

Chapter 09

데이터베이스 설치와 주요 SQL문

데이터베이스 설치와 설정

데이터베이스 설치

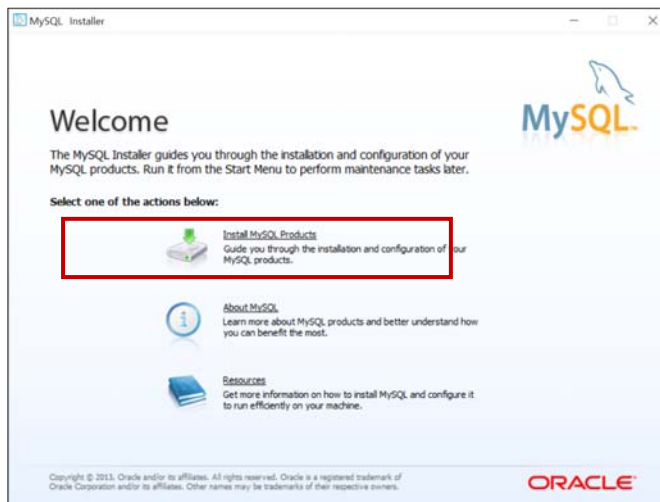
■ MySQL 설치

• 다운로드

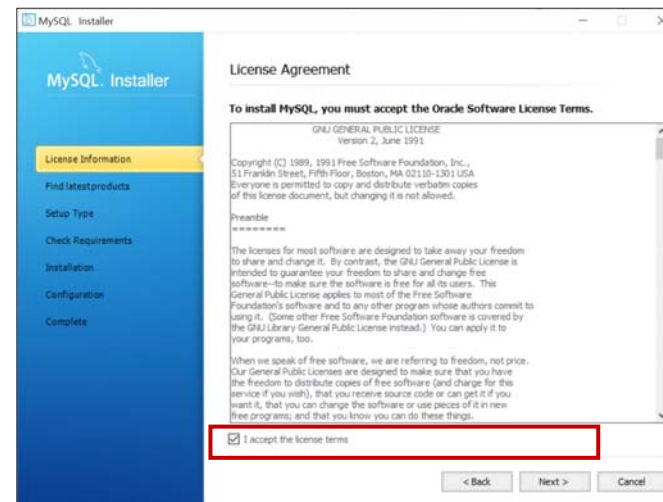
- ▶ <https://downloads.mysql.com/archives/installer/>에서 다운로드 가능
- ▶ e-campus에서 다운로드 가능
 - 2021년 4월 현재 8.0.22 버전이 출시되어 있으나, 실습에서는 Windows installer를 지원하는 5.6.14를 사용할 것임

