] : <https://downloads.mysql.com/archives/installer>로 접속



2) OS(Microsoft Windows) 선택 및 [Go to Download Page] 클릭

3) [Archives] 탭 클릭 후 버전 선택

4) 용량이 큰 [Windows, MSI Installer]로 다운로드

Product Version: 8.0.17

Operating System: Microsoft Windows

Windows (x86, 32-bit), MSI Installer

(mysql-installer-web-community-8.0.17.0.msi)

Jul 15, 2019

18.5M

Download

MD5: 567707887fc0d1fad7fc048a879a0da2 | Signature

Windows (x86, 32-bit), MSI Installer

(mysql-installer-community-8.0.17.0.msi)

Jul 15, 2019

393.4M

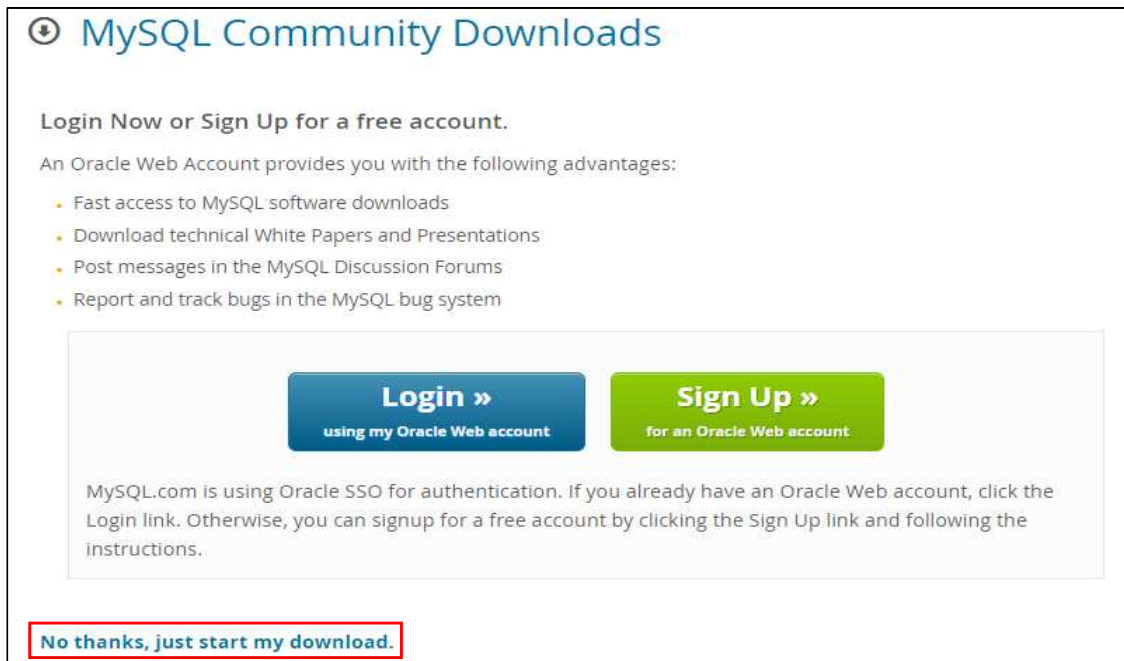
Download

MD5: 3aa8d6470fb6b58f517d3efb46e5472b | Signature

※ 수업 시 사용하는 버전과 캡처화면의 버전은 다를 수 있으므로 반드시 수업 시 안내 받은 버전으로 작업하여 주시기 바랍니다.

MySQL

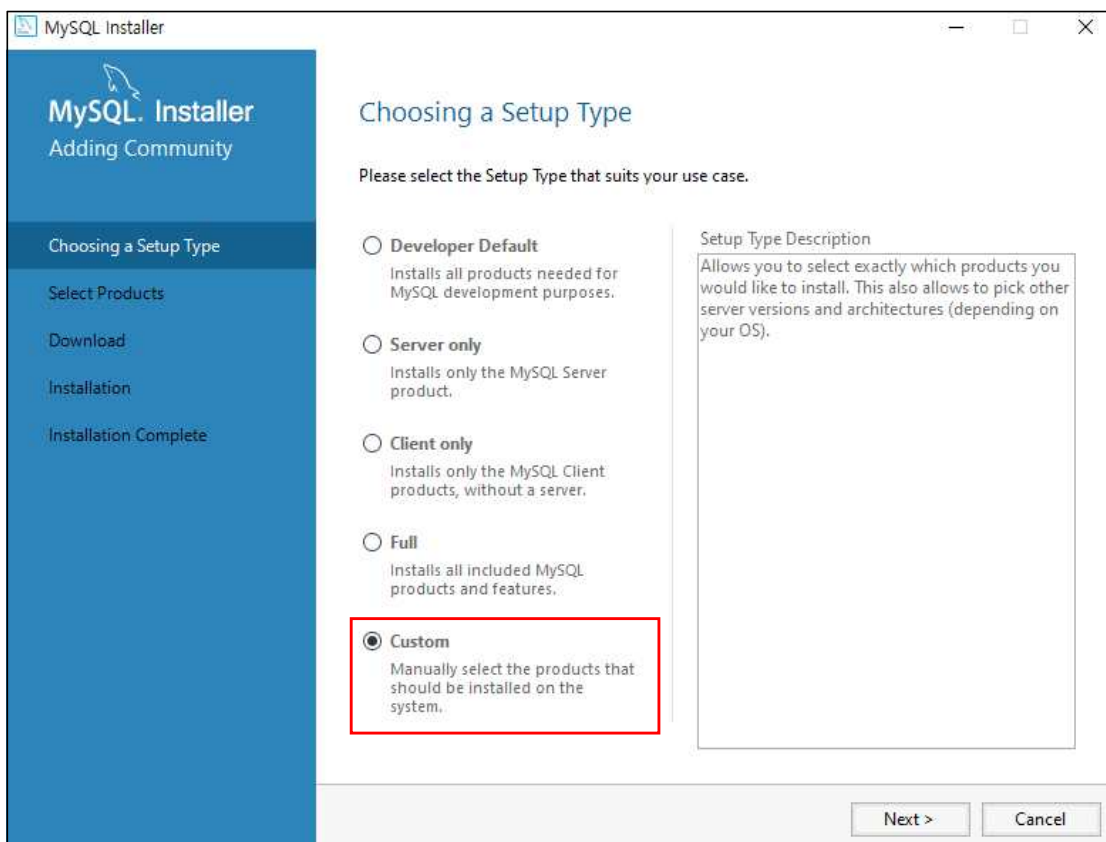
5) 로그인 없이 다운로드 가능 : [No thanks, just start my download] 클릭 후 다운로드 실시



2. MySQL 설치하기

1) [mysql-installer-community] 실행 파일 더블 클릭

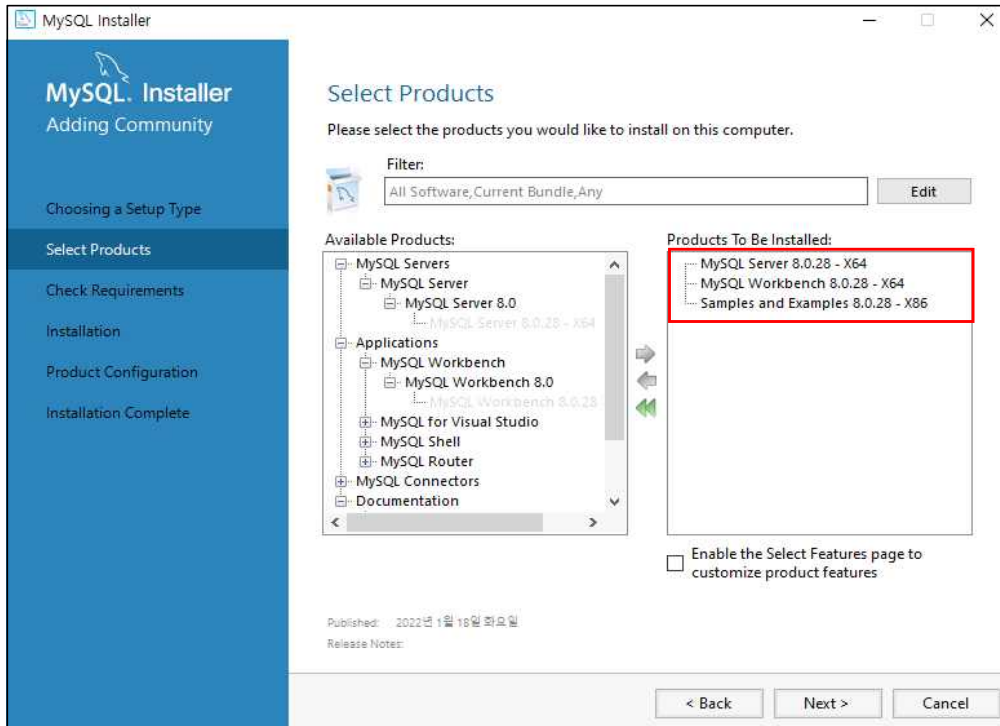
2) Choosing a Setup Type - [Custom] 선택 후 Next



MySQL

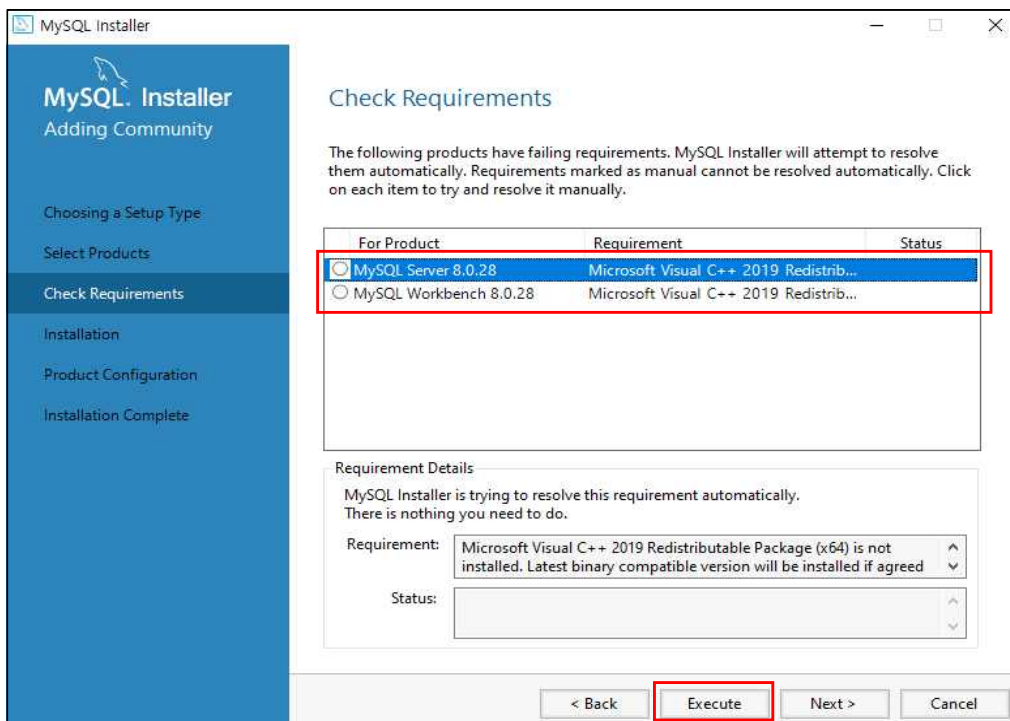
3) Select Products – 설치할 제품 선택 후 옆으로 이동(→) 후 Next

- ① MySQL Servers – MySQL Server – MySQL Server 8.0 – MySQL Server 8.0 - x64
- ② Applications – MySQL Workbench – MySQL Workbench 8.0 – MySQL Workbench 8.0 - x64
- ③ Documentation – Samples and Examples – Samples and Examples 8.0 – Samples and Examples 8.0 - x86



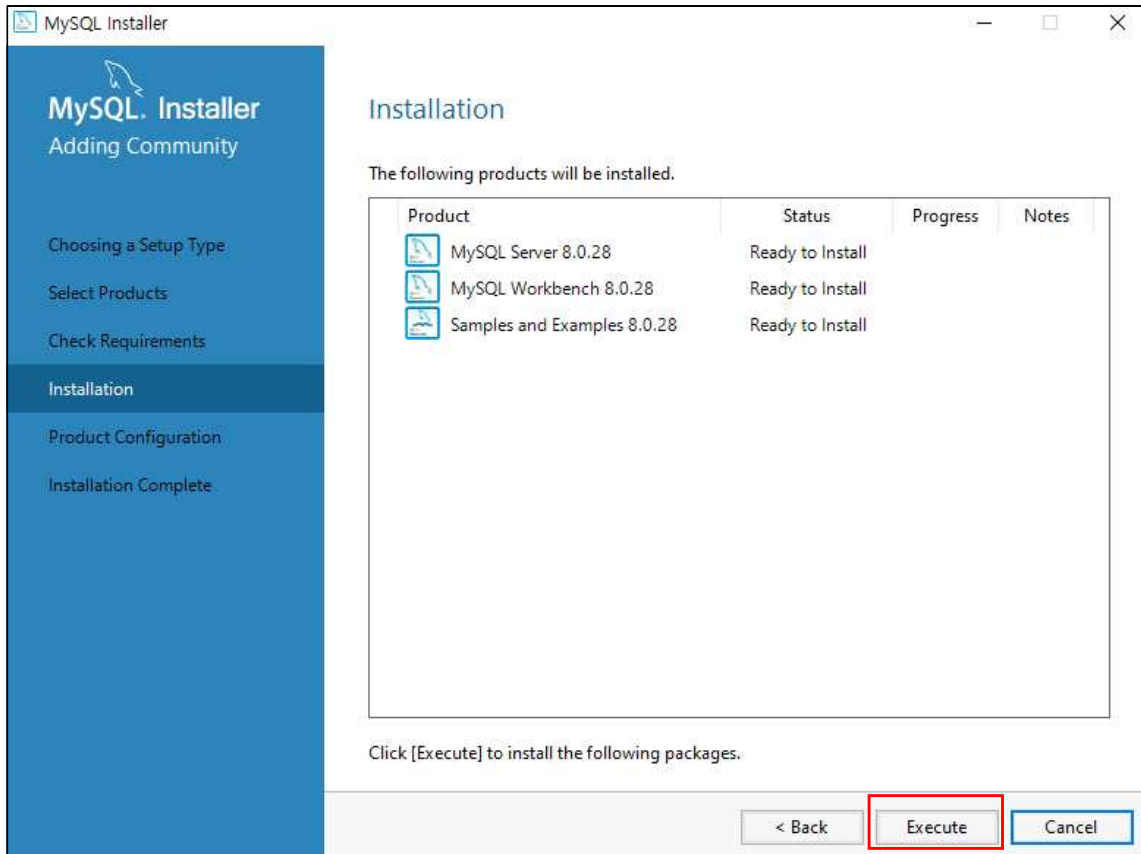
4) Check Requirements – Execute로 필요 프로그램 설치 후 Next

- 필요한 프로그램이 목록에 나타나면 선택해서 [Execute] 클릭 후 설치
- 인터넷이 연결되어 있다면 정상적으로 설치됨.

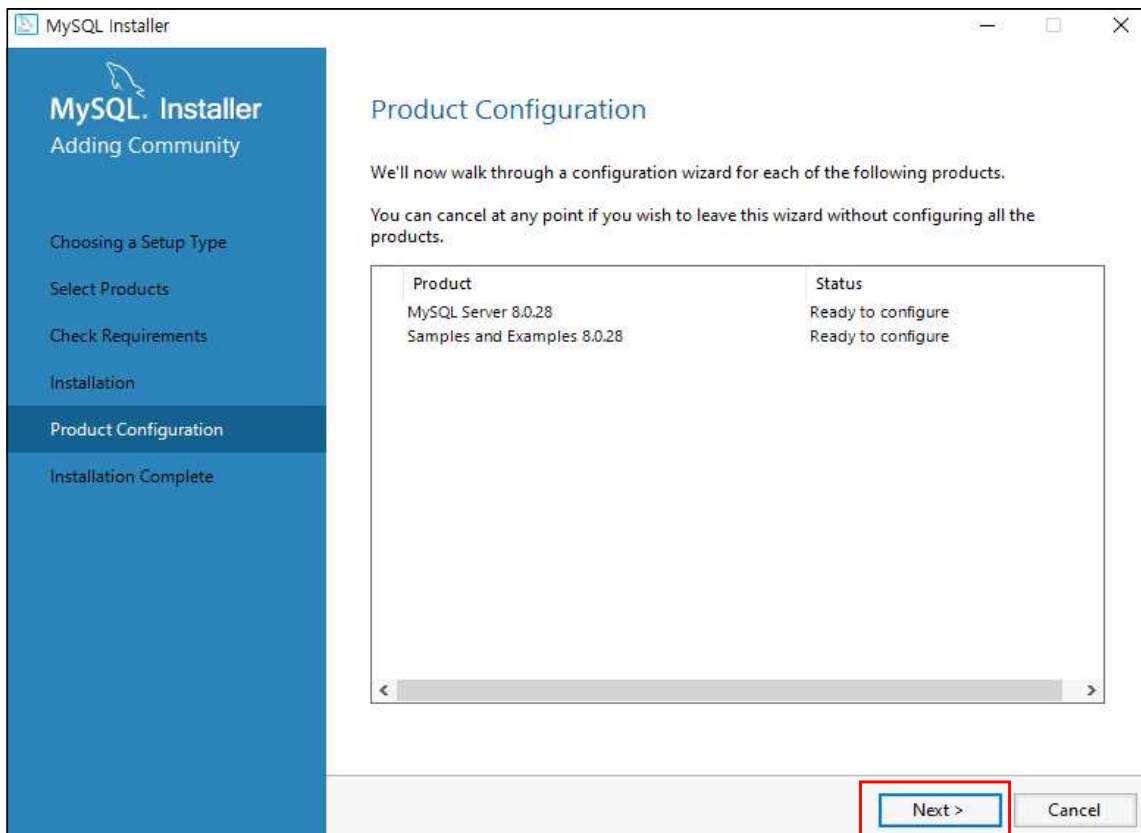


MySQL

5) Installation – 설치 프로그램 확인 및 Execute를 클릭하여 설치 후 Next



6) Product Configuration – 환경 설정이 필요한 항목 확인 후 Next



MySQL

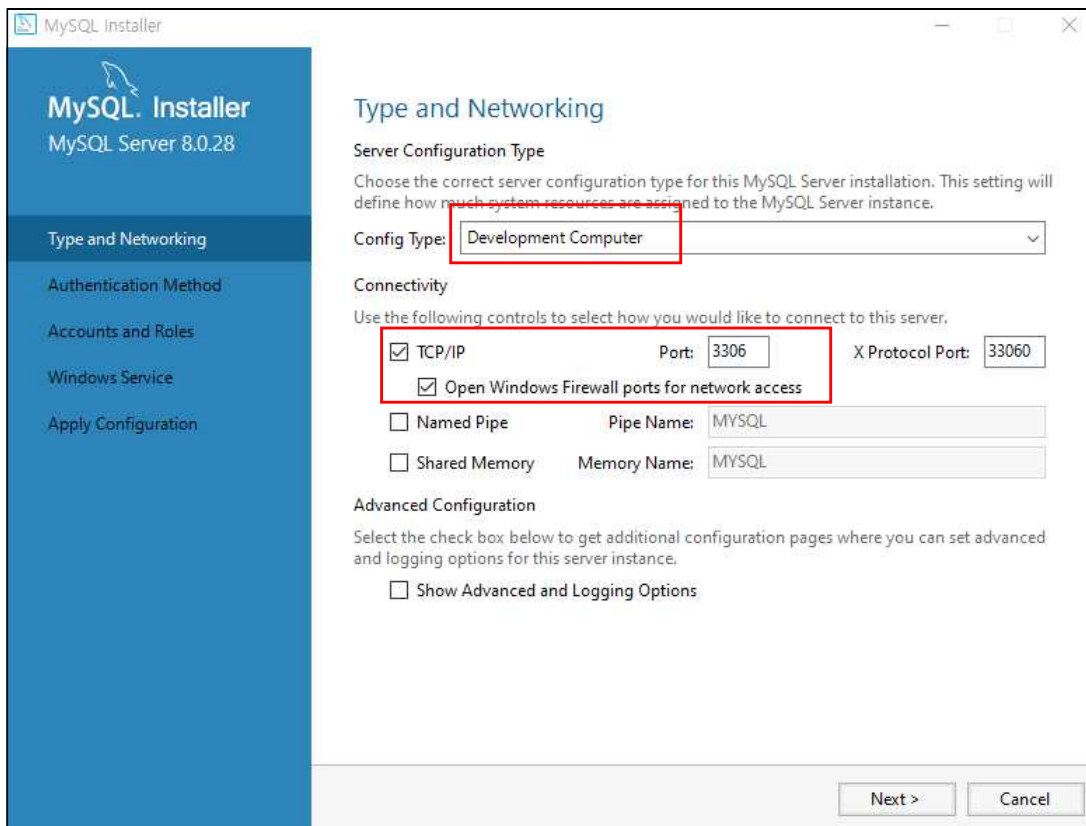
7) [환경설정1] : Type and Networking

① Config Type : Development Computer 선택

- Development Computer : MySQL 외에도 여러 프로그램을 사용하는 컴퓨터
- Server Computer : 웹 서버 등의 중요한 서버가 가동되는 컴퓨터에 MySQL을 설치할 때 선택함. MySQL이 최소한의 메모리만 사용함.
- Dedicated Computer : MySQL 전용 컴퓨터, MySQL이 사용 가능한 메모리를 최대한으로 사용하므로 성능이 좋음.

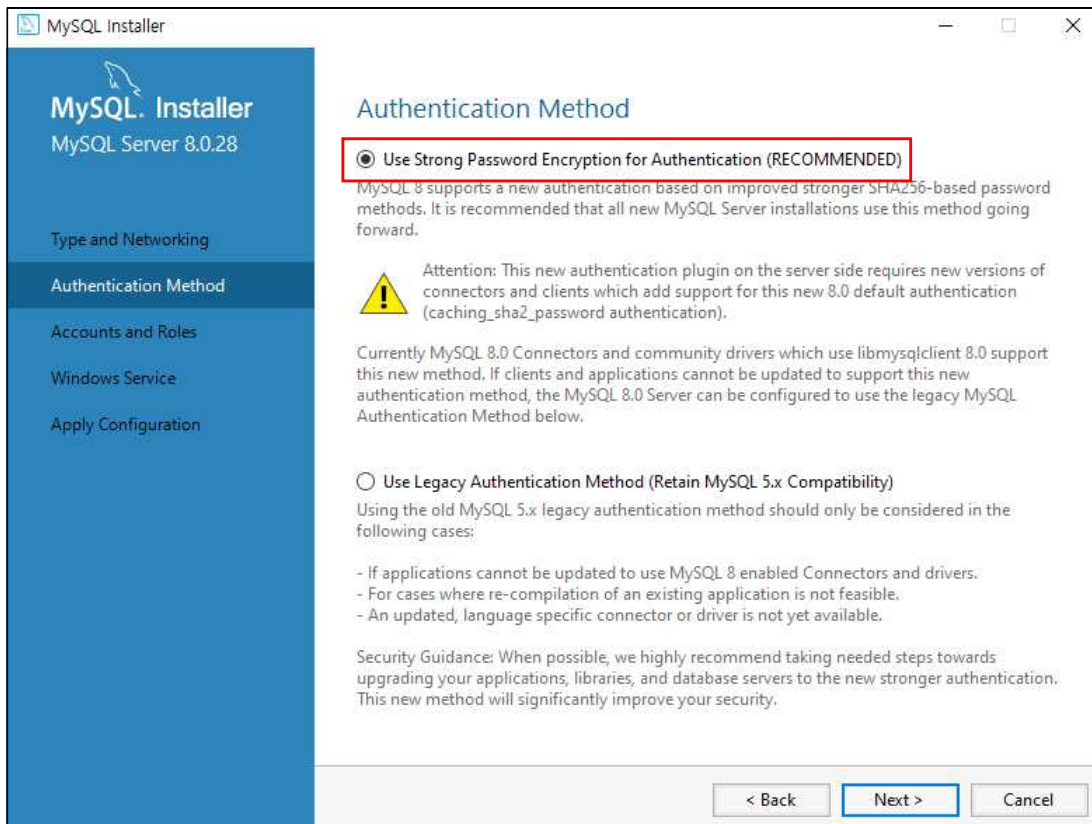
② TCP/IP 체크 및 Port번호 기억 / Open Windows Firewall ports for network access 체크

- 외부 컴퓨터에서 접근할 수 있도록 [Windows 방화벽]의 포트를 허용함.

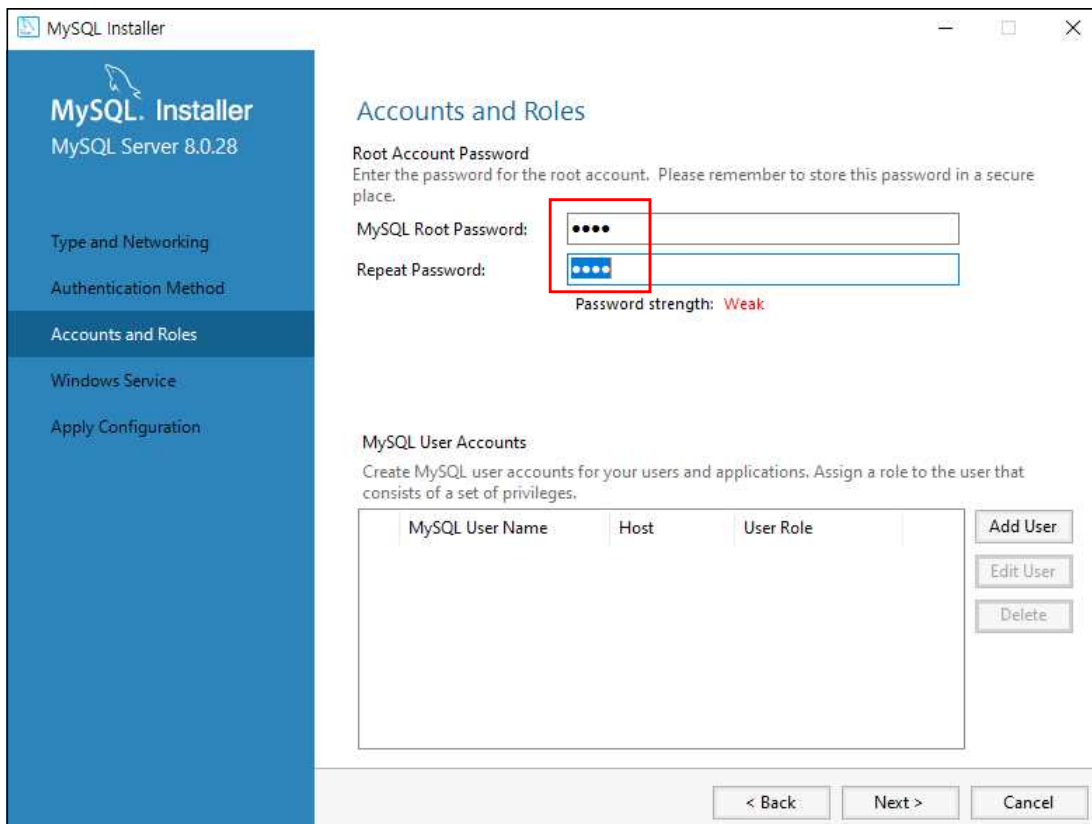


MySQL

8) [환경설정1] : Authentication Method – [Use Strong Password...] 선택된 상태에서 Next

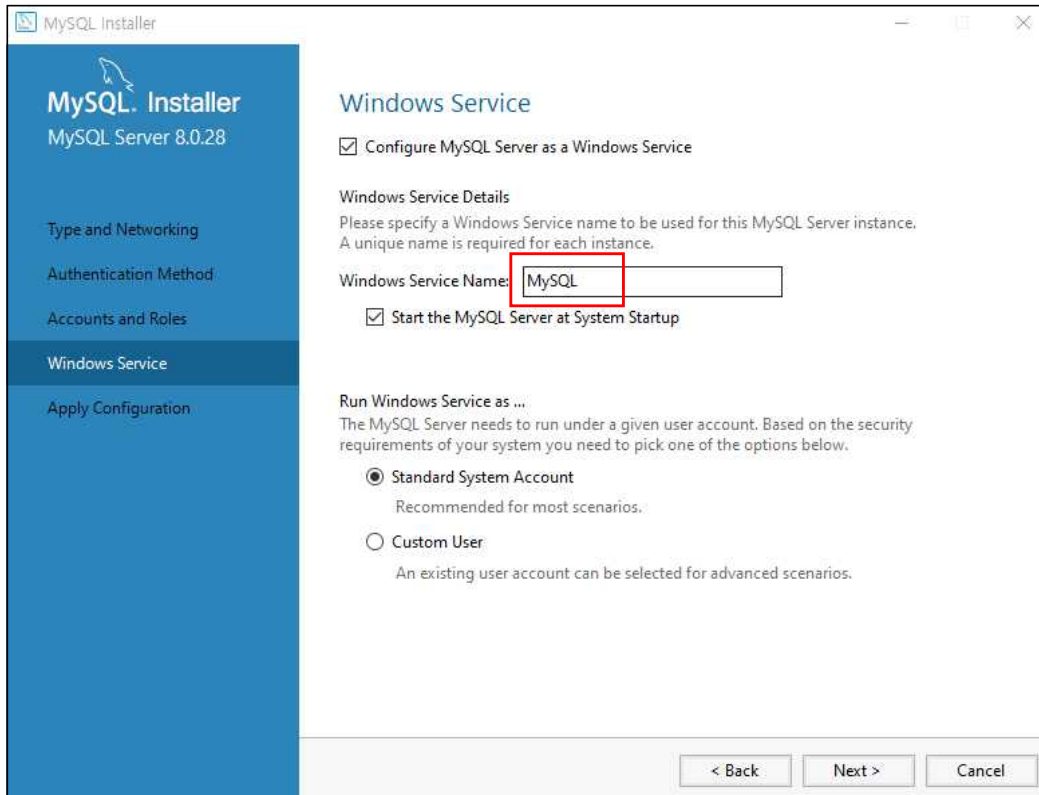


9) [환경설정1] : Accounts and Roles – MySQL 관리자(root)의 비밀번호 지정 후 Next
- MySQL Root Password : 1234

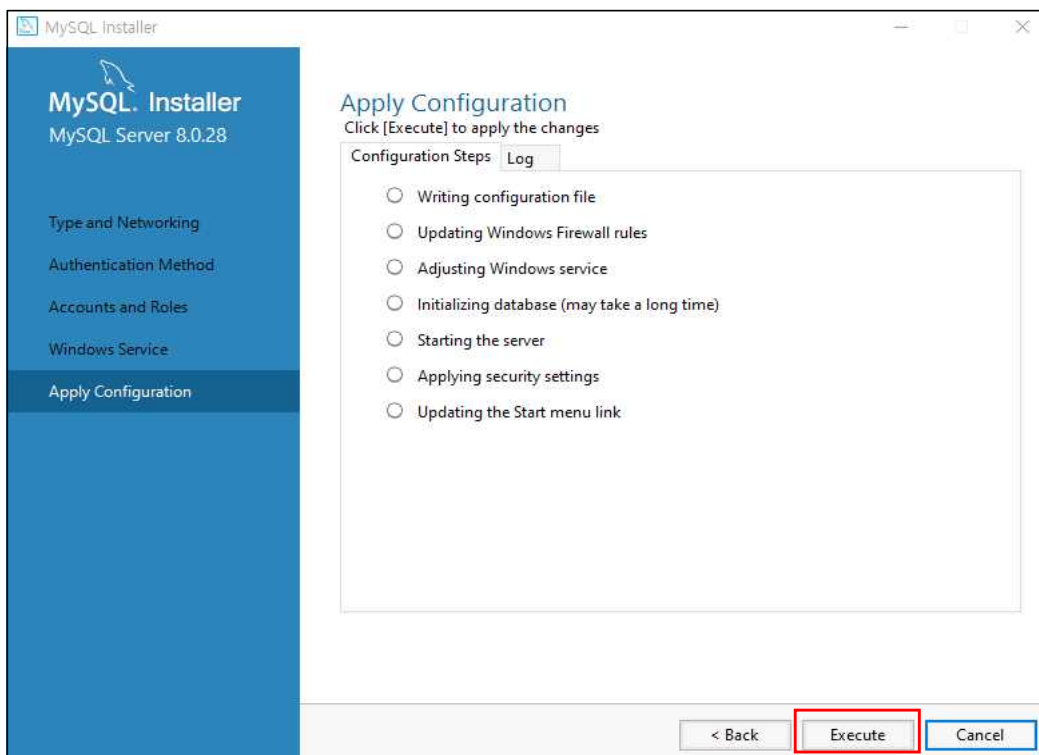


MySQL

- 10) [환경설정1] : Windows Service - Windows Service Name 설정 후 Next
- MySQL 서버를 Windows의 서비스로 등록하기 위한 설정
 - Windows Service Name : MySQL로 지정
 - Windows Service Name은 Windows의 [관리도구] - [서비스]에서 확인 가능함.