• 설치

- ▶ mysql-installer-community-5.6.14.0.msi 파일을 실행

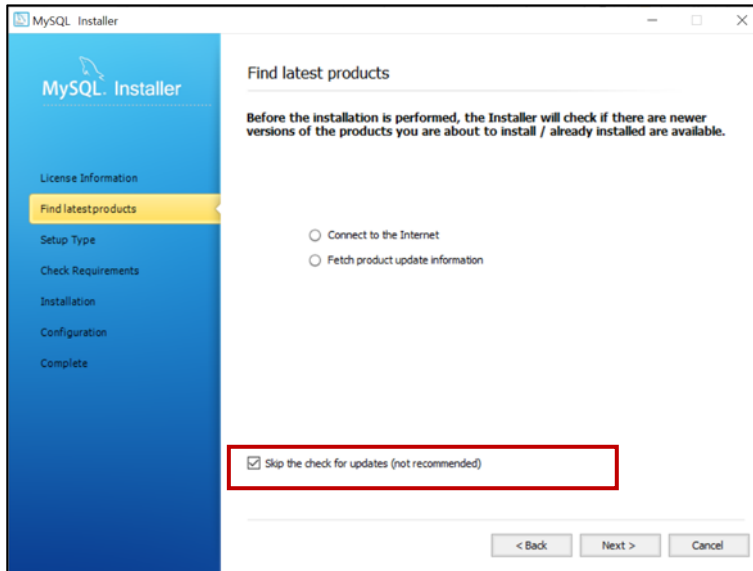


[Install MySQL Products]를 실행

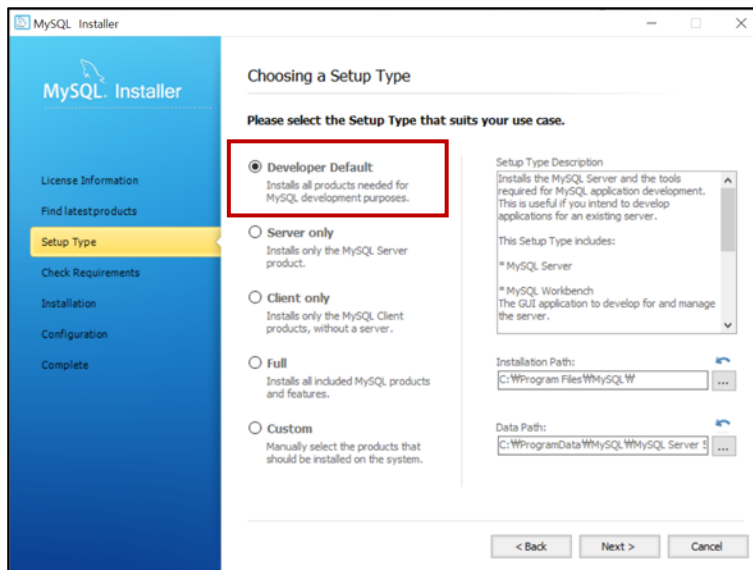


라이선스 동의

데이터베이스 설치

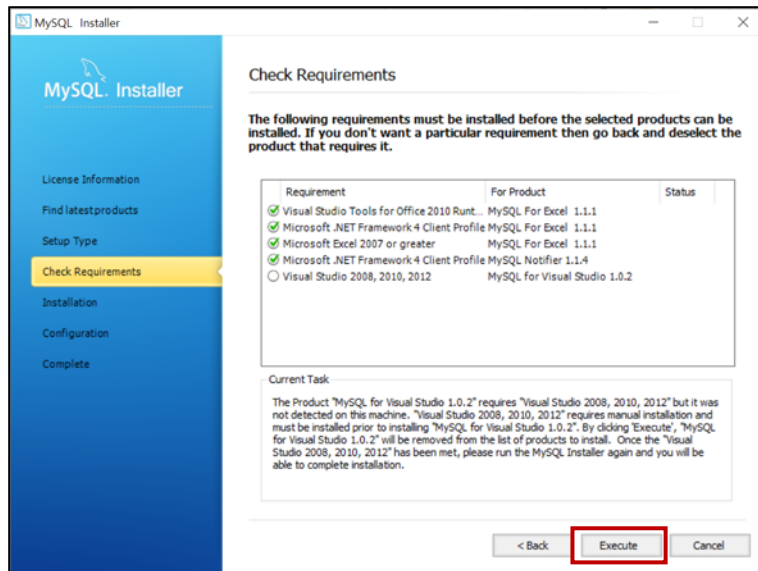


- 최신 업데이트 부분은 일단 skip



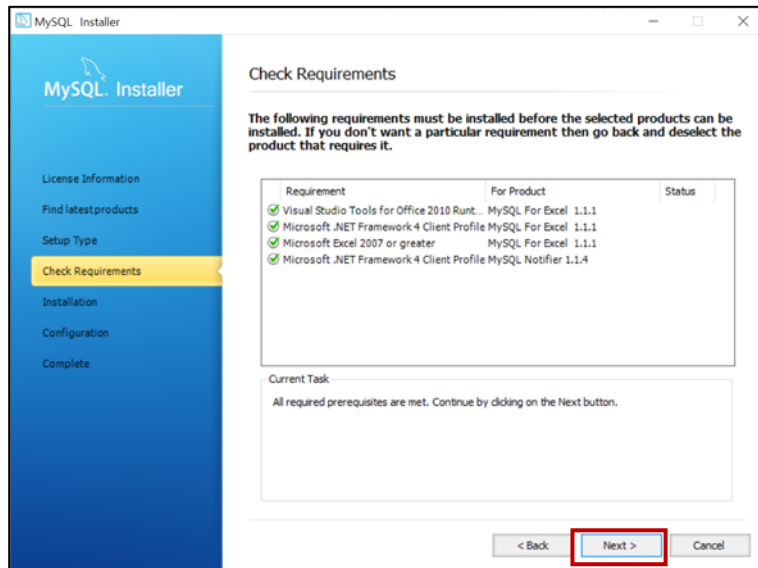
- 설치 유형은 [Developer Default]를 선택

데이터베이스 설치



• 각종 프로그램 및 라이브러리 설치

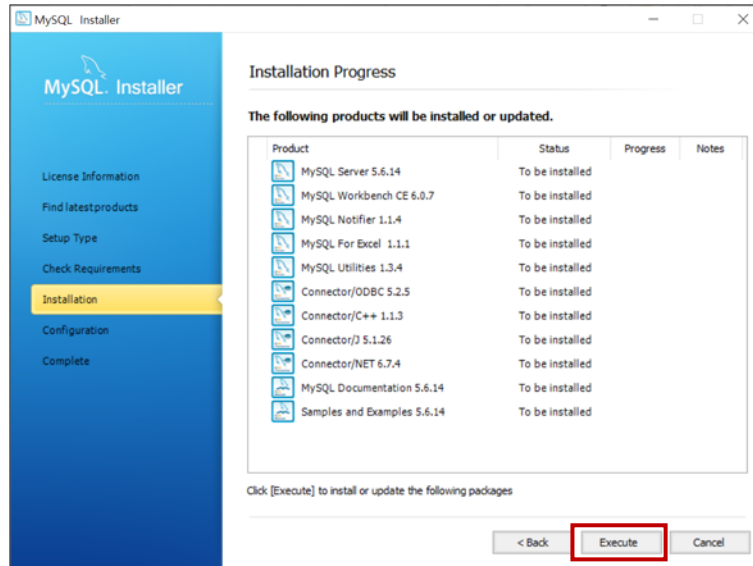
- ▶ [Execute] 버튼 클릭



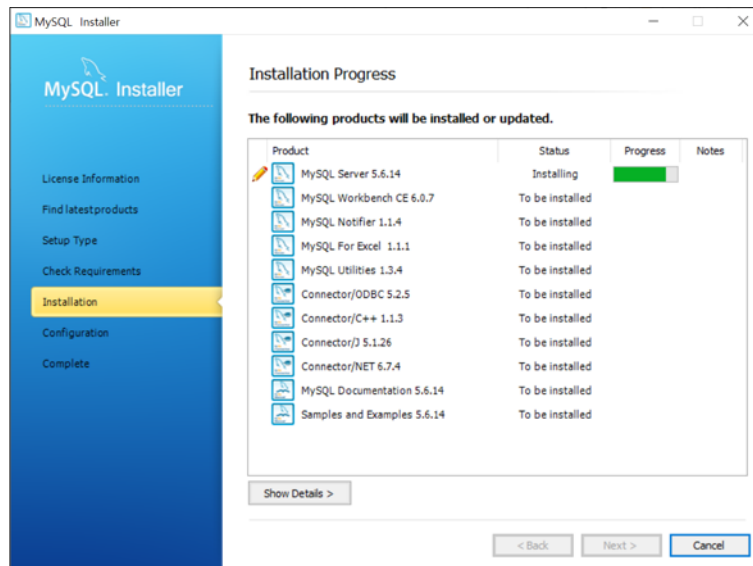
• 각종 프로그램 및 라이브러리 설치

- ▶ [Next] 버튼 클릭

데이터베이스 설치

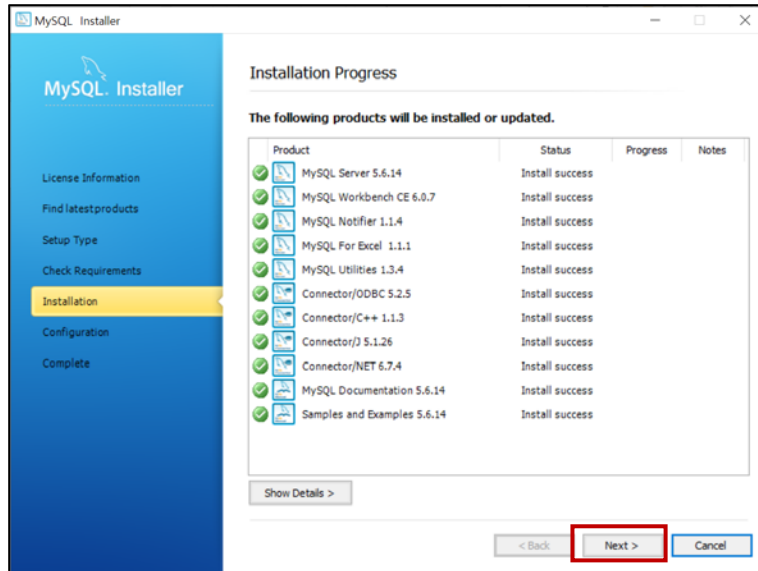


- 세부 설치 항목 확인
 - ▶ [Execute] 버튼 클릭



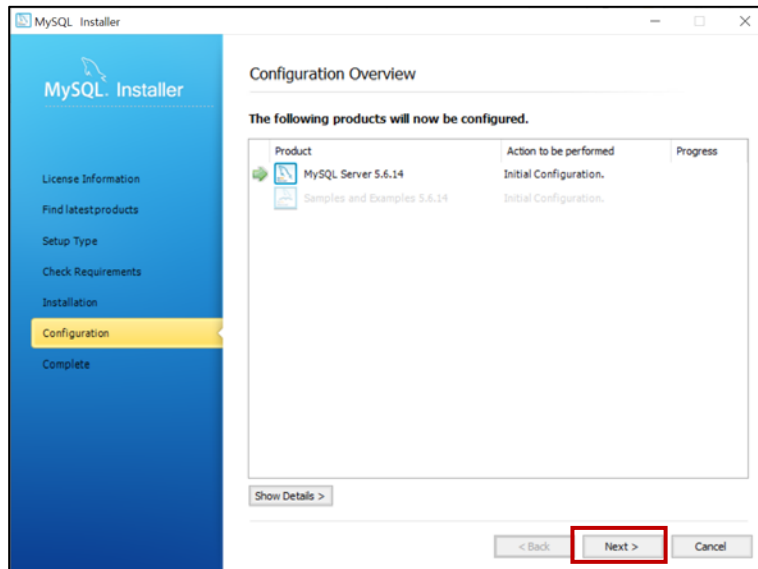
- 설치 시작

데이터베이스 설치



• 설치 완료

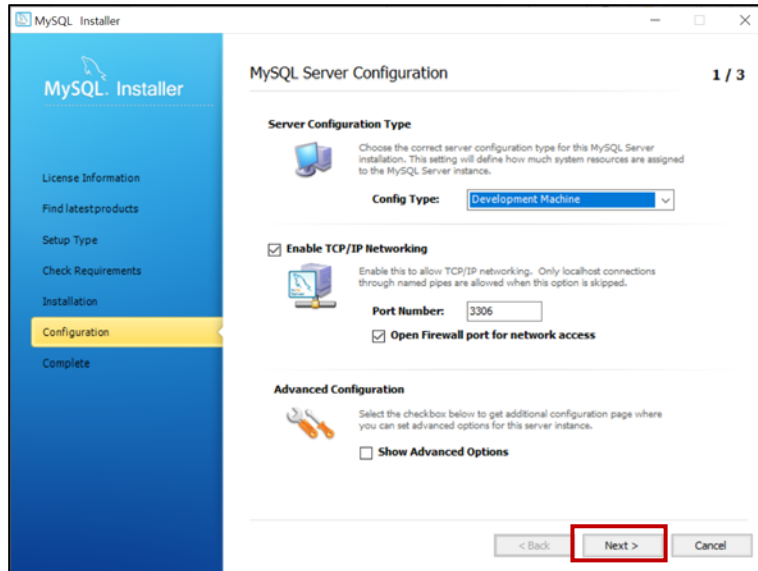
▶ [Next] 버튼 클릭



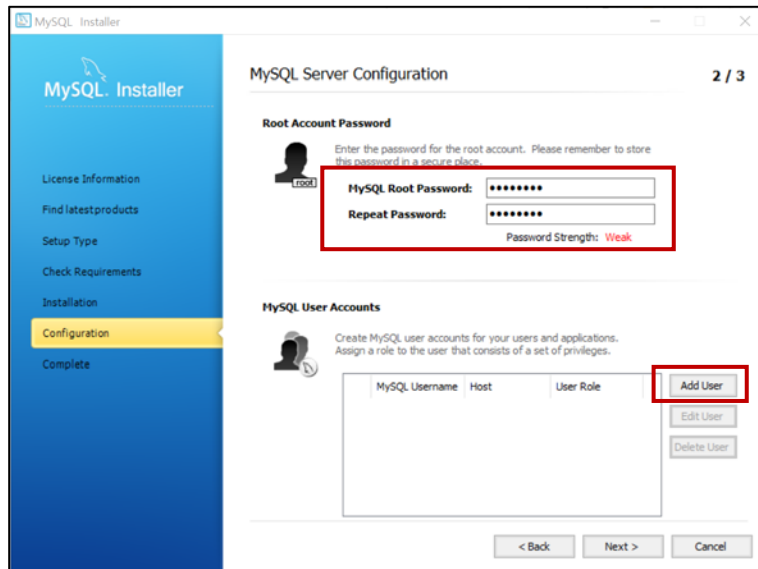
• 환경 설정 시작

▶ [Next] 버튼 클릭

데이터베이스 설치

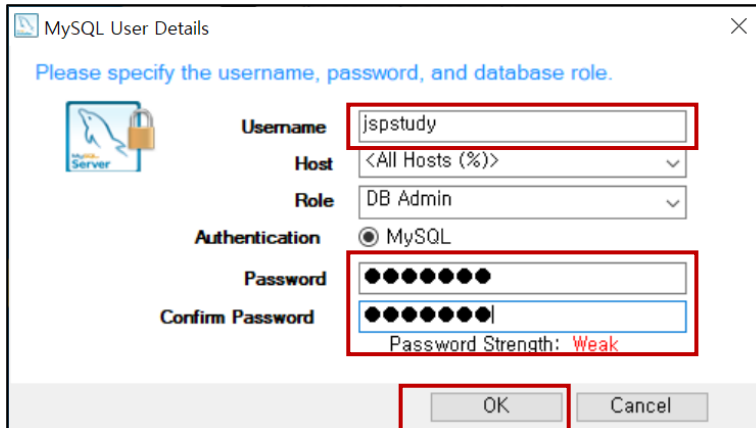


- Server Configuration Type
 - ▶ Development Machine 으로 지정
- Enable TCP/IP Networking
 - ▶ Check로 설정
 - ▶ 포트는 3306을 사용



- root 패스워드 지정
 - ▶ 임의의 패스워드를 지정 (분실하면 안됨)
- 사용자 계정 생성
 - ▶ [Add User] 버튼 클릭

데이터베이스 설치



MySQL User Details

Please specify the username, password, and database role.

Username: jspstudy

Host: <All Hosts (%)>

Role: DB Admin

Authentication: ☒ MySQL

Password: [Masked]

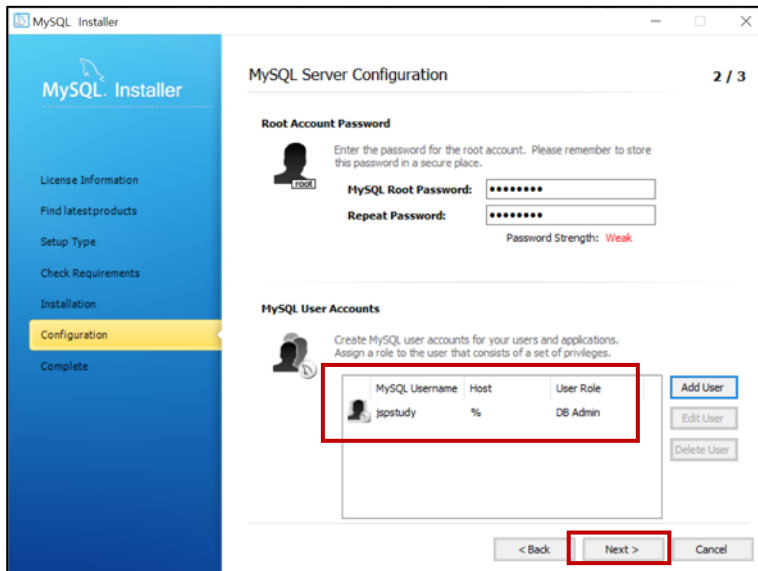
Confirm Password: [Masked]

Password Strength: Weak

OK Cancel

• 사용자 계정 생성

- ▶ Username
 - jspstudy로 지정
- ▶ Password
 - 임의의 패스워드를 지정
- ▶ 나머지 항목은 기본값을 사용
- ▶ [OK] 버튼 클릭



MySQL Installer

MySQL Server Configuration 2 / 3

Root Account Password

Enter the password for the root account. Please remember to store the password in a secure place.

MySQL Root Password: [Masked]

Repeat Password: [Masked]

Password Strength: Weak

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 Username	Host	User Role
jspstudy	%	DB Admin