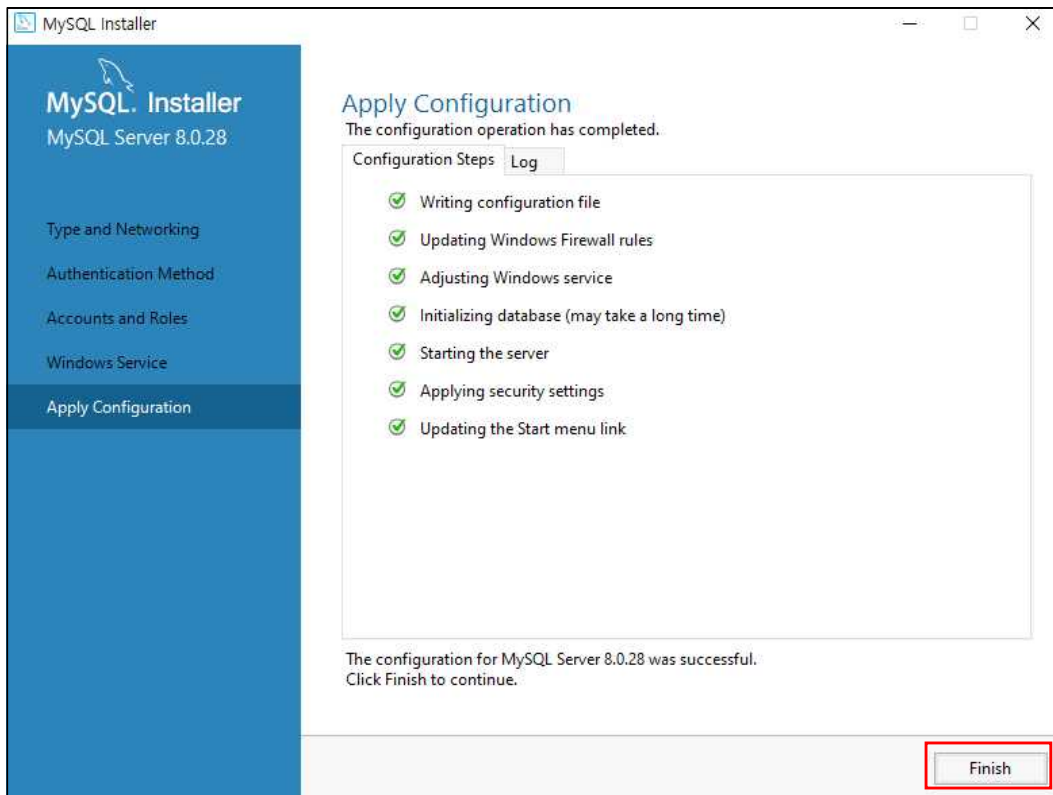


- 11) [환경설정1] : Apply Configuration – Execute 클릭 후 설정 내용 적용

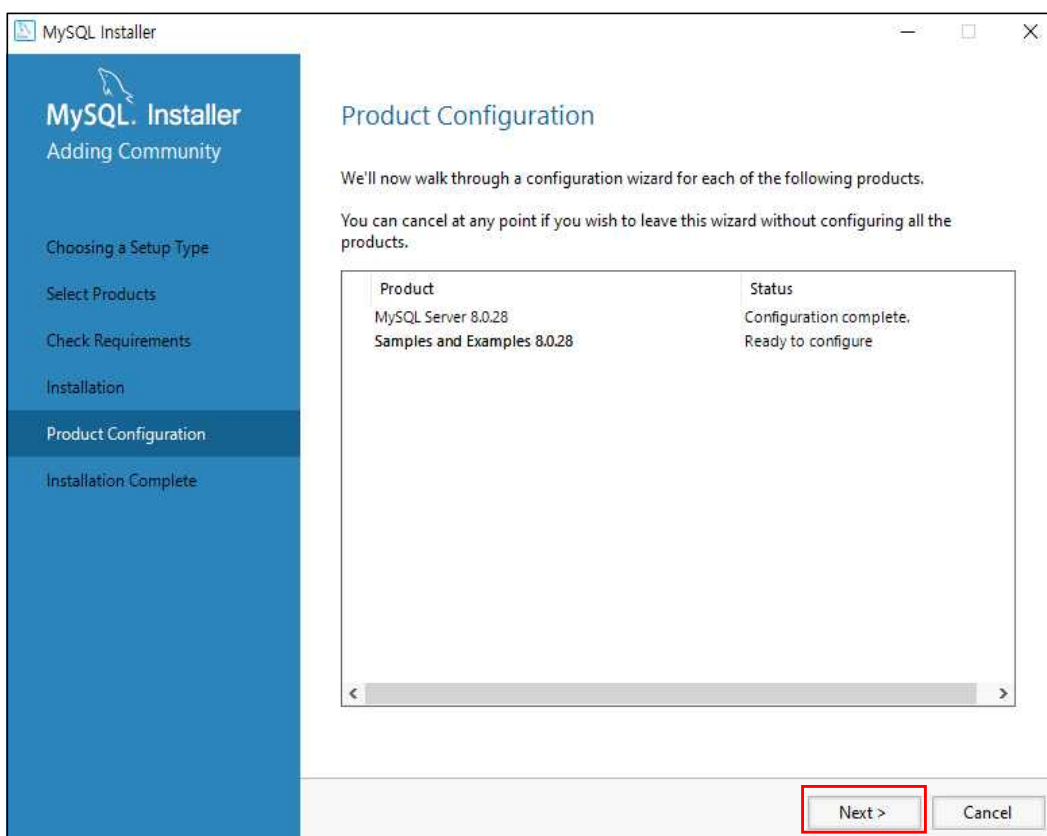


MySQL

12) [환경설정1] : Apply Configuration – 설정이 완료되면 Finish 클릭 후 설정 종료



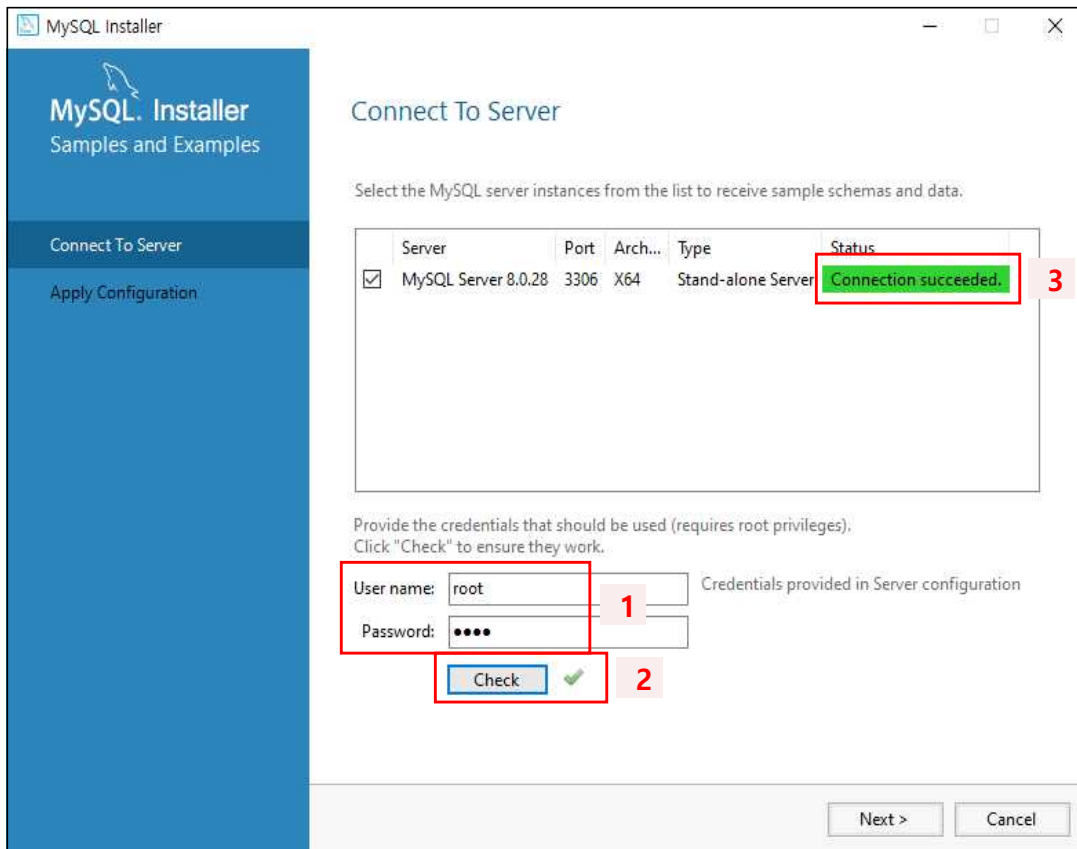
13) Product Configuration – MySQL Server 설정은 완료되었으며, Samples and Examples 설정 진행을 위해 Next