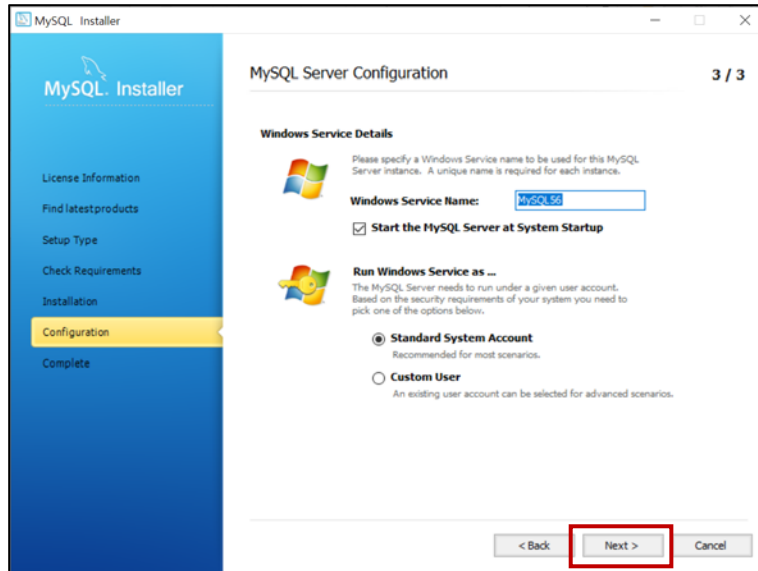
Add User Edit User Delete User

< Back Next > Cancel

• 사용자 계정 생성 확인

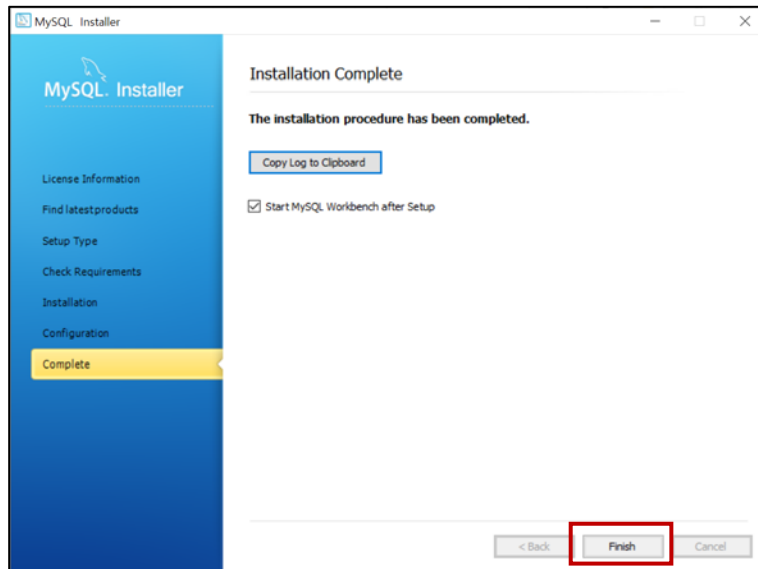
- ▶ [Next] 버튼 클릭

데이터베이스 설치



• 윈도우 서비스 설정

- ▶ 윈도우에 MySQL을 서비스 형태로 등록하기 위한 설정
- ▶ 기본 값을 사용



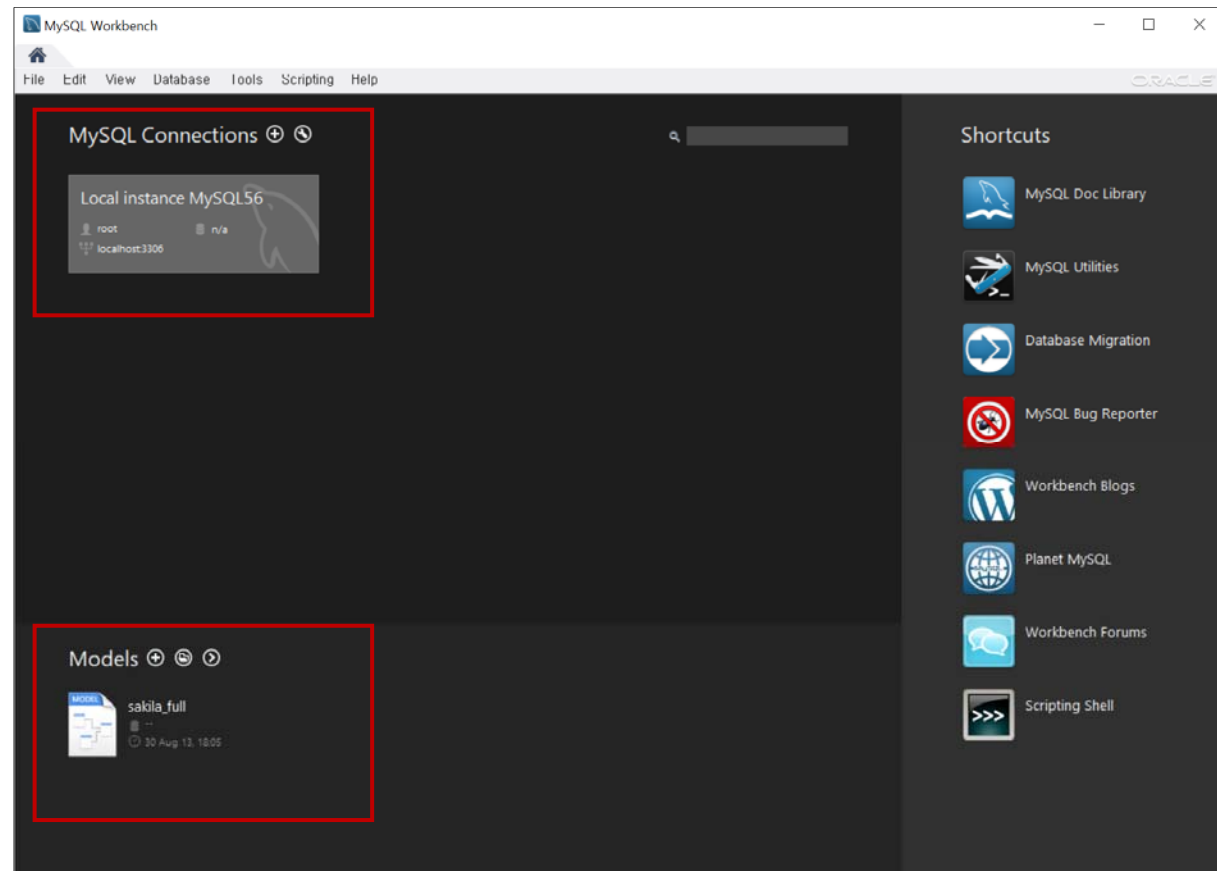
• 설치 완료

- ▶ [Finish] 버튼 클릭

MySQL Workbench 활용

■ MySQL 워크벤치

- 서버관리, 사용자 관리, 데이터 관리, 스키마 설정 등을 지원하는 GUI 환경의 도구
 - ▶ MySQL 워크 벤치의 관리자 모드는 [MySQL Connections]와 [Models] 로 구성됨
 - ▶ MySQL Connections
 - 서버에 연결 설정 부분
 - ▶ Models
 - 데이터 구조 설계
 - 다이어그램 작성



MySQL Workbench 활용

■ 사용자 개발 환경 구축

- 실습에서 사용할 스키마를 생성

- ▶ 스키마의 이름은 jspdb로 지정

- 스키마와 사용자의 연결

- ▶ MySQL 설치과정에서 jspstudy이라는 사용자를 생성했었음

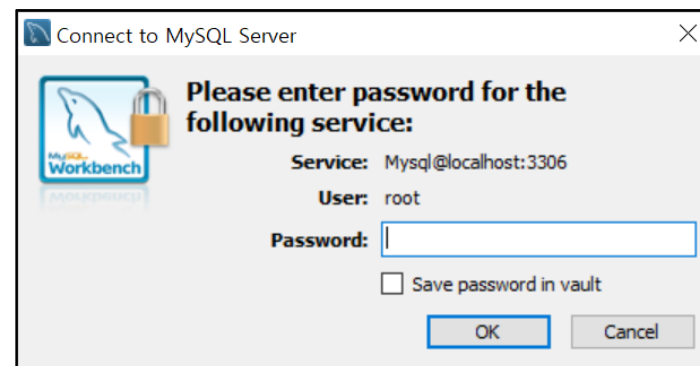
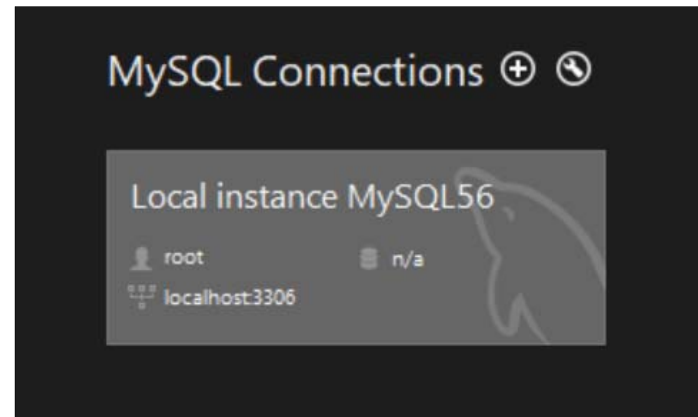
- ▶ jspstudy 사용자 계정과 jspdb 스키마를 연결