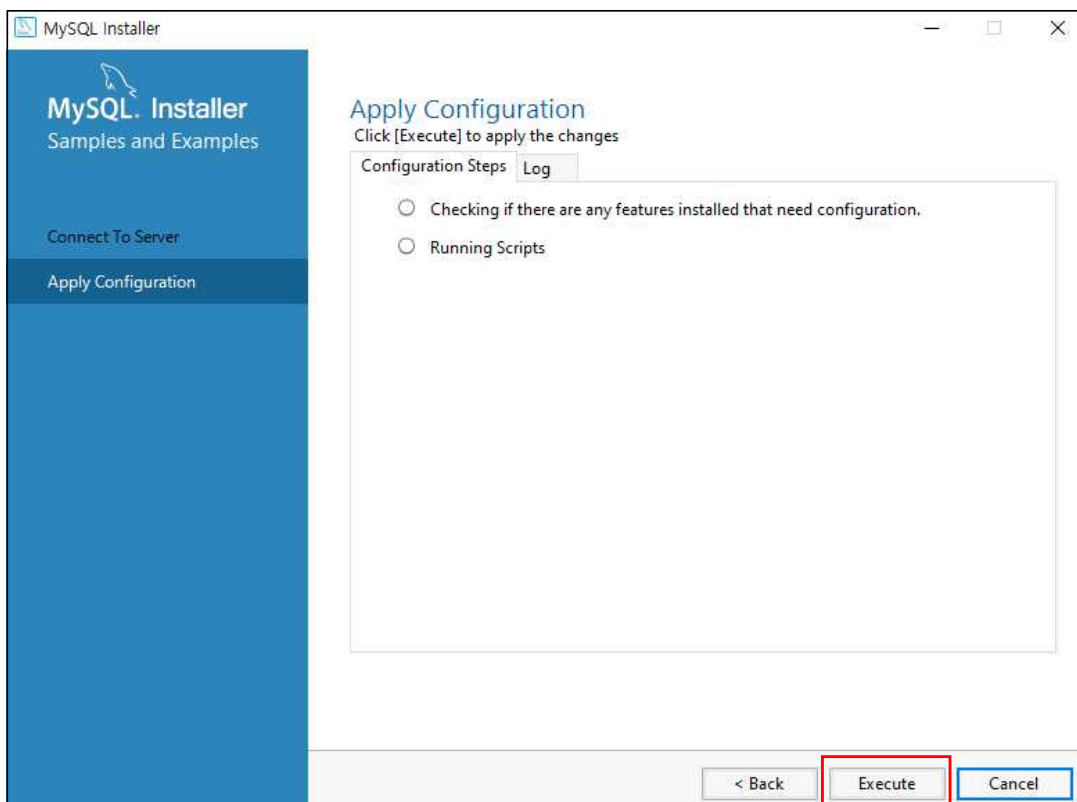
MySQL

14) [환경설정2] : Connect To Server – 서버 연결 후 Next

- Username과 Password(1234) 입력 후 Check 클릭 시 연결 성공 메시지 확인

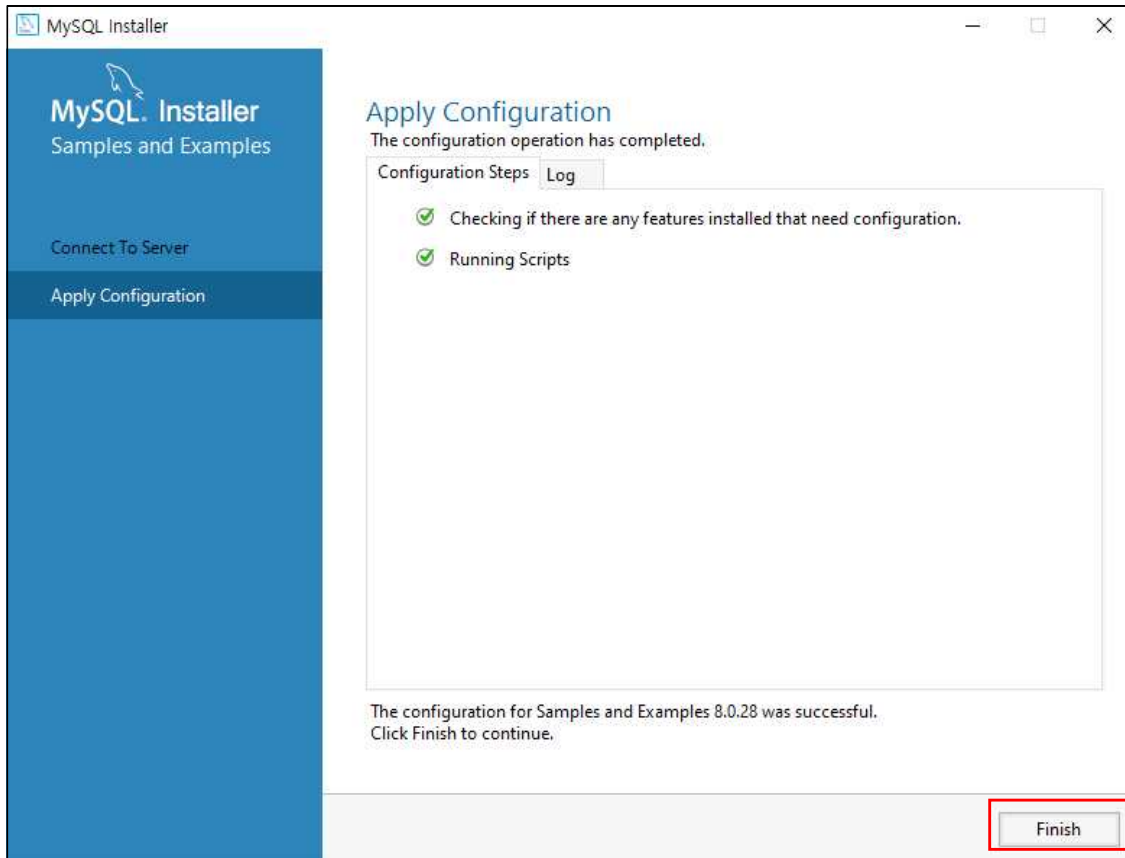


15) [환경설정2] : Apply Configuration – Execute 클릭 후 설정 내용 적용

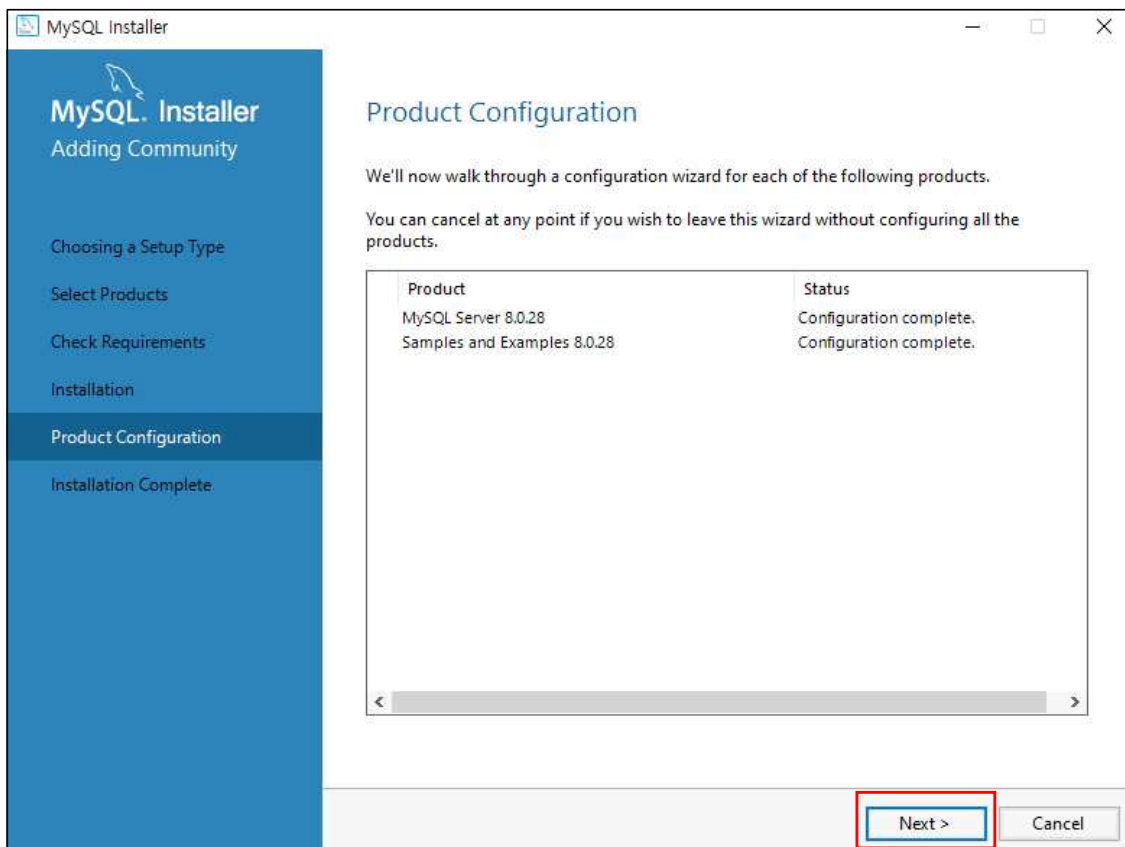


MySQL

16) [환경설정2] : Apply Configuration – 설정이 완료되면 Finish 클릭 후 설정 종료

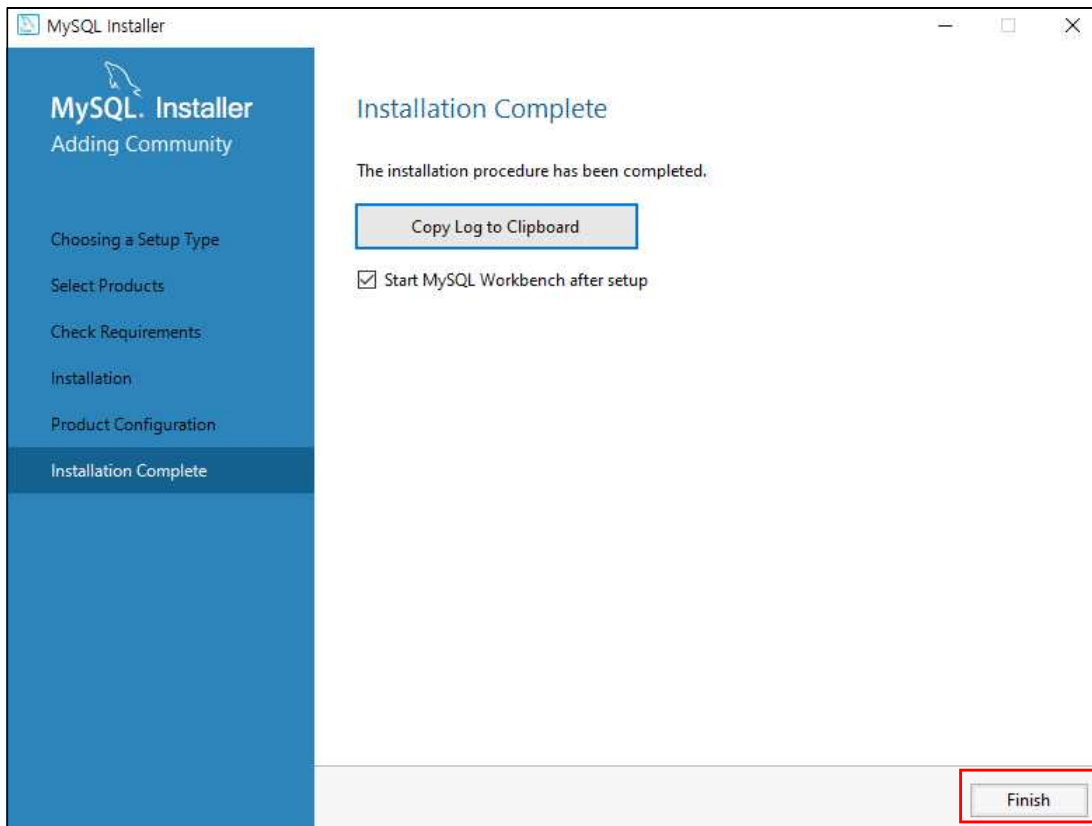


17) Product Configuration – 환경 설정 완료 후 Next

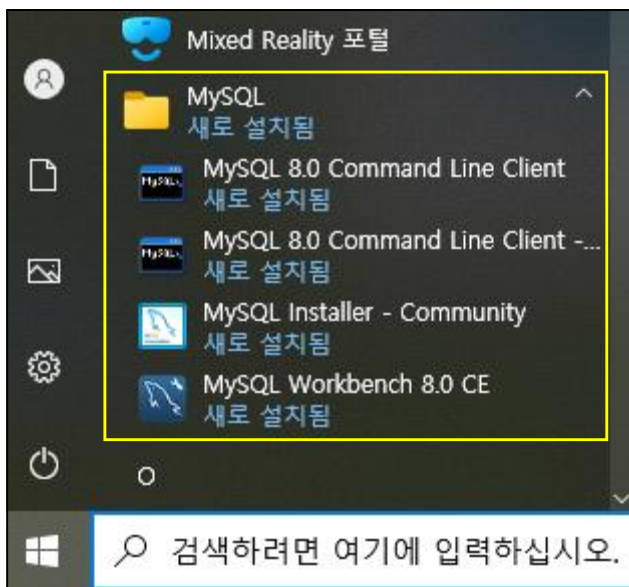


MySQL

18) Installation Complete – Finish를 클릭해서 MySQL 설치 완료

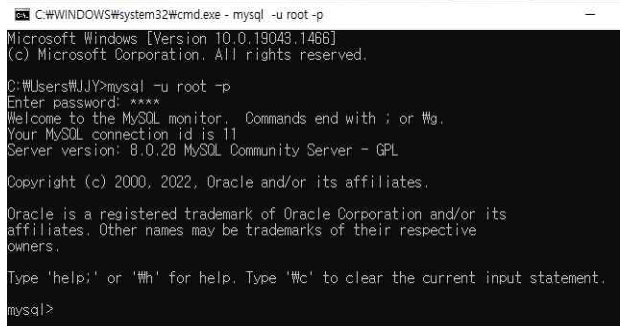
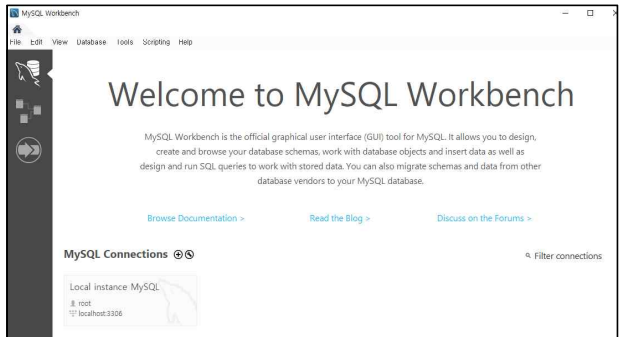


19) Windows의 [시작] 메뉴에서 MySQL 설치 확인



2. MySQL 접속 및 환경설정

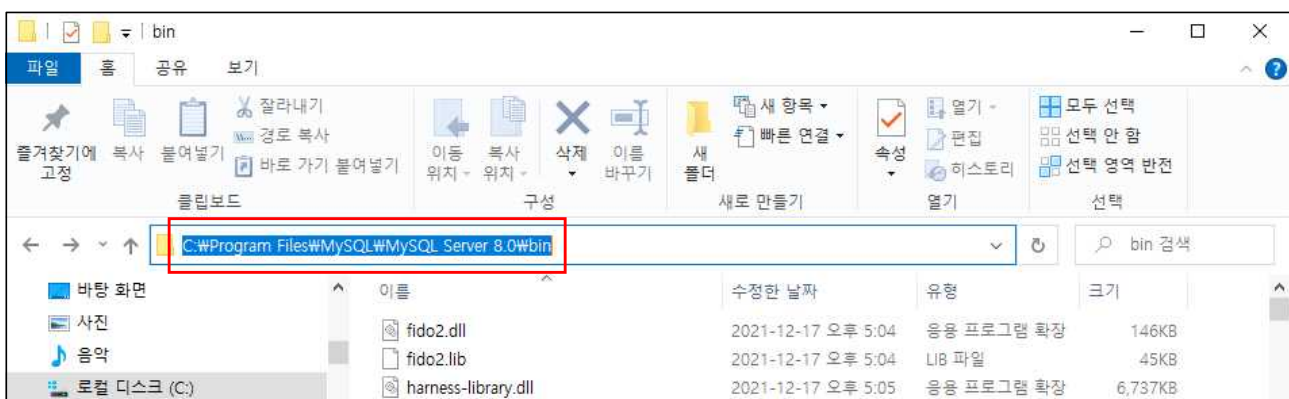
1. MySQL DBMS 접속 방법

MySQL 명령프롬프트	MySQL Workbench
MySQL과 관련된 작업을 TUI로 처리하고 결과를 확인할 수 있는 툴	MySQL과 관련된 대부분의 작업을 GUI로 처리하고 결과를 확인할 수 있는 통합 개발 툴
 <pre> C:\WINDOWS\system32\cmd.exe - mysql -u root -p Microsoft Windows [Version 10.0.19043.1466] (c) Microsoft Corporation. All rights reserved. C:\Users\JJY>mysql -u root -p Enter password: **** Welcome to the MySQL monitor. Commands end with ; or \g. Your MySQL connection id is 11 Server version: 8.0.28 MySQL Community Server - GPL Copyright (c) 2000, 2022, Oracle and/or its affiliates. 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> </pre>	

2. MySQL의 실행 파일이 있는 경로 Path에 추가하기

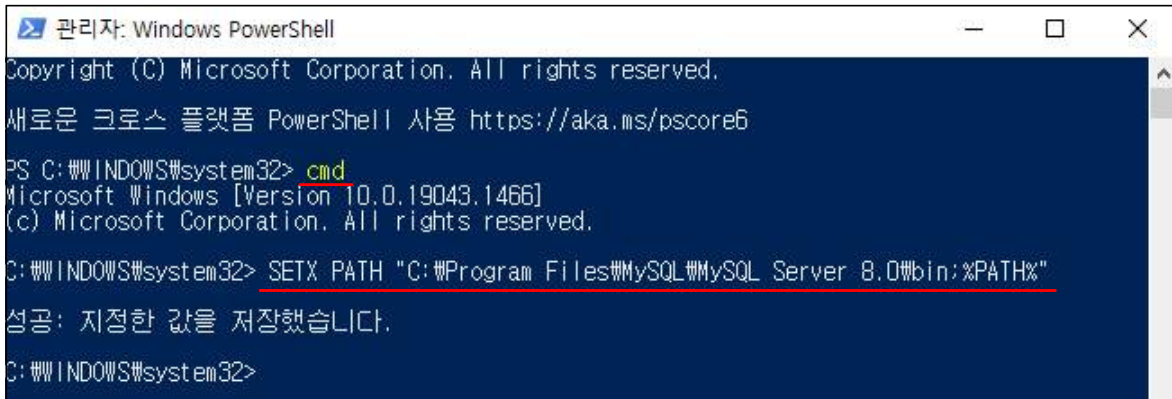
1) MySQL 관련 실행 파일 폴더로 이동

- C드라이브 >> Program Files >> MySQL >> MySQL Server 8.0 >> bin 이동 후 주소 표시줄을 선택하고 [Ctrl + C]를 눌러 복사



MySQL

- 2) Windows의 [시작]에서 오른쪽 마우스 클릭 - [Windows PowerShell(관리자)] 실행
- cmd 입력 : 명령 프롬프트로 전환
 - Path 추가 : SETX PATH "C:\Program Files\MySQL\MySQL Server 8.0\bin;%PATH%"



```
관리자: Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

새로운 크로스 플랫폼 PowerShell 사용 https://aka.ms/pscore6

PS C:\WINDOWS\system32> cmd
Microsoft Windows [Version 10.0.19043.1466]
(c) Microsoft Corporation. All rights reserved.

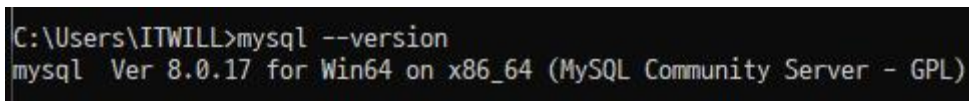
C:\WINDOWS\system32> SETX PATH "C:\Program Files\MySQL\MySQL Server 8.0\bin;%PATH%"
성공: 지정한 값을 저장했습니다.

C:\WINDOWS\system32>
```

- 3) 컴퓨터 재부팅 진행

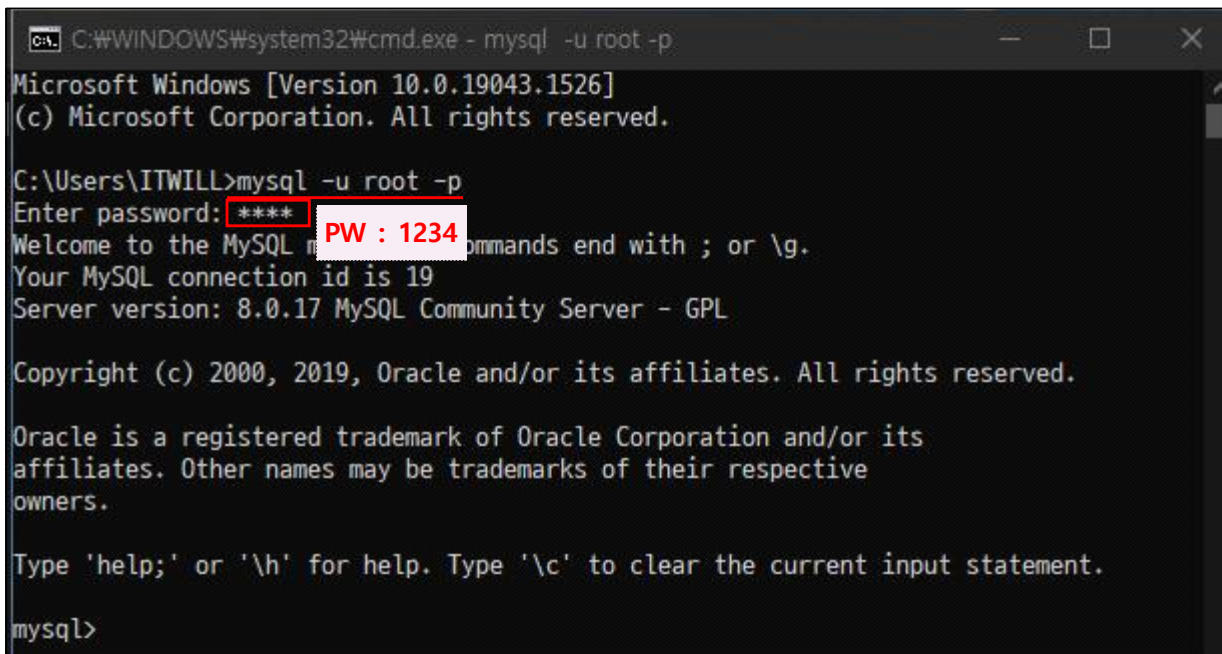
3. MySQL 명령프롬프트 사용하기

- 1) 명령프롬프트(cmd)창 열기
- Windows의 [시작] 우클릭 - [실행] - cmd 입력 후 확인
- 2) 설치된 MySQL 버전 확인



```
C:\Users\ITWILL>mysql --version
mysql Ver 8.0.17 for Win64 on x86_64 (MySQL Community Server - GPL)
```

- 3) MySQL 접속하기



```
C:\WINDOWS\system32\cmd.exe - mysql -u root -p
Microsoft Windows [Version 10.0.19043.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ITWILL>mysql -u root -p
Enter password: ***** PW : 1234
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 19
Server version: 8.0.17 MySQL Community Server - GPL

Copyright (c) 2000, 2019, 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>
```


MySQL

3) MySQL 명령어 테스트

① help 명령어 실행

```
mysql> help

For information about MySQL products and services, visit:
  http://www.mysql.com/
For developer information, including the MySQL Reference Manual, visit:
  http://dev.mysql.com/
To buy MySQL Enterprise support, training, or other products, visit:
  https://shop.mysql.com/

List of all MySQL commands:
Note that all text commands must be first on line and end with ';'
?          (\?) Synonym for `help`.
clear      (\c) Clear the current input statement.
connect    (\r) Reconnect to the server. Optional arguments are db and host.
delimiter  (\d) Set statement delimiter.
ego        (\G) Send command to mysql server, display result vertically.
exit       (\q) Exit mysql. Same as quit.
go         (\g) Send command to mysql server.
help       (\h) Display this help.
notee      (\t) Don't write into outfile.
print      (\p) Print current command.
prompt     (\R) Change your mysql prompt.
quit       (\q) Quit mysql.
rehash     (\#) Rebuild completion hash.
source     (\.) Execute an SQL script file. Takes a file name as an argument.
status     (\s) Get status information from the server.
tee        (\T) Set outfile [to_outfile]. Append everything into given outfile.
use        (\u) Use another database. Takes database name as argument.
charset    (\C) Switch to another charset. Might be needed for processing binlog with multi-byte charsets.
warnings   (\W) Show warnings after every statement.
nowarning  (\w) Don't show warnings after every statement.
resetconnection(\x) Clean session context.

For server side help, type 'help contents'
```

② show databases 명령어 실행

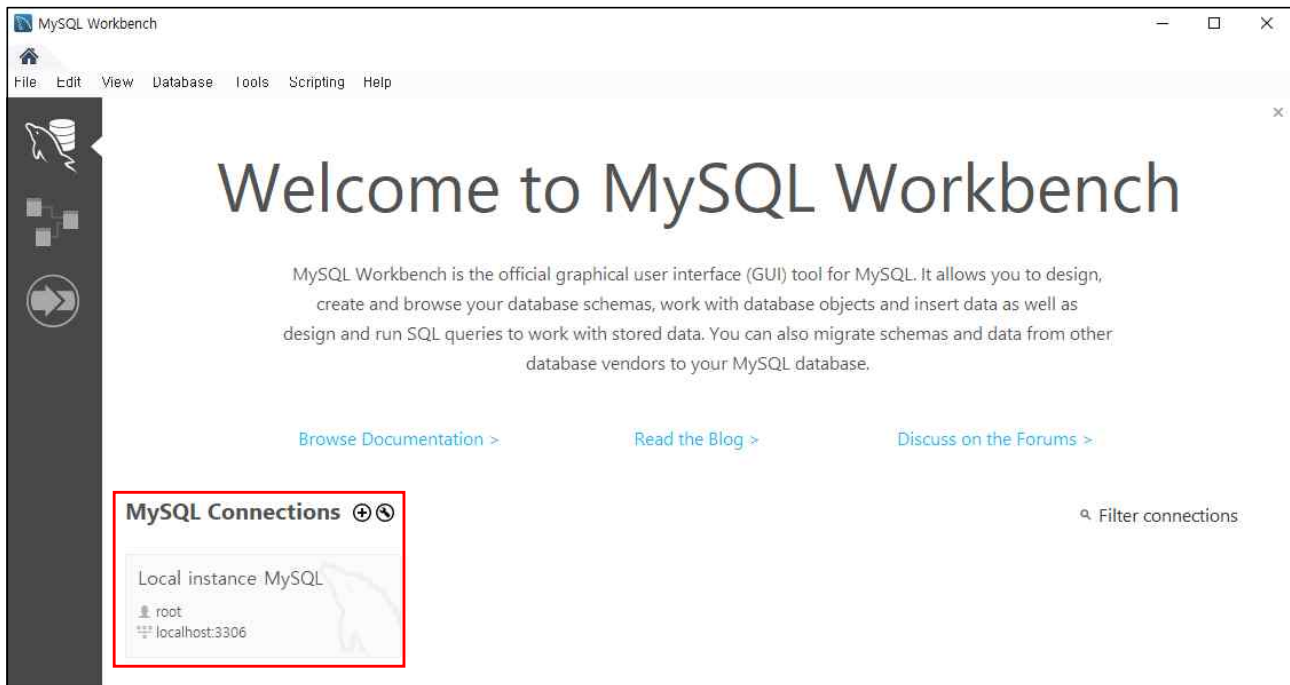
```
mysql> show databases;
+-----+
| Database |
+-----+
| employees |
| information_schema |
| mysql |
| performance_schema |
| sakila |
| shopdb |
| sys |
| world |
+-----+
8 rows in set (0.00 sec)
```


MySQL

4. MySQL Workbench 사용하기

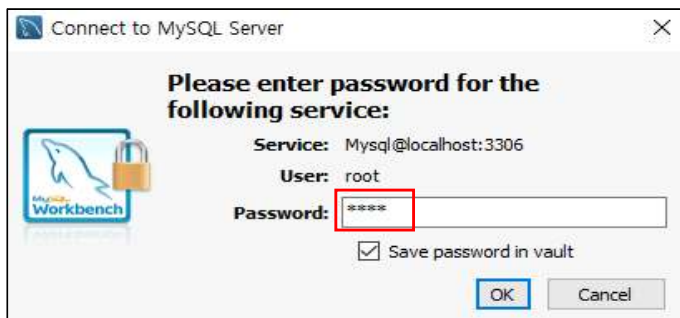
1) MySQL Workbench 실행

- MySQL Server에 연결된 상태는 아님.



2) MySQL Server에 접속하기

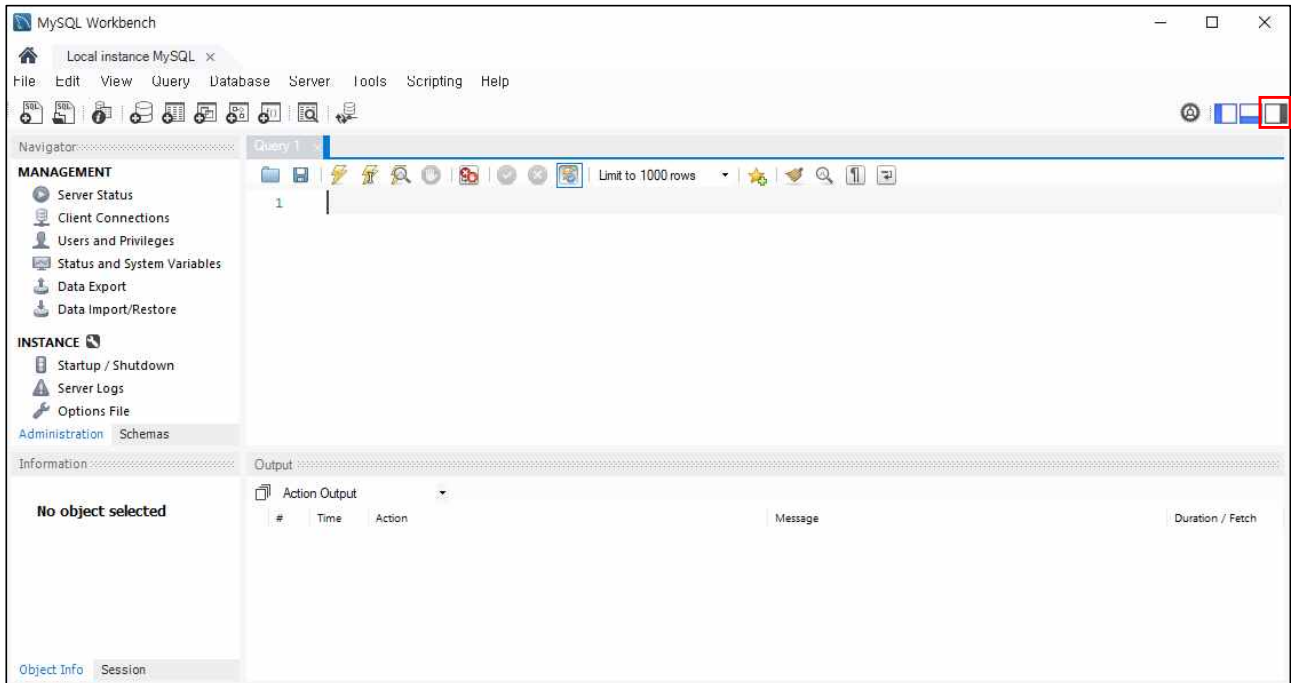
- MySQL connections - [Local instance MySQL] 클릭 - [Connect to MySQL Server] 창에 root 패스워드(1234) 입력 - OK 클릭



MySQL

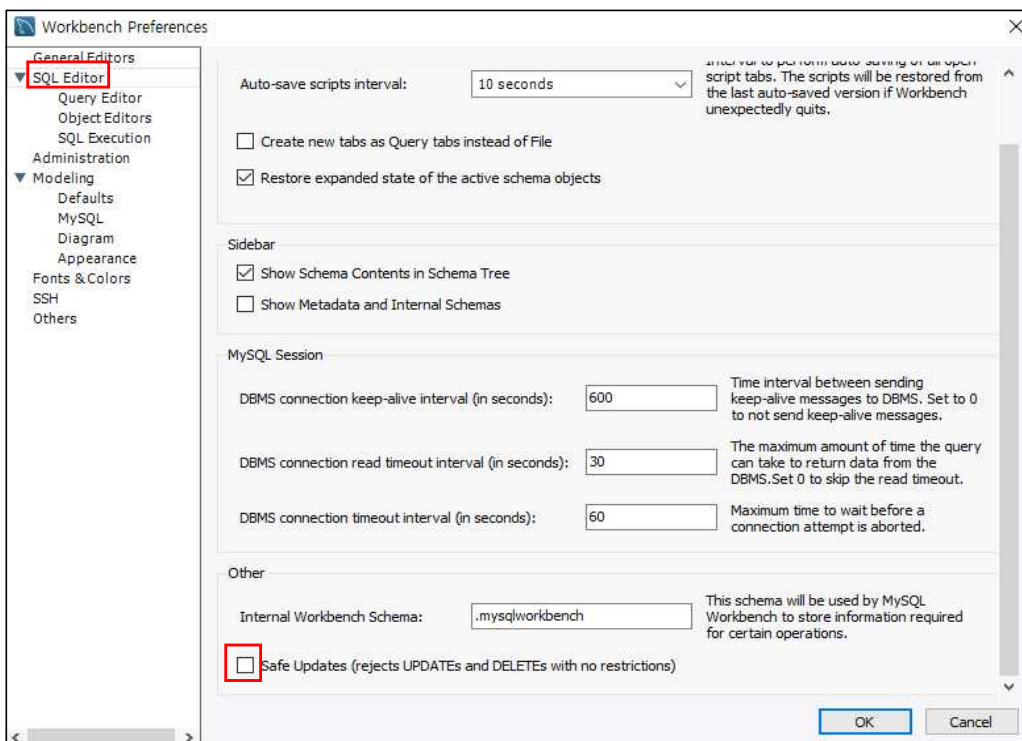
3) MySQL Server에 접속된 MySQL Workbench 화면 확인

- Navigator(왼쪽), Query1(중간), Output(아래) 부분만 사용 예정
- 오른쪽 상단 세 개의 아이콘 중 가장 오른쪽 아이콘을 클릭해서 사용하지 않는 SQL Additions 창 없애기



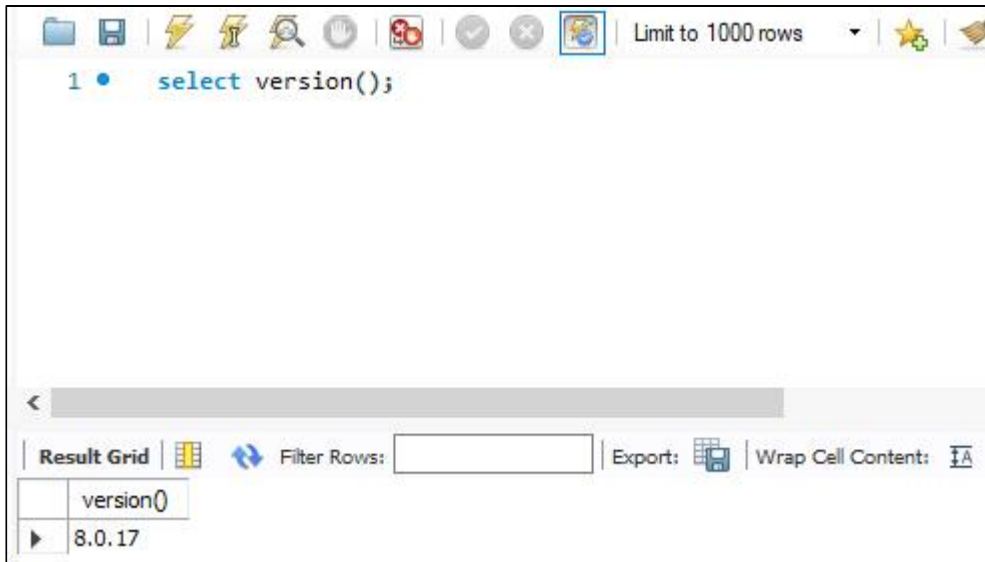
4) MySQL Workbench 설정

- [Edit] - [Preferences] - 왼쪽 [SQL Editor] 클릭 - 오른쪽 하단 [Safe Updates] 체크 해제 후 OK
- 이유 : 원활한 update와 delete를 위함.