- jspstudy 계정이 jspdb 스키마를 사용할 수 있도록 모든 권한을 지정함

MySQL Workbench 활용

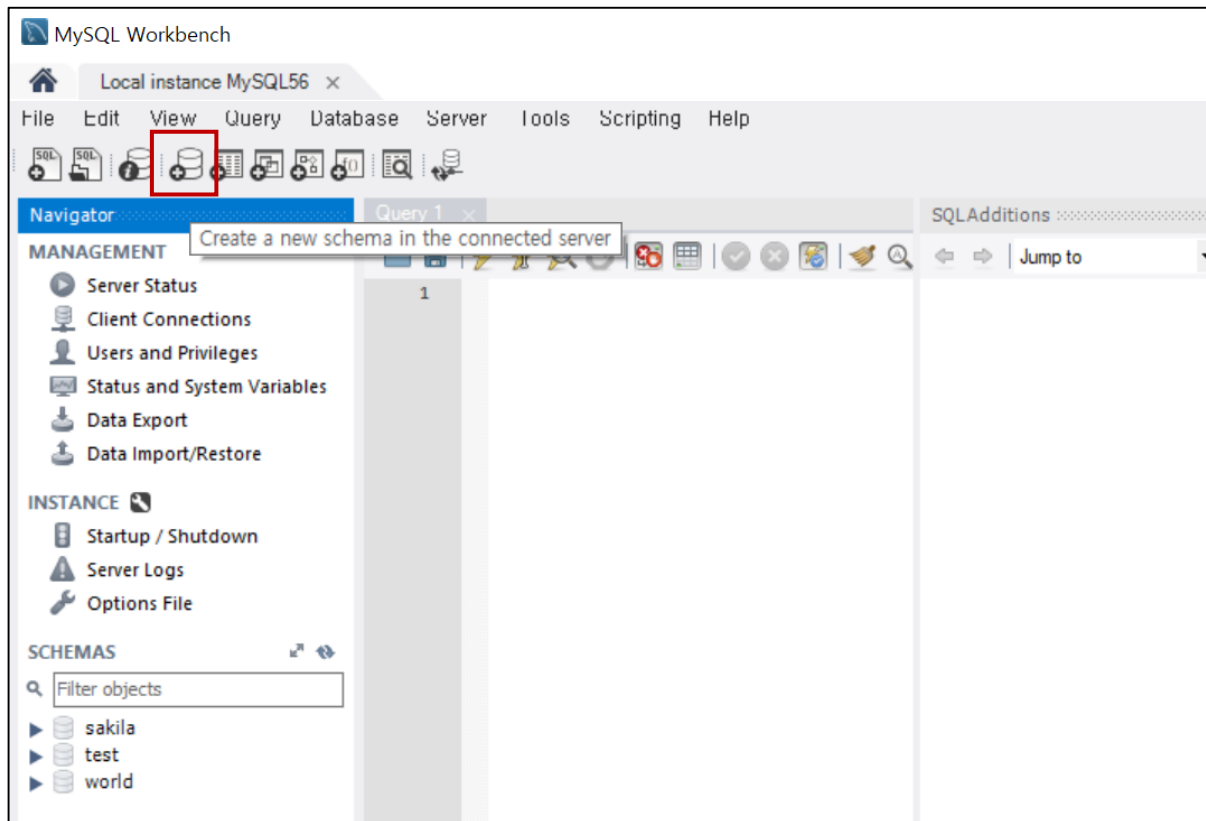
■ 스키마 생성

- [Local instance MySQL56] 실행
 - ▶ root의 패스워드 입력



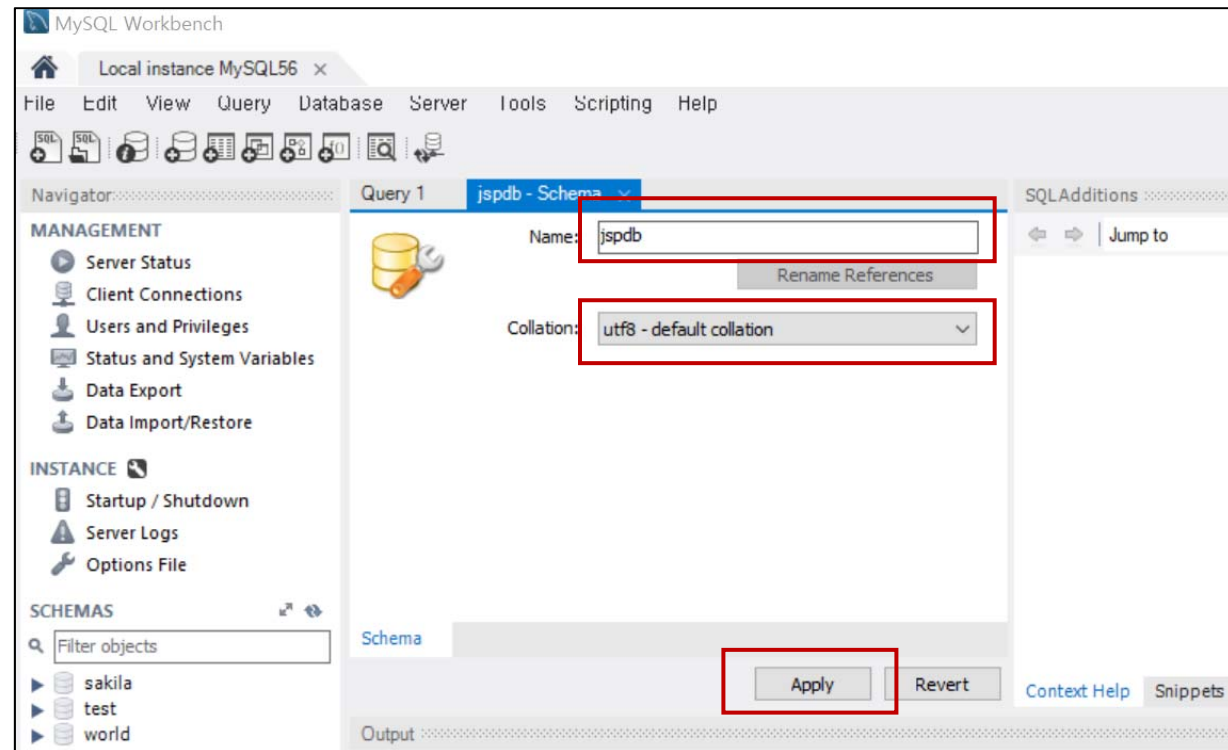
MySQL Workbench 활용

- 상단의 [Create a New Schema ...] 아이콘 실행

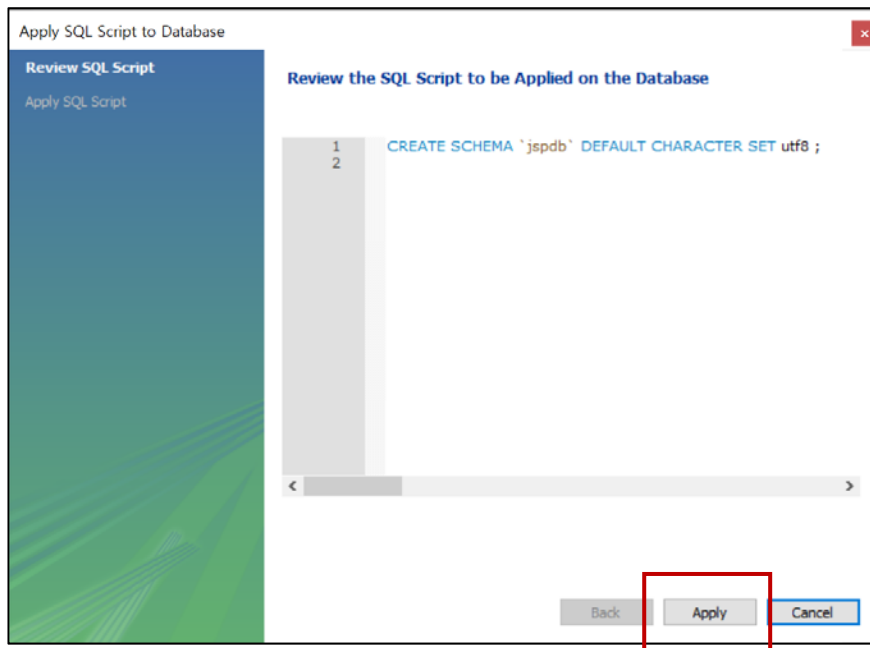


MySQL Workbench 활용

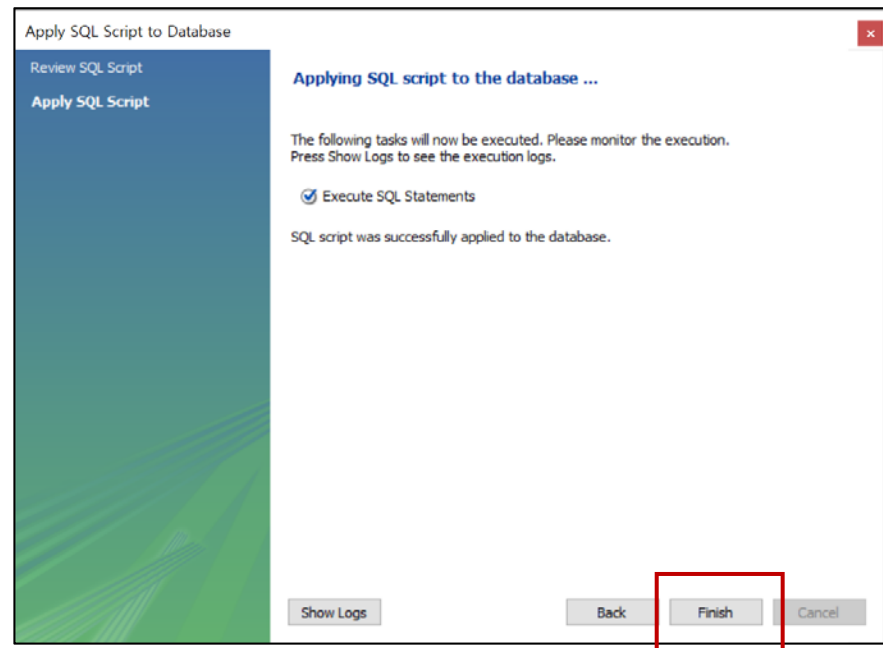
- 스키마 이름 지정
 - ▶ 스키마 이름을 jspdb로 지정
- 문자 집합 지정
 - ▶ Collation은 'utf8-default collation'으로 지정
- [Apply] 버튼 클릭