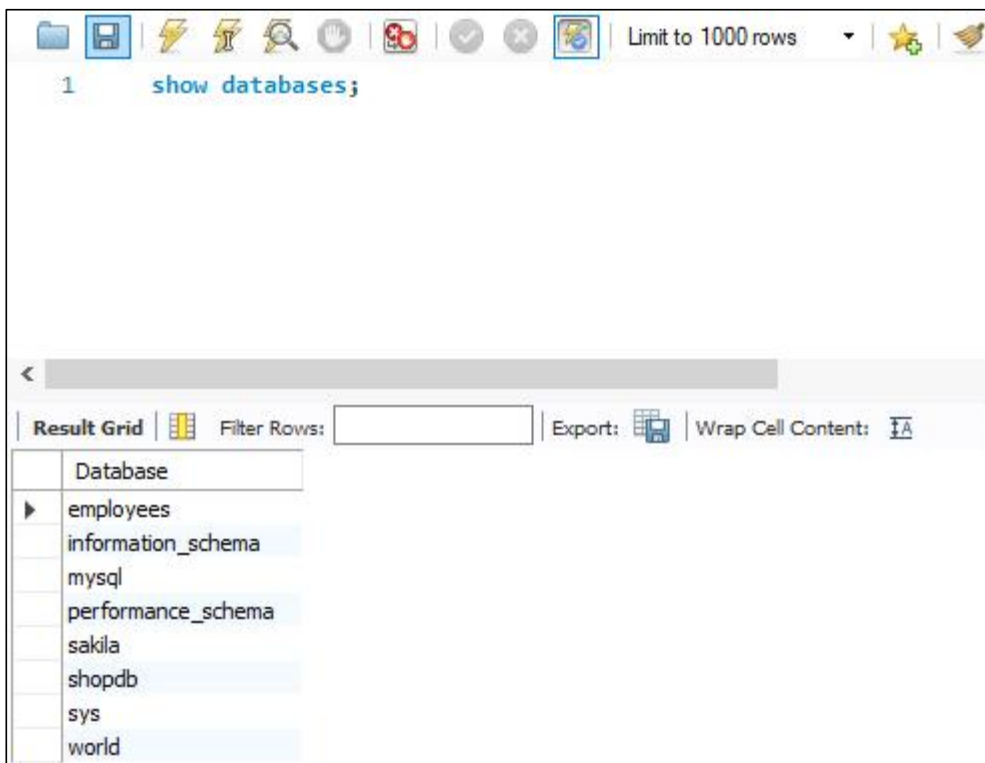


MySQL

5) 설치된 MySQL 버전 확인



6) MySQL 명령어 테스트 - show databases 명령어 실행



3. 샘플 데이터베이스(스키마) 설치 및 확인

1) employees 데이터베이스(스키마) 설치

- employees.zip 파일을 C드라이브에서 압축 풀기
- Windows의 [시작]에서 오른쪽 마우스 클릭 - [Windows PowerShell(관리자)] 실행

```
관리자: Windows PowerShell - mysql -u root -p

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

새로운 크로스 플랫폼 PowerShell 사용 https://aka.ms/pscore6

PS C:\WINDOWS\system32> cmd
Microsoft Windows [Version 10.0.19043.1466]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32> cd %employees%
C:\employees> mysql -u root -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.28 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

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> source employees.sql
```

2) hr 데이터베이스(스키마) 설치(동일 방식)

- hr.zip 파일을 C드라이브에서 압축 풀기
- Windows의 [시작]에서 오른쪽 마우스 클릭 - [Windows PowerShell(관리자)] 실행

```
PS C> cmd
C> cd %hr%
C:\hr> mysql -u root -p
Enter password : 1234
mysql> source hr.sql
```

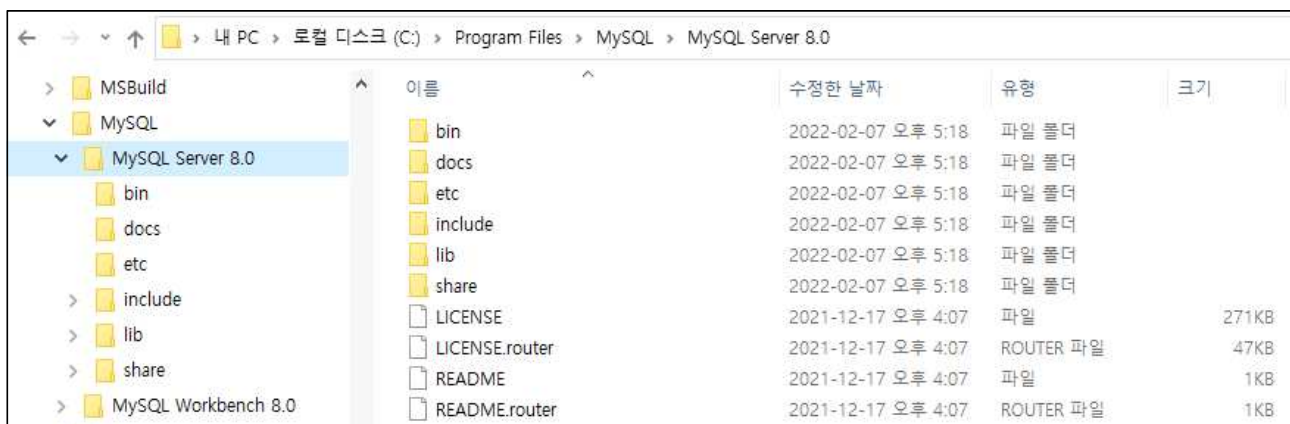
MySQL

3) 데이터베이스 확인

```
mysql> show databases;
+-----+
| Database |
+-----+
| employees |
| hr       |
| information_schema |
| mysql    |
| performance_schema |
| sakila   |
| sys      |
| world    |
+-----+
8 rows in set (0.00 sec)
```

4) MySQL 설치된 폴더 확인

① C:\Program Files\MySQL\MySQL Server 8.0으로 이동



The screenshot shows a Windows File Explorer window with the address bar set to 'C:\Program Files\MySQL\MySQL Server 8.0'. The left sidebar shows the folder hierarchy: '내 PC' > '로컬 디스크 (C:) > 'Program Files > 'MySQL > 'MySQL Server 8.0'. The main pane displays a list of files and folders with columns for '이름' (Name), '수정된 날짜' (Modified Date), '유형' (Type), and '크기' (Size). The folders listed are bin, docs, etc, include, lib, and share. The files listed are LICENSE (271KB), LICENSE.router (47KB), README (1KB), and README.router (1KB).

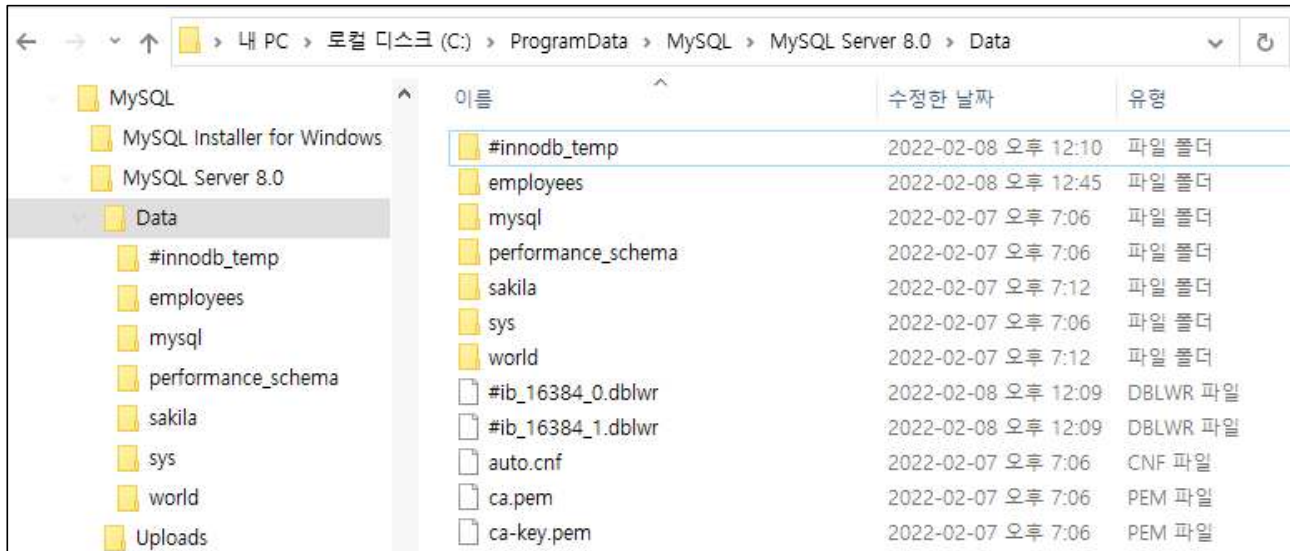
이름	수정된 날짜	유형	크기
bin	2022-02-07 오후 5:18	파일 폴더	
docs	2022-02-07 오후 5:18	파일 폴더	
etc	2022-02-07 오후 5:18	파일 폴더	
include	2022-02-07 오후 5:18	파일 폴더	
lib	2022-02-07 오후 5:18	파일 폴더	
share	2022-02-07 오후 5:18	파일 폴더	
LICENSE	2021-12-17 오후 4:07	파일	271KB
LICENSE.router	2021-12-17 오후 4:07	ROUTER 파일	47KB
README	2021-12-17 오후 4:07	파일	1KB
README.router	2021-12-17 오후 4:07	ROUTER 파일	1KB

폴더	역할
bin(★)	MySQL 서버 프로그램, 클라이언트 프로그램 및 유틸리티 프로그램 파일
docs	도움말 또는 설정 파일
etc	설정 파일 샘플
include	응용 프로그램을 개발할 때 필요한 헤더 파일
lib	MySQL 관련 라이브러리 파일
share	기타 지원 파일, 각 언어별 오류 메시지 파일 등

MySQL

② C:\ProgramData\MySQL\MySQL Server 8.0\Data로 이동

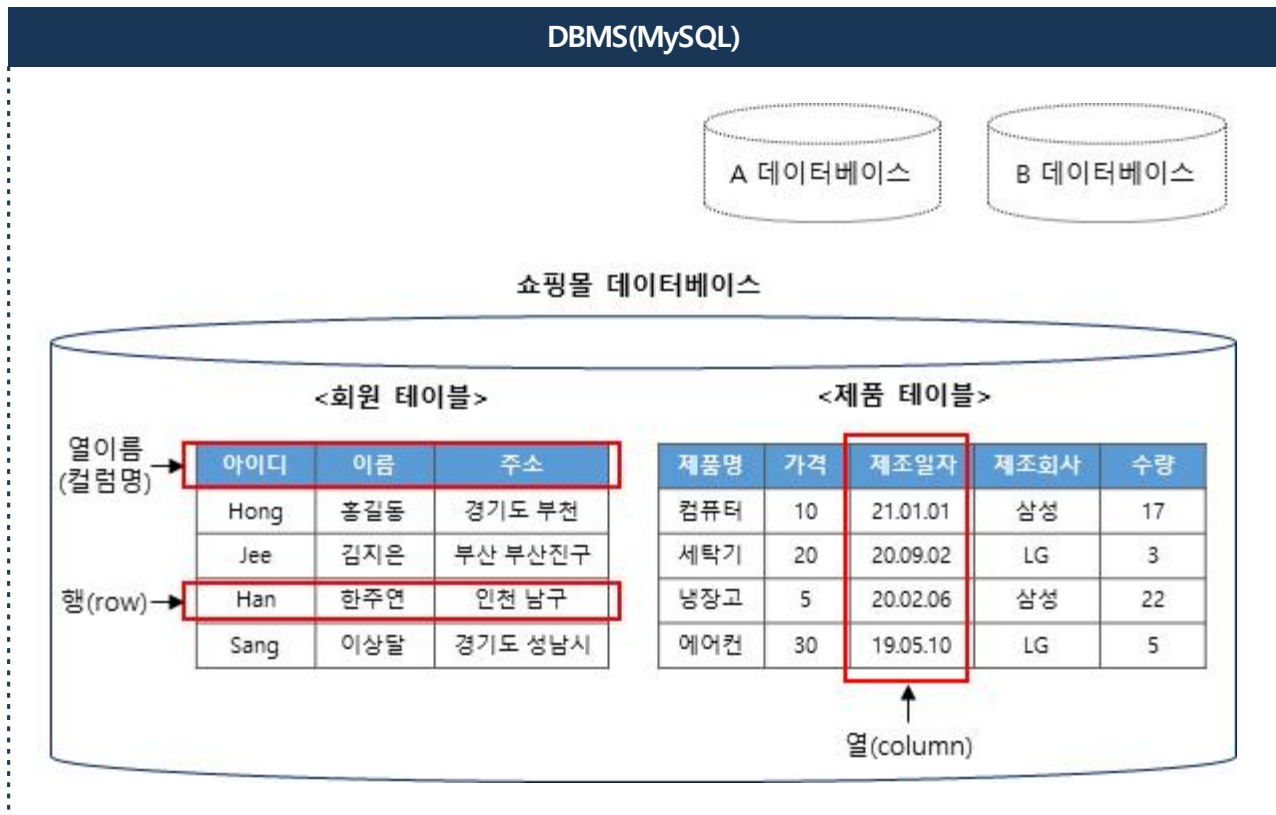
- ProgramData 폴더 안보이는 경우 [보기]-[숨긴 항목] 체크 시 나타남.



이름	수정한 날짜	유형
#innodb_temp	2022-02-08 오후 12:10	파일 폴더
employees	2022-02-08 오후 12:45	파일 폴더
mysql	2022-02-07 오후 7:06	파일 폴더
performance_schema	2022-02-07 오후 7:06	파일 폴더
sakila	2022-02-07 오후 7:12	파일 폴더
sys	2022-02-07 오후 7:06	파일 폴더
world	2022-02-07 오후 7:12	파일 폴더
#ib_16384_0.dblwr	2022-02-08 오후 12:09	DBLWR 파일
#ib_16384_1.dblwr	2022-02-08 오후 12:09	DBLWR 파일
auto.cnf	2022-02-07 오후 7:06	CNF 파일
ca.pem	2022-02-07 오후 7:06	PEM 파일
ca-key.pem	2022-02-07 오후 7:06	PEM 파일

4. 데이터베이스 관련 용어

1. 데이터베이스 필수 용어

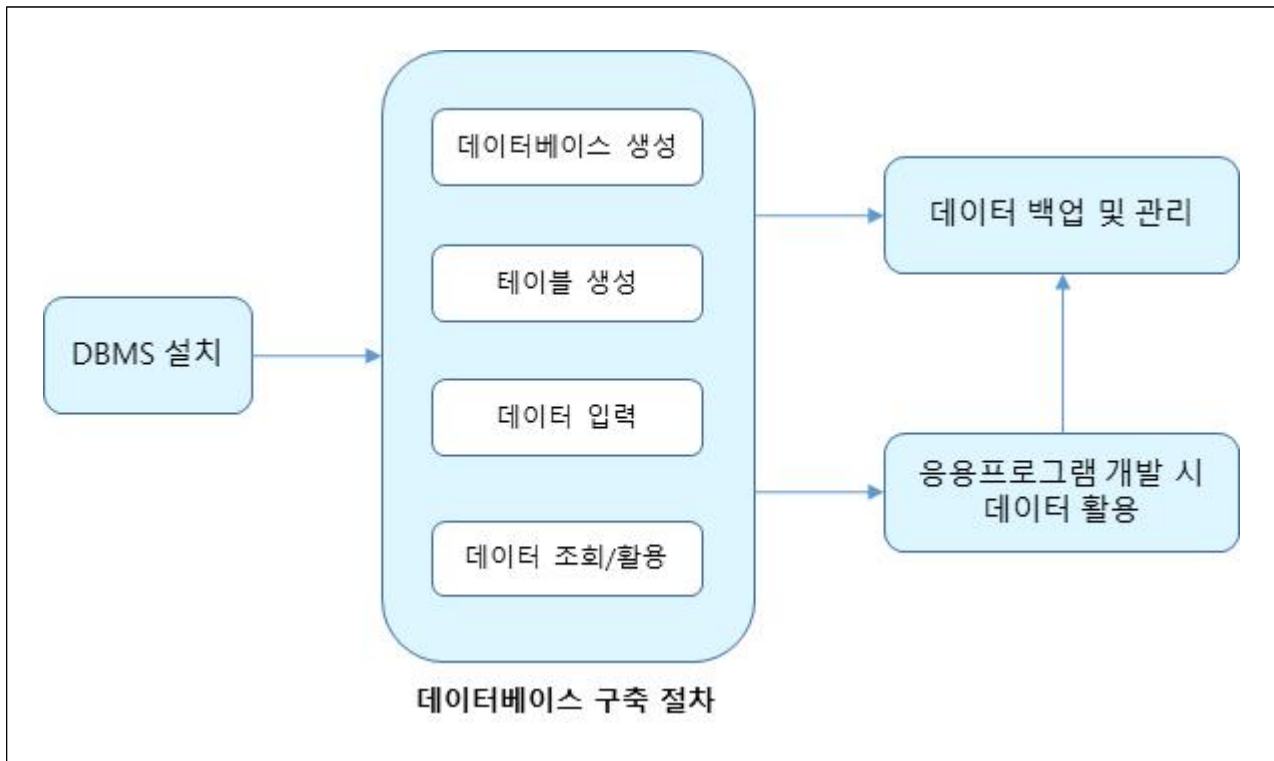


용어	설명
DBMS	Database Management System(데이터베이스 관리 시스템)의 약자로 데이터베이스를 관리하는 시스템 또는 프로그램. DBMS의 종류로 Oracle, MySQL, SQL Server, MariaDB 등이 있음.
데이터베이스(DB)	조직화된 정보들의 모음 또는 데이터 집합. 데이터베이스는 고유한 이름을 가져야 함.
객체(Object)	사람, 사물, 장소, 개념, 사건과 같은 유무형의 정보를 가지고 있는 독립적인 실체. 객체 종류는 테이블, 뷰, 인덱스 등이 있음. 데이터베이스 내에서 가장 많이 사용되는 객체는 테이블임.
테이블(Table)	데이터베이스 내에서 데이터를 저장할 때 가장 많이 사용하는 객체 유형. 열(column)과 행(row)로 이루어져 있음.
데이터(data)	테이블과 같은 객체에 저장된 실제 정보를 의미함.