MySQL Workbench 활용



[Apply] 버튼 클릭

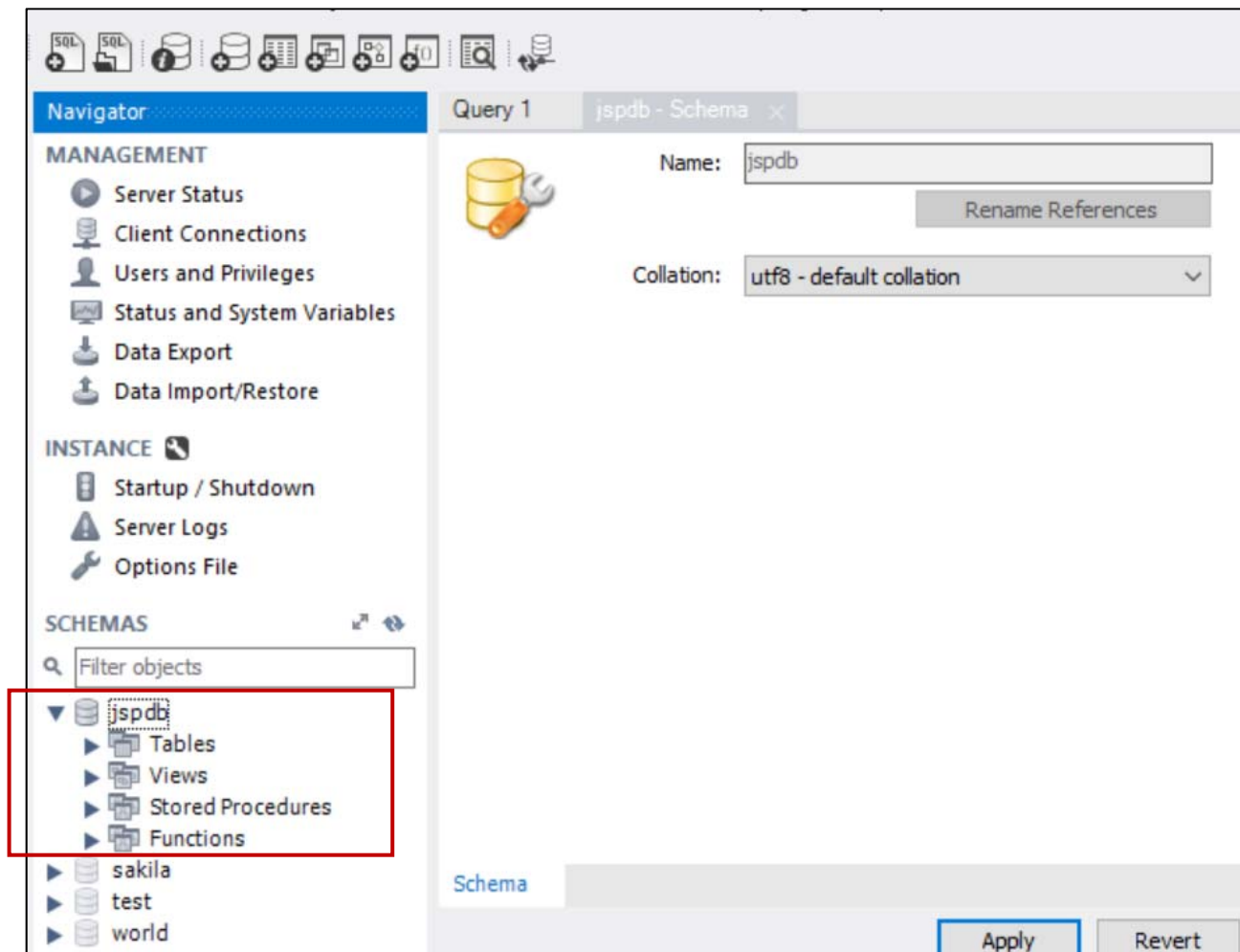


[Finish] 버튼 클릭

MySQL Workbench 활용

- 스키마 생성 확인

- jspdb 스키마 생성 확인

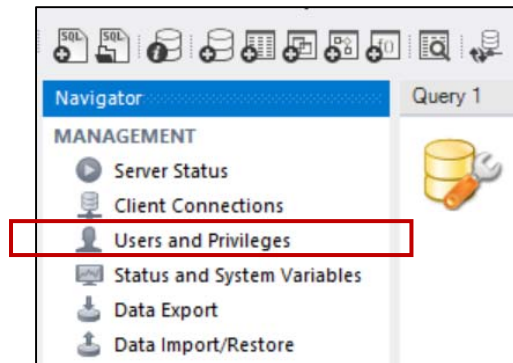


MySQL Workbench 활용

■ 스키마와 사용자 계정 연결

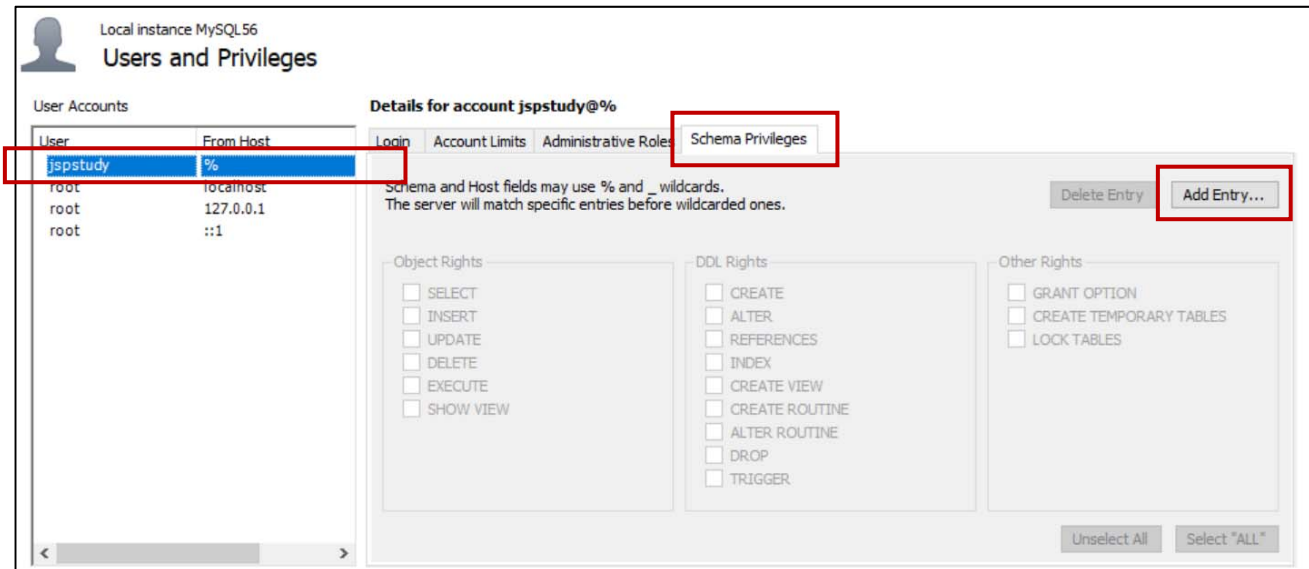
- Management의 [Users and Privileges] 실행

- ▶ root 패스워드 입력 필요



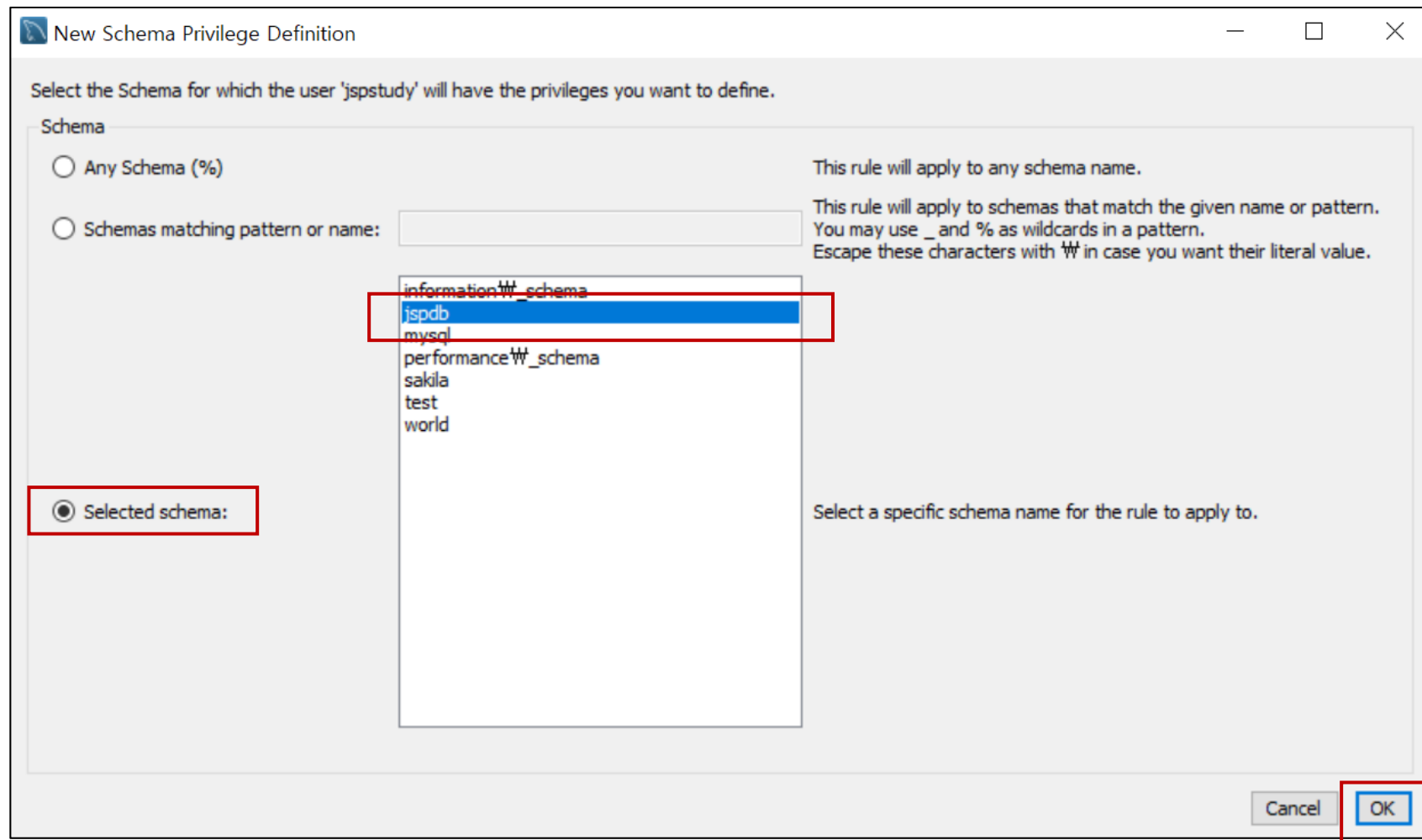
- jspstudy 사용자 계정을 선택한 다음 [Schema Privileges]를 선택

- ▶ [Add Entry...] 버튼을 클릭



MySQL Workbench 활용

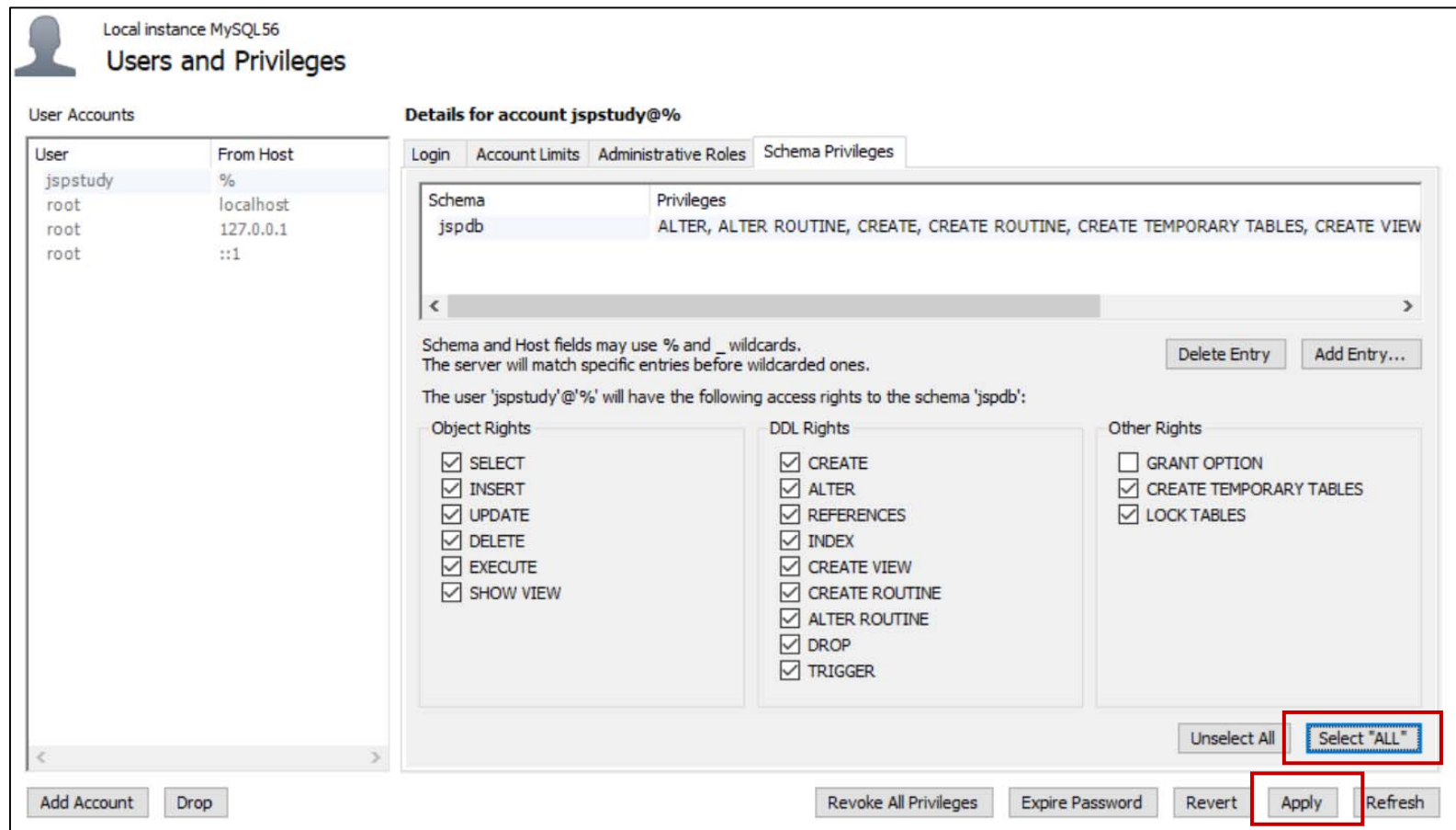
- [Select schema]를 선택한 다음 jspdb를 선택
 - ▶ [OK] 버튼 클릭



MySQL Workbench 활용

- 권한 지정

- ▶ [Select All]을 클릭하여 jspdb 스키마에 대한 모든 권한을 부여
- ▶ [Apply] 버튼 클릭



MySQL Workbench 활용

로컬 연결 생성

- [Home] 탭으로 이동

- ▶ MySQL Connections의 우측 <+> 기호 클릭

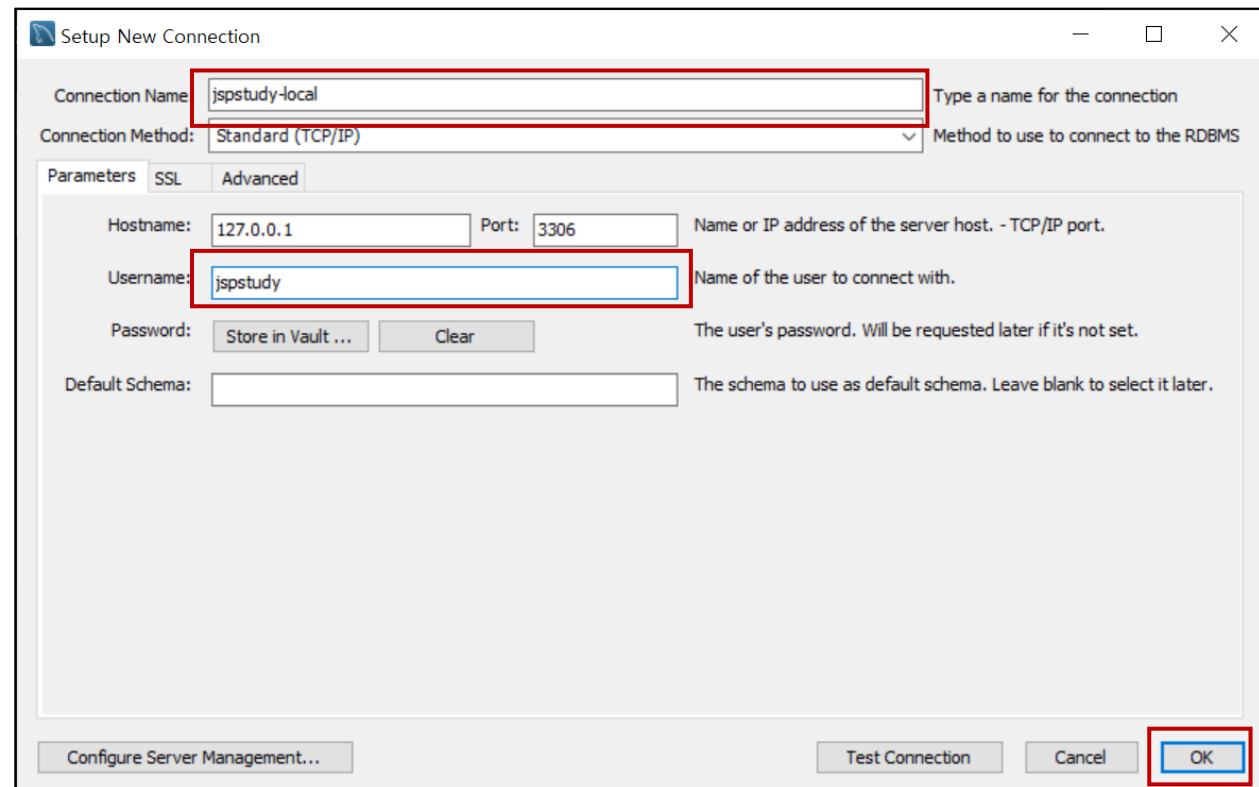
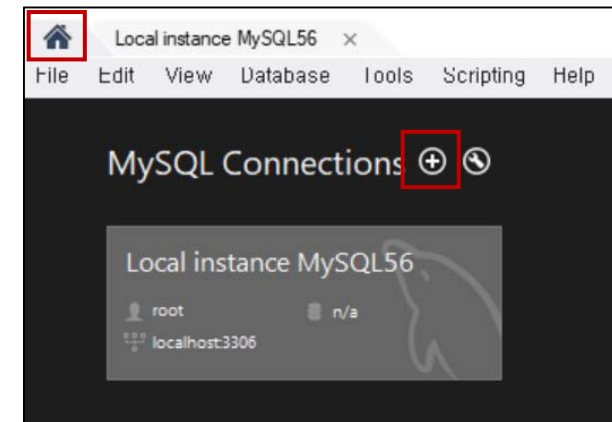
- ▶ Connection Name