MySQL

열(column)	컬럼 또는 속성(Attribute)이라고도 함. 테이블에서 세로 부분을 의미하며, 하나의 테이블은 여러 개의 열(column)으로 구성되어 있음.
열 이름(column name)	각 열(column)을 구분하기 위한 이름. 열이름 또는 컬럼명은 테이블 내에서 고유해야 함.
행(row)	로우 또는 레코드라고도 함. 테이블에서 가로에 해당하는 데이터 전체를 의미함.
SQL	사람과 DBMS가 소통하기 위해 사용하는 언어.
기본키(Primary key)	테이블의 행을 구분해 줄 수 있는 대표 컬럼에게 부여하는 제약 조건. 기본키가 설정된 컬럼에는 중복값, 빈값 들어올 수 없음.
외래키(Foreign key)	테이블과 테이블을 연결해 주는 제약조건.
스키마(Schema)	DBMS들마다 스키마의 의미는 다르게 사용되고 있음. 스키마란 데이터를 담는 그릇을 의미함. MySQL - 스키마가 데이터베이스와 동일한 의미로 사용됨. Oracle - 스키마가 유저와 동일한 의미로 사용됨.

5. 데이터베이스 구축 절차



※ 인터넷 쇼핑몰을 운영하기 위한 '쇼핑몰' 데이터베이스를 구축해 보자.

6. 데이터베이스 생성

[문법] SQL> create schema 스키마명;

1) Workbench 실행 후 MySQL에 root 사용자로 접속

2) '쇼핑몰' 데이터베이스 생성

- Schema 이름 : db

[이름 결정 시 유의사항]

- 데이터베이스(스키마)명, 테이블명, 컬럼명 등은 영문을 사용해야 한다.
- 이름은 영문으로 시작해야하며, 영문/숫자/특수문자(underscore, #, \$)가 혼합될 수는 있다.
- 실무에서 데이터베이스명, 테이블명, 컬럼명 등을 한글로 지정하는 경우는 거의 없으며, 그 이유는 호환성 등의 문제가 발생할 소지가 많기 때문이다.
- 당연히 실제 데이터는 영문, 한글 모두 사용 가능하다.

SQL> create schema shopdb;

3) Workbench 왼쪽 [SCHEMAS] 메뉴 - 새로고침 - shopdb 생성 확인

7. 테이블 생성

```
[문법] SQL> create table 테이블명
( 컬럼명1 데이터타입(컬럼사이즈),
  [컬럼명2 데이터타입(컬럼사이즈),
  컬럼명3 데이터타입(컬럼사이즈)] );
```

```
[문법] SQL> create table 테이블명
( 컬럼명1 데이터타입(컬럼사이즈) default 기본값,
  [컬럼명2 데이터타입(컬럼사이즈),
  컬럼명3 데이터타입(컬럼사이즈)] );
```

```
[문법] SQL> create table 테이블명
( 컬럼명1 데이터타입(컬럼사이즈) 제약조건유형,
  [컬럼명2 데이터타입(컬럼사이즈),
  컬럼명3 데이터타입(컬럼사이즈)] );
```

1. 데이터 타입

데이터타입		설명
숫자 데이터 타입	SMALLINT	정수형 데이터 타입(2 byte) -32,768 ~ 32,767
	INT	정수형 데이터 타입(4 byte) -2,147,483,648 ~ 2,147,483,647
	BIGINT	정수형 데이터 타입(8byte) 무제한 수 표현 가능
	FLOAT(p)	부동 소수형 데이터 타입(4 byte) 소수점 아래 7자리까지 표현 가능
문자 데이터 타입	CHAR(n)	고정 길이 문자형 n : 1~255 byte까지 지정 가능
	VARCHAR(n)	가변 길이 문자형 n : 1~65,535 byte까지 지정 가능
	LONGTEXT	대용량 문자열 데이터 타입
	LONGBLOB	대용량 바이너리 데이터 타입
날짜/시간 데이터 타입	DATE	날짜 데이터 타입 YYYY-MM-DD 형식으로 사용됨.
	DATETIME	날짜/시간 데이터 타입 YYYY-MM-DD HH:MM:SS 형식으로 사용됨.

2. 제약조건

제약조건 유형	설명
NOT NULL	<ul style="list-style-type: none"> - 컬럼에 NULL값이 들어오는 것을 막기 위한 제약조건 - 사번, 이름, 주민번호, 전화번호 등의 필수 컬럼에 선언한다. ※ null값이란? 모르는 값, 정의되지 않은 값이라고 하며, 0와 공백과는 다른 특수한 값이다.
UNIQUE	<ul style="list-style-type: none"> - 중복되지 않는 유일한 값을 입력해야 한다는 제약조건 - 주민번호, 전화번호, 메일 등 고유한 값이 필요한 컬럼에 선언한다.
PRIMARY KEY	<ul style="list-style-type: none"> - 기본키 제약조건으로 NOT NULL + UNIQUE의 성격을 모두 가지고 있는 제약조건 - PRIMARY KEY 제약조건이 선언된 컬럼에는 null값도 입력될 수 없으며, 중복된 값도 입력될 수 없다. - PRIMARY KEY 제약조건은 테이블당 한번만 선언이 가능하다. - 사번, 학번, 회원번호, 제품번호 등의 컬럼에 사용된다.
FOREIGN KEY	<ul style="list-style-type: none"> - 외래키 제약조건으로 특정 테이블에서 다른 테이블을 참조하는 제약조건 - FOREIGN KEY 제약조건 때문에 테이블과 테이블이 연결된다.
CHECK	<ul style="list-style-type: none"> - 컬럼이 만족해야하는 조건문을 지정하는 제약조건

3. 테이블 차트 및 테이블 생성 작업

1) 테이블명 : members

컬럼명	member_id	member_name	birth	job	phone	address
데이터타입	int	varchar	date	varchar	varchar	varchar
컬럼사이즈	-	8	-	20	20	80
제약조건	PK	NN	NN	-	UK	-

```
SQL> create table members
( member_id int primary key,
  member_name varchar(8) not null,
  birth date not null,
  job varchar(20),
  phone varchar(20) unique,
  address varchar(80) );
```

```
SQL> desc members;
```


MySQL

2) 테이블명 : products

컬럼명	prod_id	prod_name	price	make_date	company
데이터타입	int	varchar	int	date	varchar
컬럼사이즈	-	20	-	-	10
제약조건	PK	NN	CK (price > 0)	-	NN

```
SQL> create table products
      ( prod_id int primary key,
        prod_name varchar(20) not null,
        price int check (price > 0),
        make_date date,
        company varchar(10) not null );
```

```
SQL> desc products;
```

3) 테이블명 : orders

컬럼명	order_num	member_id	prod_id	order_date
데이터타입	int	int	int	date
컬럼사이즈	-	-	-	-
제약조건	PK	FK	FK	-
Default값	-	-	-	now()

```
SQL> select now()
      from dual;
```

```
SQL> create table orders
      ( order_num int,
        member_id int,
        prod_id int,
        order_date datetime default now(),
        primary key(order_num),
        foreign key(member_id) references members(member_id),
        foreign key(prod_id) references products(prod_id) );
```

```
SQL> desc orders;
```

8. 테이블 생성 Review 및 추가 속성

1. AUTO_INCREMENT 속성이란?

- 테이블 생성(또는 수정) 시 특정 컬럼에 auto_increment 속성을 부여할 경우 insert 시 자동으로 1부터 시작해서 1씩 증가하는 값을 반환해주는 속성이다.
- 시작 값 또는 증가 값을 변경하고 싶은 경우에는 테이블 생성 후 테이블 수정 구문(alter table)으로 변경 가능하다.
- primary key 또는 unique 제약조건이 지정된 컬럼만 auto_increment 속성을 활용할 수 있다.
- auto_increment 속성은 숫자 형식의 데이터 타입에만 활용 가능하다.
- auto_increment 속성이 지정된 컬럼은 insert 작업 시 null값을 지정하면 자동으로 값이 입력된다.

// 테이블 생성 시 auto_increment 속성 선언 문법

[문법] SQL> create table 테이블명

(컬럼1 int auto_increment primary key,
컬럼2 데이터타입,
컬럼3 데이터타입);

// 테이블 생성 후 auto_increment 입력값(시작값) 변경 (default = 1)

[문법] SQL> alter table 테이블명 auto_increment=입력값;

// 증가값(증가 사이즈) 변경 (default = 1)

[문법] SQL> set @@auto_increment_increment=증가값;

// 현재 몇 번까지 사용되었는지 확인하는 방법

[문법] SQL> select last_insert_id();

2. 테이블 생성 시 AUTO_INCREMENT 속성 활용하기

1) 테이블명 : stu20

컬럼명	stu_id	stu_name	age
데이터타입	int	varchar	int
컬럼사이즈	-	5	-
제약조건	PK	NN	CK (age > 19)
기타	auto_increment 설정	-	-

SQL> create table stu20

(stu_id int auto_increment primary key,
stu_name varchar(5) not null,
age int check (age > 19));

SQL> desc stu20;

9. 데이터 삽입

1. 테이블의 데이터 변경을 위한 SQL문

- 데이터 조작어(DML : Data Manipulation Language)
 - insert : 데이터 삽입
 - update : 데이터 수정
 - delete : 데이터 삭제

2. 데이터 삽입 : INSERT

```
[문법] SQL> insert into 테이블명[(컬럼1, 컬럼2, 컬럼3, ...)]  
      values (값1, 값2, 값3, ...);
```

1) members 테이블에 데이터 삽입 예제

① 테이블에 데이터 삽입 시 제약조건 유의 예제

- 문자 및 날짜는 작은 따옴표로 묶어서 표현한다.
- 날짜 : 년도-월-일 순서로 작성한다.

```
SQL> desc members;  
SQL> insert into members  
      values (100, '홍길동', '1991-12-30', '학생', '010-1111-1111', '부산 부산진구 부전동');  
SQL> select *  
      from members;
```

// 생략된 컬럼에는 null값 자동 삽입됨.

```
SQL> insert into members(member_id, member_name, birth, phone)  
      values (101, '김민수', '1990-03-05', '010-2222-2222');  
SQL> select *  
      from members;
```

// not null 제약조건이 선언된 컬럼에 값을 생략한 경우

```
SQL> insert into members(member_id, member_name, phone, address)  
      values (102, '최아영', '010-3333-3333', '서울 강남구 선릉로');  
=> Error Code: 1364. Field 'birth' doesn't have a default value
```

[수정]

```
SQL> insert into members(member_id, member_name, birth, phone, address)  
      values (102, '최아영', '1987-11-23', '010-3333-3333', '서울 강남구 선릉로');  
SQL> select *  
      from members;
```

// **unique 제약조건**이 선언된 컬럼에 중복 값이 삽입된 경우

```
SQL> insert into members(member_id, member_name, birth, job, phone)
      values (103, '홍길동', '1988-05-10', '회사원', '010-1111-1111');
```

=> Error Code: 1062. Duplicate entry '010-1111-1111' for key 'phone'

[수정]

```
SQL> insert into members(member_id, member_name, birth, job, phone)
      values (103, '홍길동', '1988-05-10', '회사원', '010-4444-4444');
```

```
SQL> select *
      from members;
```

// **primary key 제약조건**이 선언된 컬럼에 중복 값이 삽입된 경우

```
SQL> insert into members(member_id, member_name, birth, job)
      values (101, '강주영', '1998-08-09', '대학생');
```

=> Error Code: 1062. Duplicate entry '101' for key 'PRIMARY'

[수정]

```
SQL> insert into members(member_id, member_name, birth, job)
      values (104, '강주영', '1998-08-09', '대학생');
```

```
SQL> select *
      from members;
```

// 단, unique 제약조건이 선언된 컬럼에 null값은 허용함.

// **primary key 제약조건**이 선언된 컬럼에 null값이 삽입된 경우

```
SQL> insert into members(member_name, birth, job, phone, address)
      values ('고승현', '1995-07-10', '트레이너', '010-5555-5555', '경기도 부천시 원미구');
```

=> Error Code: 1364. Field 'member_id' doesn't have a default value

[수정]

```
SQL> insert into members(member_id, member_name, birth, job, phone, address)
      values (105, '고승현', '1995-07-10', '트레이너', '010-5555-5555', '경기도 부천시 원미구');
```

```
SQL> select *
      from members;
```

MySQL

② 추가 데이터 삽입하기

- 다음 5개의 행을 members 테이블에 삽입하시오.
- 삽입 후 삽입된 데이터를 조회하시오.

member_id	member_name	birth	job	phone	address
106	정유빈	1970-02-04	회사원	010-6666-6666	null
107	이영수	1988-12-06	null	010-7777-7777	null
108	김철수	1999-01-15	null	010-8888-8888	부산 해운대구 센텀로
109	최승현	1995-04-22	간호사	010-9999-9999	서울 강북고 수유동
110	한주연	2001-08-24	승무원	010-1010-1010	대구 수성구 수성로

2) products 테이블에 데이터 삽입 예제

① 테이블에 데이터 삽입 시 null값 삽입하기

```
// null값 수동 삽입 방법
SQL> desc products;
SQL> insert into products
      values (10, '냉장고', 500, null, '삼성');
SQL> select *
      from products;
```

② 추가 데이터 삽입하기

- 다음 5개의 행을 products 테이블에 삽입하시오.
- 삽입 후 삽입된 데이터를 조회하시오.

prod_id	prod_name	price	make_date	company
20	컴퓨터	150	2022-01-13	애플
30	세탁기	250	2020-03-10	LG
40	TV	200	2021-09-30	LG
50	전자렌지	50	2019-06-20	삼성
60	건조기	300	2021-07-09	LG

MySQL

3) orders 테이블에 데이터 삽입 예제

① 테이블에 데이터 삽입 시 제약조건 유의 예제

```
// datetime 데이터타입 컬럼에 데이터 삽입 예제
```

```
SQL> desc orders;
```

```
SQL> insert into orders
```

```
values (1, 101, 20, '2022-02-01');
```

```
SQL> select *
```

```
from orders;
```

```
SQL> insert into orders
```

```
values (2, 107, 40, '2022-02-05 17:51');
```

```
SQL> select *
```

```
from orders;
```

```
// now 함수 사용해서 오늘 날짜 삽입
```

```
SQL> insert into orders
```

```
values (3, 106, 50, now( ));
```

```
SQL> select *
```

```
from orders;
```

```
// insert 시 default 값이 선언된 컬럼 생략 시 default 값이 삽입됨.
```

```
SQL> insert into orders(order_num, member_id, prod_id)
```

```
values (4, 103, 10);
```

```
SQL> select *
```

```
from orders;
```

```
// insert 시 default 값이 선언된 컬럼에 수동으로 default 값 삽입
```

```
SQL> insert into orders
```

```
values (5, 108, 50, default);
```

```
SQL> select *
```

```
from orders;
```


// foreign key 제약조건이 선언된 컬럼 데이터 삽입1

```
SQL> insert into orders
```

```
values (6, 120, 30, default);
```

=> Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails

- FK 제약조건 : 다른 테이블의 특정 컬럼을 참조하는 제약조건
- 관계 : orders 테이블의 member_id 컬럼(자식 컬럼, FK 제약조건이 선언된 컬럼)
→ members 테이블의 member_id 컬럼(부모 컬럼)
- FK 제약조건이 선언된 컬럼(자식 컬럼)에는 반드시 부모 컬럼의 데이터 값 중 하나만 삽입/수정될 수 있다.
- 오류 원인 : members 테이블에는 120번 멤버는 존재하지 않으므로 주문을 할 수 없다.

[수정]

```
SQL> insert into orders
```

```
values (6, 103, 30, default);
```

```
SQL> select *
```

```
from orders;
```

// foreign key 제약조건이 선언된 컬럼 데이터 삽입2

```
SQL> insert into orders
```

```
values (7, 105, 80, default);
```

=> Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails

- 관계 : orders 테이블의 prod_id 컬럼(자식 컬럼, FK 제약조건이 선언된 컬럼)
→ products 테이블의 prod_id 컬럼(부모 컬럼)
- 오류 원인 : products 테이블에는 80번 상품이 존재하지 않으므로 해당 상품을 주문할 수 없다.

[수정]

```
SQL> insert into orders
```

```
values (7, 105, 60, default);
```

```
SQL> select *
```

```
from orders;
```

MySQL

② 추가 데이터 삽입하기

- 다음 3개의 행을 orders 테이블에 삽입하시오.
- 삽입 후 삽입된 데이터를 조회하시오.

order_num	member_id	prod_id	order_date
8	110	40	2021-12-30 10:30:45
9	107	30	default
10	101	60	now()

3. AUTO_INCREMENT 속성이 선언된 테이블에 데이터 삽입

1) stu20 테이블에 데이터 삽입 예제

① auto_increment 속성이 선언된 컬럼에 자동으로 데이터 삽입

// 기본값은 1부터 시작해서 1씩 증가하는 값을 반환함.

```
SQL> insert into stu20  
      values (null, '김온달', 28);
```

```
SQL> select *  
      from stu20;
```

```
SQL> insert into stu20  
      values (null, '이평강', 24);
```

```
SQL> select *  
      from stu20;
```

② auto_increment 입력값(시작값) 변경

// 다음 입력값이 100으로 변경됨.

```
SQL> alter table stu20 auto_increment=100;
```

```
SQL> insert into stu20  
      values (null, '최찬미', 29);
```

```
SQL> select *  
      from stu20;
```

// 증가값은 변경하지 않았으므로 기본인 1씩 증가하는 값이 반환됨.

```
SQL> insert into stu20  
      values (null, '김동희', 31);
```

```
SQL> select *  
      from stu20;
```

MySQL

③ auto_increment 증가값(증가 사이즈) 변경

```
// 다음 입력값으로 5 증가된 값이 반환됨.  
SQL> set @@auto_increment_increment=5;  
SQL> insert into stu20  
      values (null, '박혜경', 22);  
SQL> select *  
      from stu20;
```

```
SQL> insert into stu20  
      values (null, '문진원', 27);  
SQL> select *  
      from stu20;
```

10. 데이터 수정

1. 데이터 수정 : UPDATE

```
[문법] SQL> update 테이블명  
        set 컬럼명 = 값  
        [where 조건문];
```

```
[문법] SQL> update 테이블명  
        set 컬럼명1 = 값1, 컬럼명2 = 값2, ...  
        [where 조건문];
```

2. 데이터 수정 예제

1) products 테이블의 모든 상품 가격을 50 인상하시오.

```
SQL> update products  
      set price = price + 50;  
SQL> select *  
      from products;  
// where절 없이 update 구문 작성 시 테이블의 모든 행이 수정된다.
```

2) products 테이블의 TV 제품 가격을 30 인상하시오.

```
SQL> update products  
      set price = price + 30  
      where prod_name = 'TV';  
SQL> select *  
      from products;  
// where절 작성 시 특정 행이 수정된다.
```

3) members 테이블의 105번 사원 전화번호를 010-5050-5050으로 변경하시오.

```
SQL> update members  
      set phone = '010-5050-5050'  
      where member_id = 105;  
SQL> select *  
      from members;  
// where절 작성 시 특정 행이 수정된다.
```

MySQL

4) orders 테이블의 2번 주문의 주문자(member_id)를 120으로 변경하시오.

// foreign key 제약조건이 선언된 컬럼 데이터 수정

```
SQL> update orders
```

```
    set member_id = 120
```

```
    where order_num = 2;
```

=> Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails

- 오류 원인 : members 테이블에는 120번 멤버는 존재하지 않으므로 주문을 할 수 없다.
그래서 insert 뿐 아니라 update도 되지 않는다.

[수정]

```
SQL> update orders
```

```
    set member_id = 109
```

```
    where order_num = 2;
```

```
SQL> select *
```

```
    from orders;
```

// members 테이블에 109번 멤버는 존재하므로 update 가능하다.

11. 데이터 삭제

1. 데이터 삭제 : DELETE

```
[문법] SQL> delete from 테이블명  
[where 조건문];
```

2. 데이터 삭제 예제

1) stu20 테이블에서 나이가 25세 이하인 학생을 삭제하시오.

```
SQL> delete from stu20  
      where age <= 25;  
SQL> select *  
      from stu20;  
// where절 작성 시 특정 행이 삭제된다.
```

2) stu20 테이블의 모든 학생을 삭제하시오.

```
SQL> delete from stu20;  
SQL> select *  
      from stu20;  
// where절 없이 delete 구문 작성 시 테이블의 모든 행이 삭제된다.
```

3. 실수로 DML 작업 시 되돌리는 방법

1) DML 작업 후 저장하는 명령어

```
SQL> commit;
```

2) DML 작업 후 취소하는 명령어

```
SQL> rollback;
```

3) MySQL Workbench 사용 시에는 설정 확인하기

① [Query] - [Auto-Commit Transactions] 가 체크되어 있는 경우(기본)

- DML 발생 시 바로 자동 저장되므로 작업 취소를 할 수 없다.
- 장점 : 따로 저장할 필요가 없으므로 편함.
- 단점 : 자동 저장되므로 작업 후 되돌릴 수 없음.

MySQL

- ② [Query] - [Auto-Commit Transactions] 가 체크 해제되어 있는 경우
- DML 작업 후 commit 또는 rollback을 개발자가 결정해야 한다.
 - 장점 : 작업 후 select 구문을 통한 충분한 미리보기 후 저장 또는 취소를 결정할 수 있음.
그러므로 작업을 실수한 경우 되돌릴 수 있음.
 - 단점 : 작업 후 commit 또는 rollback으로 마무리를 반드시 해줘야 함.

12. 테이블의 데이터 조회

1. 데이터 조회

```
[문법] SQL> select *  
        from 테이블명  
        [ where 조건문 ];
```

```
[문법] SQL> select 컬럼1, 컬럼2, 컬럼3, ...  
        from 테이블명  
        [ where 조건문 ];
```

1) 테이블의 모든 컬럼, 모든 행 조회하기

```
// members 테이블의 모든 데이터 조회  
SQL> select *  
      from members;  
  
// products 테이블의 모든 데이터 조회  
SQL> select *  
      from products;  
  
// orders 테이블의 모든 데이터 조회  
SQL> select *  
      from orders;
```

2) 테이블의 특정 컬럼 조회하기

- select절에 출력하고 싶은 컬럼을 출력하고 싶은 순서대로 나열하면 됨.

```
// members 테이블의 특정 컬럼 조회  
SQL> select member_id, member_name, phone  
      from members;  
  
// products 테이블의 특정 컬럼 조회  
SQL> select company, prod_name, price  
      from products;  
  
// orders 테이블의 특정 컬럼 조회  
SQL> select order_num, order_date  
      from orders;
```

MySQL

3) select 구문에 산술연산자 활용하기

- select 절에 산술 연산자(+, -, *, %)를 활용한 산술식 작성 가능함.
- 컬럼 alias : 컬럼 제목을 컬럼명이나 산술식이 아닌 원하는 제목으로 출력할 경우 사용

```
[문법] SQL> select 컬럼1, 컬럼2 [AS] 'alias'
        from 테이블명
        [where 조건문];
```

```
SQL> select prod_id, prod_name, price + 50 as '인상 예정 가격'
        from products;
```

```
SQL> select prod_id, prod_name, price as '원가', price * 0.9 as '10% 세일 가격'
        from products;
```

2. 데이터 조회 시 WHERE 작성 방법

- 테이블의 모든 행이 아닌 특정 행의 출력을 원할 경우 where절에 조건문을 작성해야 함.
- where절의 조건문은 좌변 비교연산자 우변 순서대로 작성한다.
- 좌변 : 비교할 컬럼명
- 우변 : 비교할 값(숫자, 문자, 날짜 등)
- 비교연산자

비교연산자	내용	비교연산자	내용
=	같다	<> (또는 !=)	같지 않다
>	크다	<	작다
>=	크거나 같다	<=	작거나 같다
between A and B	A 이상 B 이하	not between A and B	A 미만 B 초과
like	작성된 패턴이 일치하는 값 검색	not like	작성된 패턴과 일치하지 않는 값 검색
is null	null 값 검색	is not null	null이 아닌 값 검색

1) where절을 활용한 데이터 검색 - 단일행 비교연산자(=, >, >=, <, <=, <>)

```
// members 테이블에서 member_id가 105번인 회원 정보만 조회
SQL> select *
        from members
        where member_id = 105;
```

```
// members 테이블에서 이름이 '홍길동'인 회원만 조회
SQL> select *
      from members
      where member_name = '홍길동';

// members 테이블에서 '회사원'이 아닌 회원만 조회
SQL> select *
      from members
      where job <> '회사원';

// products 테이블에서 가격이 300 이상인 제품의 이름과 가격 조회
SQL> select prod_name, price
      from products
      where price >= 300;

// members 테이블에서 생년월일이 1990년 이전인 회원의 이름, 생년월일, 전화번호 조회
SQL> select *
      from members
      where birth < '1990-01-01';
```

2) where절을 활용한 데이터 검색 – BETWEEN 비교연산자

- A 이상 B 이하의 값을 찾아주는 비교연산자로 범위 검색 시 사용함.
- A → 하한 값, B → 상한 값 작성해야함.
- 모든 데이터 타입에 활용 가능함.

```
[문법] SQL> select 컬럼1, 컬럼2, 컬럼3, ...
      from 테이블명
      where 좌변 [not] between A and B;
```

```
// between A and B 비교연산자 예제
```

```
SQL> select prod_name, price
      from products
      where price between 300 and 500;

SQL> select *
      from members
      where birth between '1990-01-01' and '1991-12-31';

SQL> select *
```

```
from members
where member_name between '김민수' and '이영수';
```

// not between A and B 비교연산자 예제
// A 미만 B 초과 값 찾아주는 비교연산자

```
SQL> select prod_name, price
      from products
      where price not between 300 and 500;
```

```
SQL> select *
      from members
      where birth not between '1990-01-01' and '1991-12-31';
```

```
SQL> select *
      from members
      where member_name not between '김민수' and '이영수';
```

3) where절을 활용한 데이터 검색 – LIKE 비교연산자

- 패턴 일치 여부를 비교해 주는 연산자로 값의 일부분만 아는 경우 사용됨.
- like 비교연산자에 사용되는 기호

%	문자가 안와도 되고 여러개 올 수 있음을 나타내는 기호
_	반드시 문자가 하나 와야됨을 나타내는 기호

[문법] SQL> select 컬럼1, 컬럼2, 컬럼3, ...
 from 테이블명
 where 좌변 [not] like '패턴';

```
// like 비교연산자 예제
SQL> select *
      from members
      where address like '부산%';

SQL> select member_id, member_name, birth, phone
      from members
      where member_name like '%영%';

SQL> select member_id, member_name, birth, phone
```

```
from members
where member_name like '_영%';
```

```
SQL> select *
      from products
      where make_date like '2021%';
```

// not like 비교연산자 예제

```
SQL> select *
      from members
      where address not like '서울%';
```

```
SQL> select *
      from members
      where birth not like '19__%';
```

- 4) where절을 활용한 데이터 검색 – IS NULL 비교연산자
- 값이 null 인지를 비교해주는 연산자

[문법] SQL> select 컬럼1, 컬럼2, 컬럼3, ...
 from 테이블명
 where 좌변 is [not] null;

// is null 비교연산자 예제

```
SQL> select *
      from members
      where phone is null;
```

```
SQL> select *
      from members
      where job is null;
```

```
SQL> select *
      from products
      where make_date is null;
```

// is not null 비교연산자 예제

```
SQL> select *  
      from members  
      where address is not null;
```

3. WHERE절에 여러 조건문 작성 방법

- where절에 조건문을 여러 개 작성할 경우 AND, OR 논리연산자로 나열한다.
- where절에 AND와 OR 논리연산자가 함께 사용되어 있는 경우 우선순위가 높은 연산자는 AND 연산자이다.
- 만약, 우선순위를 지정하고 싶은 경우에는 괄호를 사용한다.

```
[문법] SQL> select 컬럼1, 컬럼2, 컬럼3, ...  
      from 테이블명  
      where 조건문1 and 조건문2;
```

```
SQL> select 컬럼1, 컬럼2, 컬럼3, ...  
      from 테이블명  
      where 조건문1 or 조건문2;
```

// where절에 조건문이 여러 개 작성되는 예제

```
SQL> select prod_name, price  
      from products  
      where price >= 300 and price <= 500;  
(==)
```

```
SQL> select prod_name, price  
      from products  
      where price between 300 and 500;
```

```
SQL> select *  
      from members  
      where birth < '1990-01-01' or birth > '1991-12-31';  
(==)
```

```
SQL> select *  
      from members  
      where birth not between '1990-01-01' and '1991-12-31';
```

```
SQL> select *  
      from members  
      where address like '부산%' and job = '학생';
```

```
SQL> select *
```

```
from products
where company = 'LG' and price <=300;
```

// where절에 and와 or가 함께 사용된 경우 작업 순서 비교하기

[예제1] and 연산자가 or 연산자보다 우선순위가 높다.

```
SQL> select *
      from products
      where company = 'LG' or company = '삼성' and price <= 300;
```

[예제2] or 연산을 먼저하고 싶으면 괄호로 묶어 준다.

```
SQL> select *
      from products
      where (company = 'LG' or company = '삼성') and price <= 300;
```


13. 테이블의 데이터 조회 시 정렬하기

1. 데이터 정렬

```
[문법] SQL> select *  
          from 테이블명  
          [ where 조건문 ]  
          [ order by 컬럼명 asc];  
// 오름차순 정렬 : asc (생략 가능)
```

```
[문법] SQL> select *  
          from 테이블명  
          [ where 조건문 ]  
          [ order by 컬럼명 desc];  
// 내림차순 정렬 : desc
```

// 숫자 컬럼을 기준으로 정렬된 결과 출력하기

```
SQL> select *  
      from products  
      order by price asc;
```

```
SQL> select *  
      from products  
      order by price desc;
```

```
SQL> select *  
      from orders  
      order by member_id;
```

// 날짜 컬럼을 기준으로 정렬된 결과 출력하기

// 오름차순, 내림차순 정렬 시 null값의 출력 순서 확인하기

```
SQL> select *  
      from products  
      order by make_date;
```

```
SQL> select *  
      from products  
      order by make_date desc;
```

```
SQL> select *  
      from products  
      where make_date is not null  
      order by make_date desc;
```

```
SQL> select *  
      from orders  
      order by order_date;
```

// 문자 컬럼을 기준으로 정렬된 결과 출력하기

```
SQL> select *  
      from members  
      order by member_name;
```

```
SQL> select *  
      from members  
      order by member_name desc;
```

// 여러 컬럼을 기준으로 정렬된 결과 출력하기

```
SQL> select *  
      from products  
      order by company, price desc;
```

```
SQL> select *  
      from products  
      order by company desc, price desc;
```