- 임의의 이름을 지정
 - jspstudy-local을 지정

- ▶ Username

- 사용자 계정을 지정
 - jspstudy를 지정

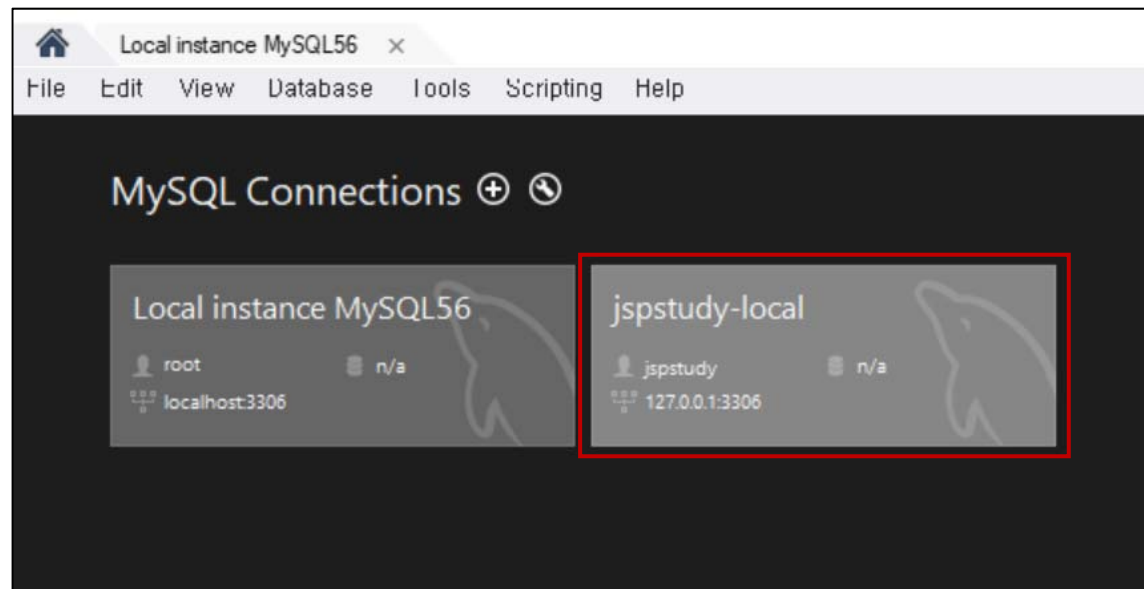
- ▶ [OK] 버튼 클릭



MySQL Workbench 활용

- 연결 생성 확인

- ▶ 연결 이름인 jspstudy-local을 클릭하여 jspstudy 계정으로 jspdb 스키마에 연결 가능
- ▶ jspbook 계정의 패스워드 필요



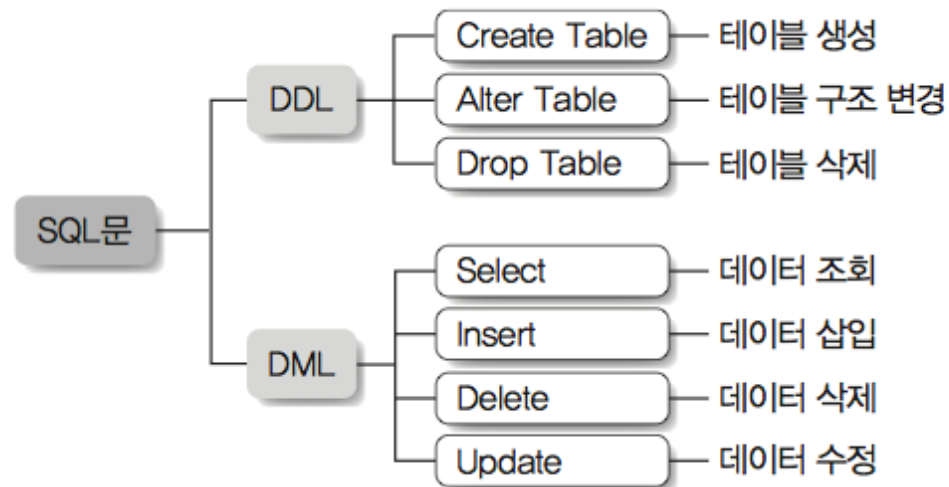
수업을 위한 기본적인 SQL 구문

SQL의 개요

■ SQL(Structured Query Language)

- 데이터베이스의 데이터를 관리하기 위한 쿼리 언어

- ▶ 데이터베이스 종류와 상관 없이 모든 데이터베이스는 SQL 을 통해서만 데이터 관리가 가능
- ▶ 기본적으로는 ANSI 표준이며 데이터베이스 회사별로 조금씩 차이가 있음



테이블 생성

■ 테이블의 생성

• 필드명

- ▶ 테이블은 하나 이상의 필드로 구성될 수 있음
 - 동일한 테이블을 구성하는 필드의 이름은 중복될 수 없음
- ▶ 각각의 필드에는 데이터형에 부합하는 데이터들이 저장되어야 함

• 속성

- ▶ 속성의 값으로 NULL을 지정하는 경우
 - 해당 필드에 데이터가 입력되지 않아도 된다는 것을 의미
 - 속성의 지정을 생략하면 기본 값으로 NULL이 지정
- ▶ 속성의 값으로 NOT NULL을 지정하는 경우
 - 해당 필드에 반드시 데이터가 입력되어야 함
 - NOT NULL 속성이 지정된 필드에 데이터가 입력되지 않으면 오류가 발생됨

• Primary Key

- ▶ 테이블 내의 데이터를 구별할 수 있는 식별자 필드를 지정
- ▶ 기본 키로 지정된 필드의 속성은 반드시 NOT NULL로 지정되어야 함

```
CREATE TABLE 테이블 이름 (  
    필드명 데이터형(크기) 속성,  
    ...  
    primary key(필드명)  
);
```

테이블 생성

■ 테이블 생성 실습

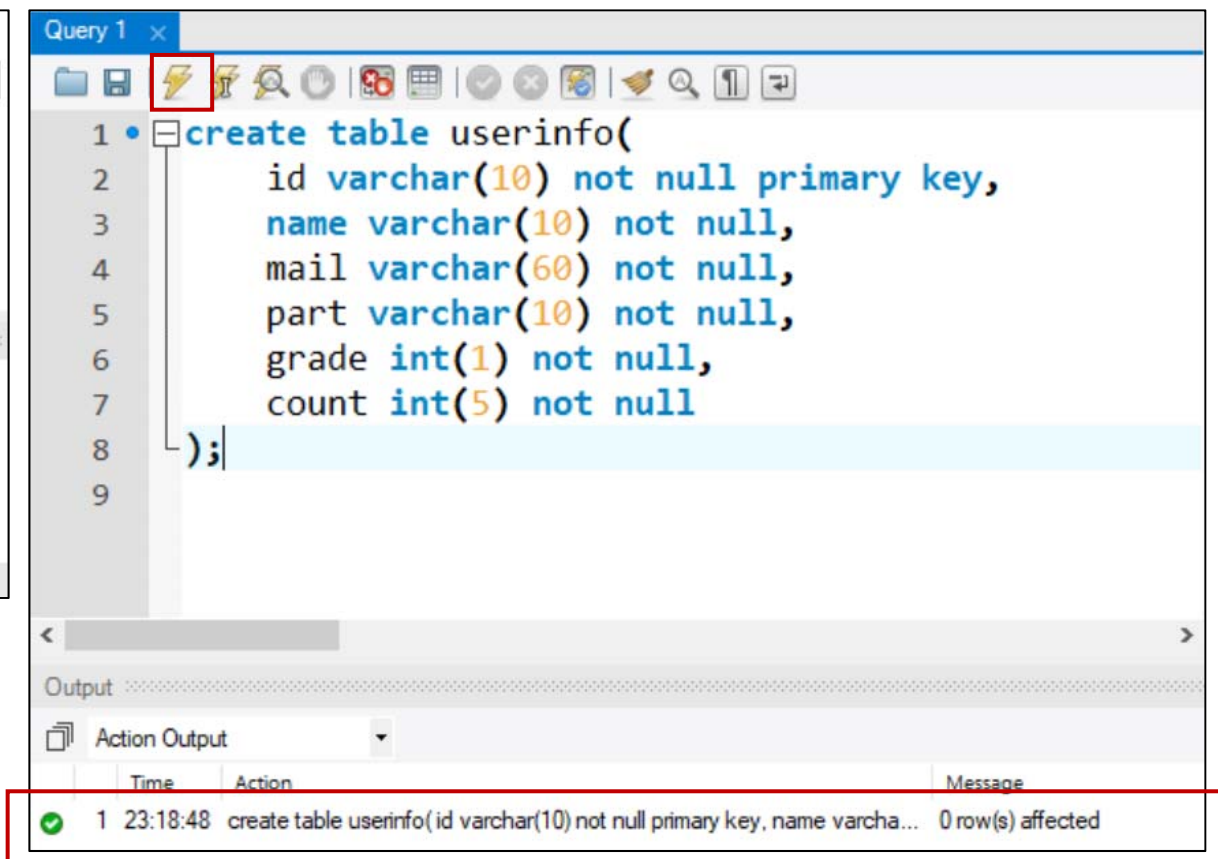
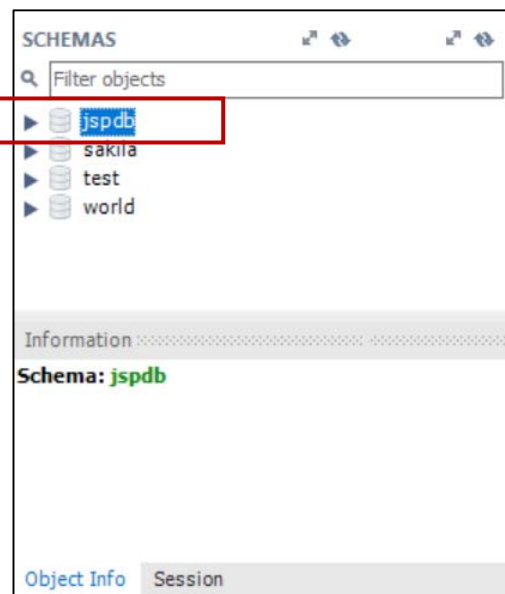
- 테이블 이름 : userinfo

컬럼명	데이터형	속성	비고
id	varchar(10)	not null, Primary key	아이디 입력 필드
name	varchar(10)	not null	성명 입력 필드
mail	varchar(60)	not null	전자우편 주소 입력 필드
part	varchar(10)	not null	근무 부서 입력 필드
grade	int(1)	not null	등급 입력 필드
count	int(5)	not null	판매량 입력 필드

테이블 생성

■ 질의 수행

- 질의문 작성 후 상단의 노란색 번개 모양 아이콘을 클릭하여 질의를 수행

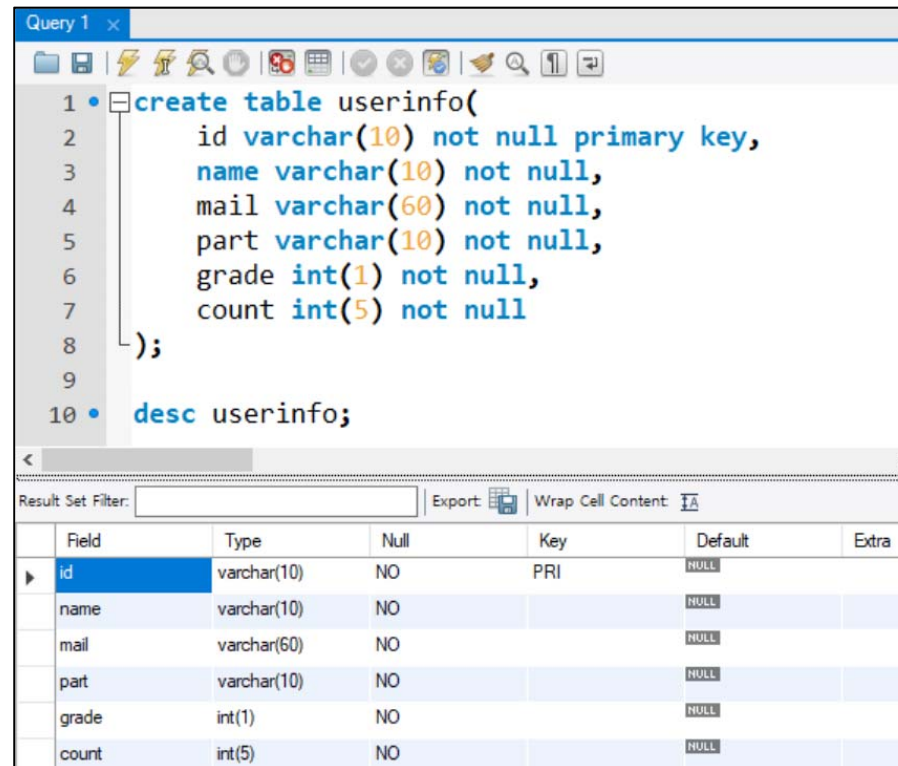


테이블 생성

■ 테이블의 구조 출력

desc 테이블이름;

- describe 명령어의 약어 표현으로 테이블의 구조를 출력함
 - ▶ desc 명령어 문장에 커서를 위치시키고 중간의 번개 모양(I 문자가 중앙에 있는) 아이콘 클릭



The screenshot shows a SQL IDE window titled 'Query 1'. The SQL script contains the following code:

```
1 • create table userinfo(  
2     id varchar(10) not null primary key,  
3     name varchar(10) not null,  
4     mail varchar(60) not null,  
5     part varchar(10) not null,  
6     grade int(1) not null,  
7     count int(5) not null  
8 );  
9  
10 • desc userinfo;
```

Below the script, the 'Result Set Filter' is empty, and the 'Export' button is visible. The 'Wrap Cell Content' checkbox is checked. The result set is displayed as a table with the following columns: Field, Type, Null, Key, Default, and Extra.

Field	Type	Null	Key	Default	Extra
id	varchar(10)	NO	PRI	NULL	
name	varchar(10)	NO		NULL	
mail	varchar(60)	NO		NULL	
part	varchar(10)	NO		NULL	
grade	int(1)	NO		NULL	
count	int(5)	NO		NULL	

테이블 구조 변경

■ 테이블의 구조 변경

alter table	테이블 이름	add	필드명	자료형 (속성)
		modify	필드명	자료형 (속성)
		change	기존필드명	새필드명 자료형 (속성)
		drop	기존필드명	

- **ADD**

- ▶ 새로운 필드를 추가 (새로 추가된 필드는 마지막 필드로 추가됨)

- **MODIFY**

- ▶ 기존 필드의 자료형과 속성을 변경

- **CHANGE**

- ▶ 기존 필드의 필드 이름, 데이터형, 속성을 모두 변경 가능

- **DROP**

- ▶ 특정 필드를 제거

테이블 구조 변경

- 테이블 구조 변경 실습 1

- ▶ mail 필드의 이름을 email로 변경하고, 데이터 형을 varchar(60)에서 varchar(50)으로 변경

The screenshot shows a SQL query editor window titled "Query 1". The query contains two lines of SQL code:

```
1 • alter table userinfo change mail email varchar(50);  
2 • desc userinfo;
```

Below the query editor, the "Result Set Filter" is empty. The "Export" button is visible, and the "Wrap Cell Content" option is set to "Off". The result set is displayed as a table with the following columns: Field, Type, Null, Key, Default, and Extra.

Field	Type	Null	Key	Default	Extra
id	varchar(10)	NO	PRI	NULL	
name	varchar(10)	NO		NULL	
email	varchar(50)	YES		NULL	
part	varchar(10)	NO		NULL	
grade	int(1)	NO		NULL	
count	int(5)	NO		NULL	

The "email" row is highlighted with a red box, indicating the result of the table structure change.

테이블 구조 변경

- 테이블 구조 변경 실습 2

- ▶ email 필드에 not null 속성을 추가

The screenshot shows a database query editor window titled 'Query 1'. It contains two SQL statements:

```
1 • alter table userinfo modify email varchar(50) not null;  
2 • desc userinfo;
```

Below the queries, there is a 'Result Set Filter' field and an 'Export' button. The main area displays the table structure of 'userinfo' as a table with the following columns: Field, Type, Null, Key, Default, and Extra.

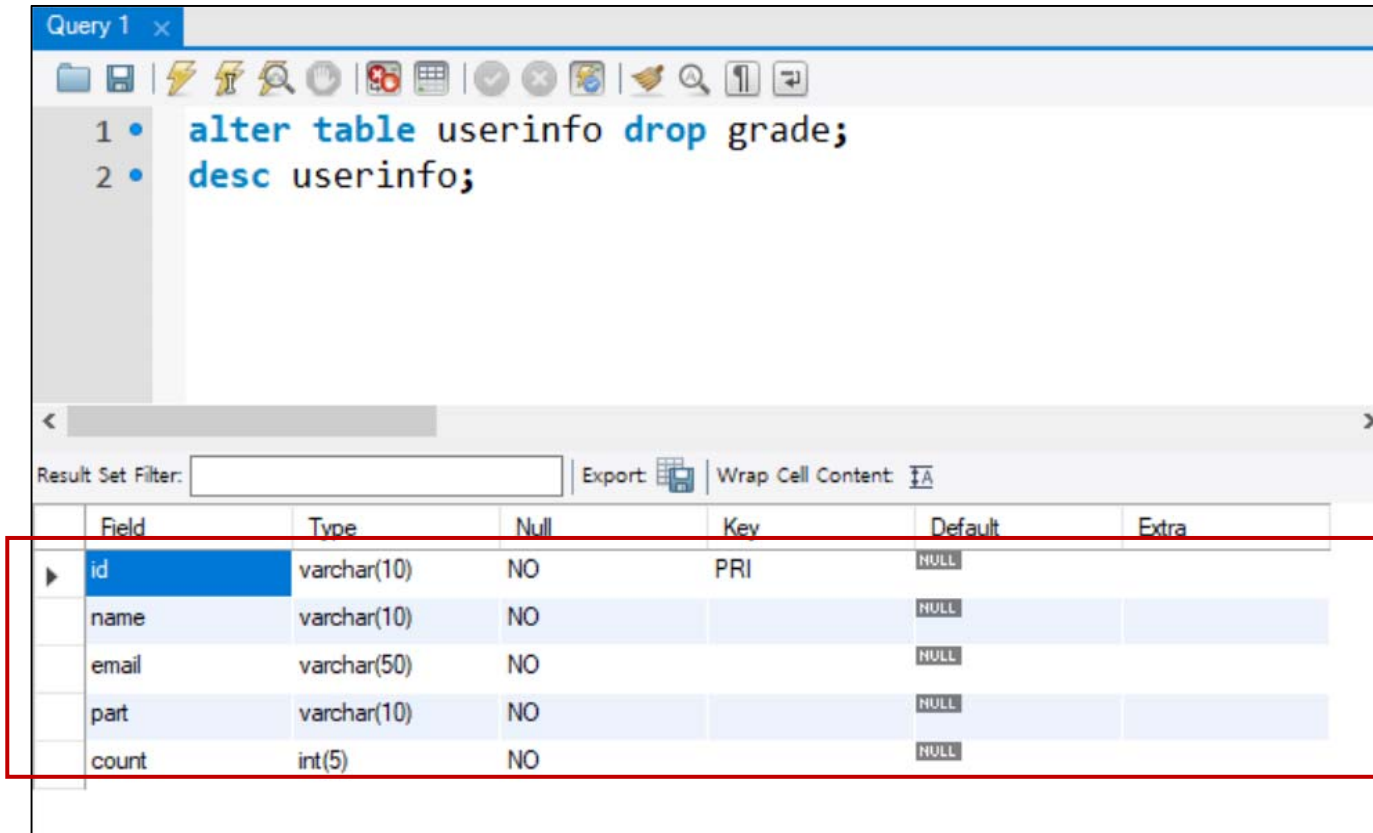
Field	Type	Null	Key	Default	Extra
id	varchar(10)	NO	PRI	NULL	
name	varchar(10)	NO		NULL	
email	varchar(50)	NO		NULL	
part	varchar(10)	NO		NULL	
grade	int(1)	NO		NULL	
count	int(5)	NO		NULL	

A red rectangle highlights the 'email' row in the table structure, indicating the successful update of its attributes to 'varchar(50) not null'.

테이블 구조 변경

- 테이블 구조 변경 실습 3

- ▶ grade 필드를 제거



The screenshot shows a database query editor window titled "Query 1". The SQL commands entered are:

```
1 • alter table userinfo drop grade;  
2 • desc userinfo;
```

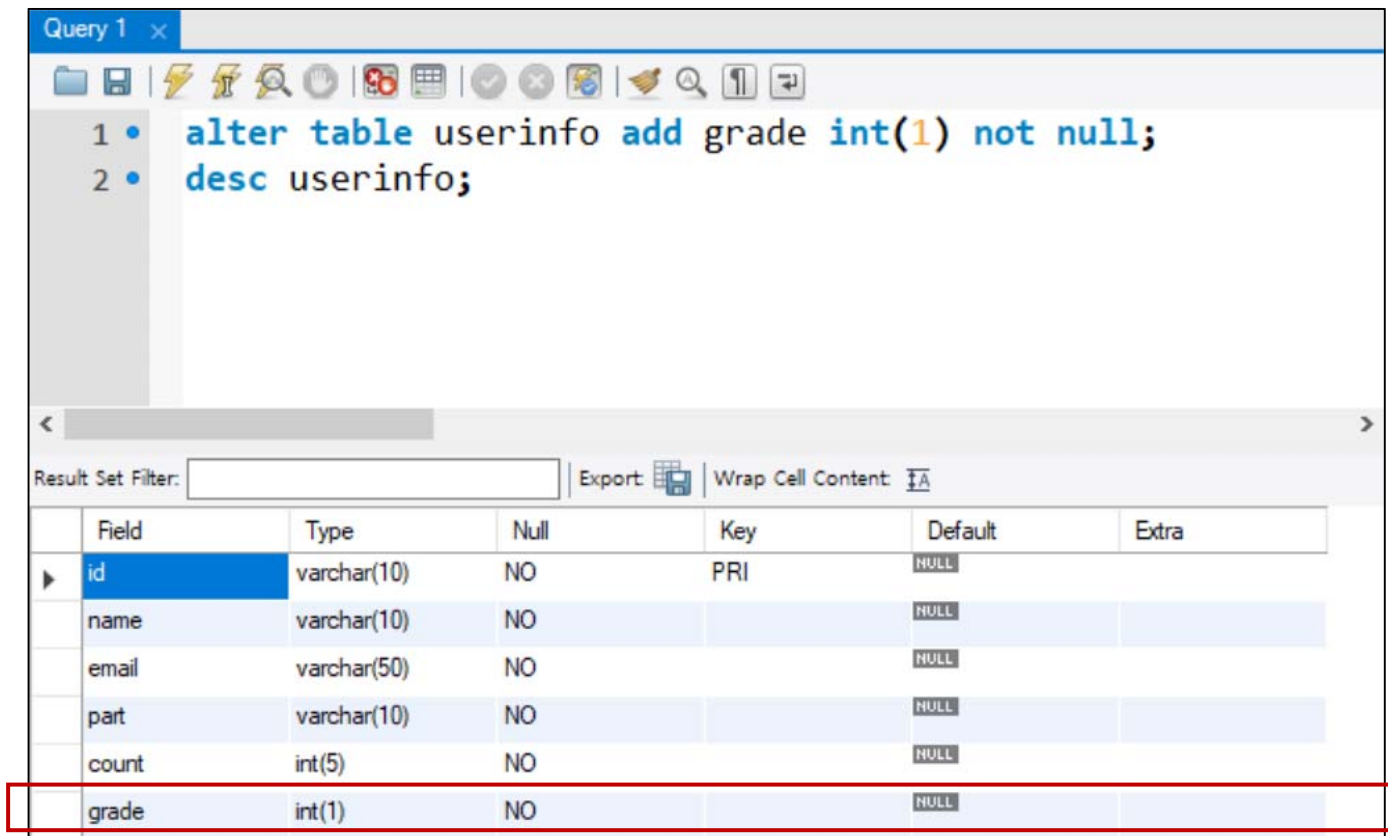
Below the query editor, the "Result Set Filter" is empty. The "Export" button is visible. The "Wrap Cell Content" option is set to "I A".

Field	Type	Null	Key	Default	Extra
id	varchar(10)	NO	PRI	NULL	
name	varchar(10)	NO		NULL	
email	varchar(50)	NO		NULL	
part	varchar(10)	NO		NULL	
count	int(5)	NO		NULL	

테이블 구조 변경



- 테이블 구조 변경 실습 4

- ▶ 자료형이 int(1)이고 not null 속성을 가진 grade 필드를 추가



Query 1 x

```
1 • alter table userinfo add grade int(1) not null;
2 • desc userinfo;
```

Result Set Filter: Export  Wrap Cell Content 

Field	Type	Null	Key	Default	Extra
id	varchar(10)	NO	PRI	NULL	
name	varchar(10)	NO		NULL	
email	varchar(50)	NO		NULL	
part	varchar(10)	NO		NULL	
count	int(5)	NO		NULL	
grade	int(1)	NO		NULL	

레코드의 삽입

■ 레코드의 삽입 : 일부 필드에만 값을 삽입

```
insert into 테이블이름 (필드1,...,필드n) values (값1,...,값n);
```

- 레코드를 구성하는 모든 필드가 아닌 일부 필드에만 데이터를 입력하고자 할 때 사용
 - ▶ (필드1, ..., 필드n)은 테이블 필드의 물리적인 순서와 일치시킬 필요는 없음
 - ▶ (필드1, ..., 필드n)과 (값1, ..., 값n)은 반드시 일대일 대응이 되어야 함
- 각 값들은 필드의 데이터형에 맞게 입력되어야 함

레코드의 삽입

■ 레코드의 삽입 : 모든 필드에만 값을 삽입

```
insert into 테이블이름 values (값1,...,값n);
```

- 테이블에 있는 모든 필드에 값을 입력할 때 사용
 - ▶ (값1, ..., 값n)은 테이블에 있는 필드의 물리적인 순서와 정확하게 일치해야 함
 - ▶ 데이터형도 일치해야 함
- 하나의 필드에만 값을 입력해도 무조건 하나의 새로운 레코드가 생성됨
- 이미 존재하는 레코드 중 비어있는 필드에 새로운 값을 삽입하려면 UPDATE라는 질의어를 사용

레코드의 삽입

- 레코드 삽입 실습

▶ 다음의 레코드를 userinfo 테이블에 삽입

id	name	email	part	count	grade
hhjung	정형호	hhjung@abc.co.kr	영업부	300	1
mkkang	강민구	mkkang@abc.co.kr	관리부	250	2
nylee	이나영	nylee@abc.co.kr	영업부	150	3
slcho	조승래	slcho@abc.co.kr	관리부	220	2
smcho	조수민	smcho@abc.co.kr	영업부	350	1

레코드의 삽입

Query 1 x

```
1 • insert into userinfo values ('hhjung', '정형호', 'hhjung@abc.co.kr', '영업부', 300, 1);
2 • insert into userinfo values ('mkkang', '강민구', 'mkkang@abc.co.kr', '관리부', 250, 2);
3 • insert into userinfo values ('nylee', '이나영', 'nylee@abc.co.kr', '영업부', 150, 3);
4 • insert into userinfo values ('slcho', '조승래', 'slcho@abc.co.kr', '관리부', 220, 2);
5 • insert into userinfo values ('smcho', '조수민', 'smcho@abc.co.kr', '영업부', 350, 1);
6
7 • select * from userinfo;
8
```

Result Set Filter: Edit Export/Import Wrap Cell Content

	id	name	email	part	count	grade
▶	hhjung	정형호	hhjung@abc.co.kr	영업부	300	1
	mkkang	강민구	mkkang@abc.co...	관리부	250	2
	nylee	이나영	nylee@abc.co.kr	영업부	150	3
	slcho	조승래	slcho@abc.co.kr	관리부	220	2
	smcho	조수민	smcho@abc.co.kr	영업부	350	1
*	NULL	NULL	NULL	NULL	NULL	NULL

레코드의 검색

■ 레코드의 검색

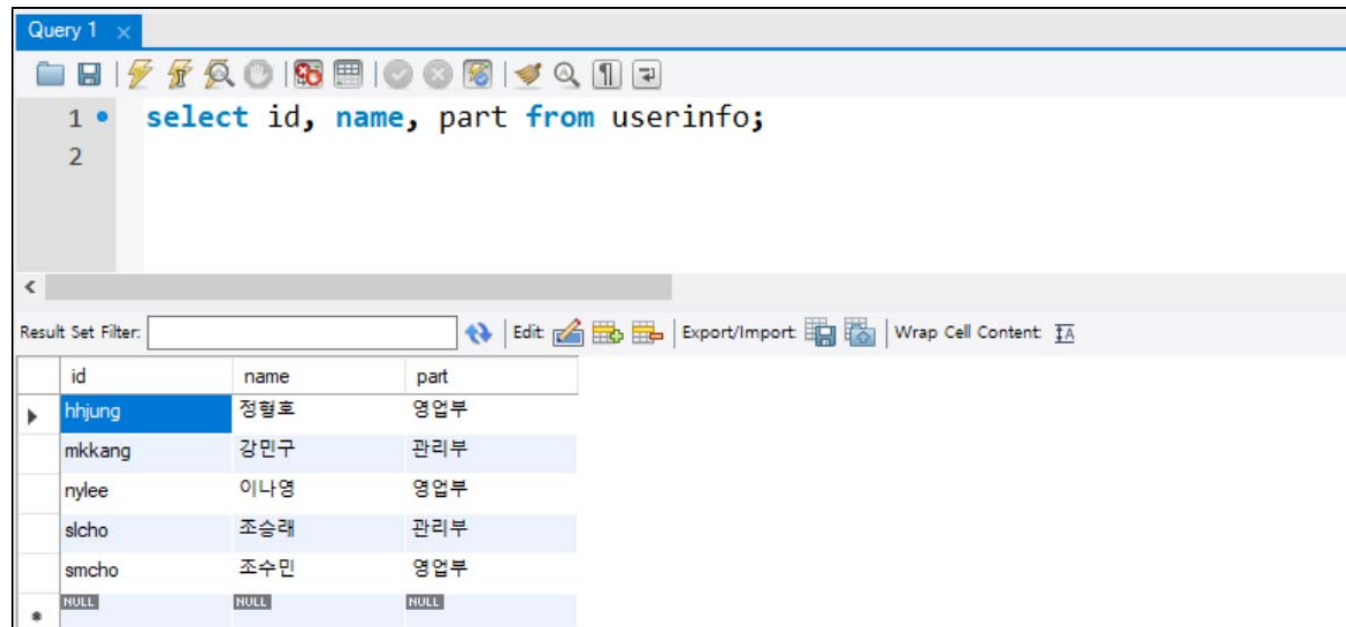
```
select [DISTINCT] 필드1,...필드n from 테이블이름 [where 검색조건 ]  
[order by 필드명 [ASC/DESC] ]  
[ group by 필드명 ]  
[ having 조건 ]  
[ 필드명 between 값1 and 값2 ]  
[ 필드명 like 문자열 ]
```

레코드의 검색

■ 레코드 검색

- 일부 필드 검색

- ▶ 검색하고자 하는 필드의 이름을 콤마(,)로 구분하여 지정



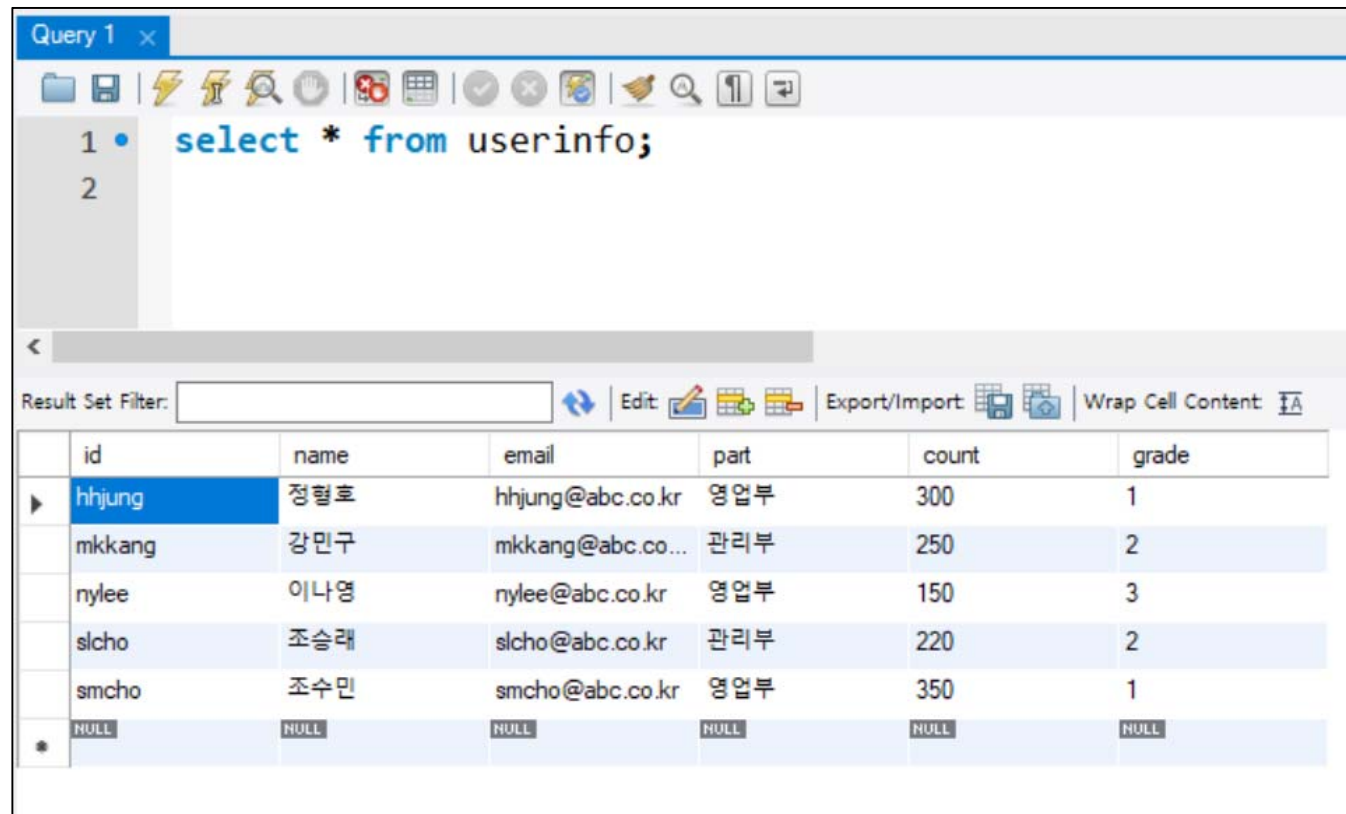
The screenshot shows a database query tool interface. At the top, there's a tab labeled 'Query 1'. Below it is a toolbar with various icons. The main area contains a SQL query: `1 • select id, name, part from userinfo;` and a line number '2'. Below the query is a 'Result Set Filter' input field. At the bottom, there's a table of results with columns 'id', 'name', and 'part'. The first row is highlighted in blue.

id	name	part
hhjung	정형호	영업부
mkkang	강민구	관리부
nylee	이나영	영업부
slcho	조승래	관리부
smcho	조수민	영업부
NULL	NULL	NULL

레코드의 검색

- 전체 필드 검색

- ▶ 검색하고자 하는 필드명 대신 * 기호를 지정



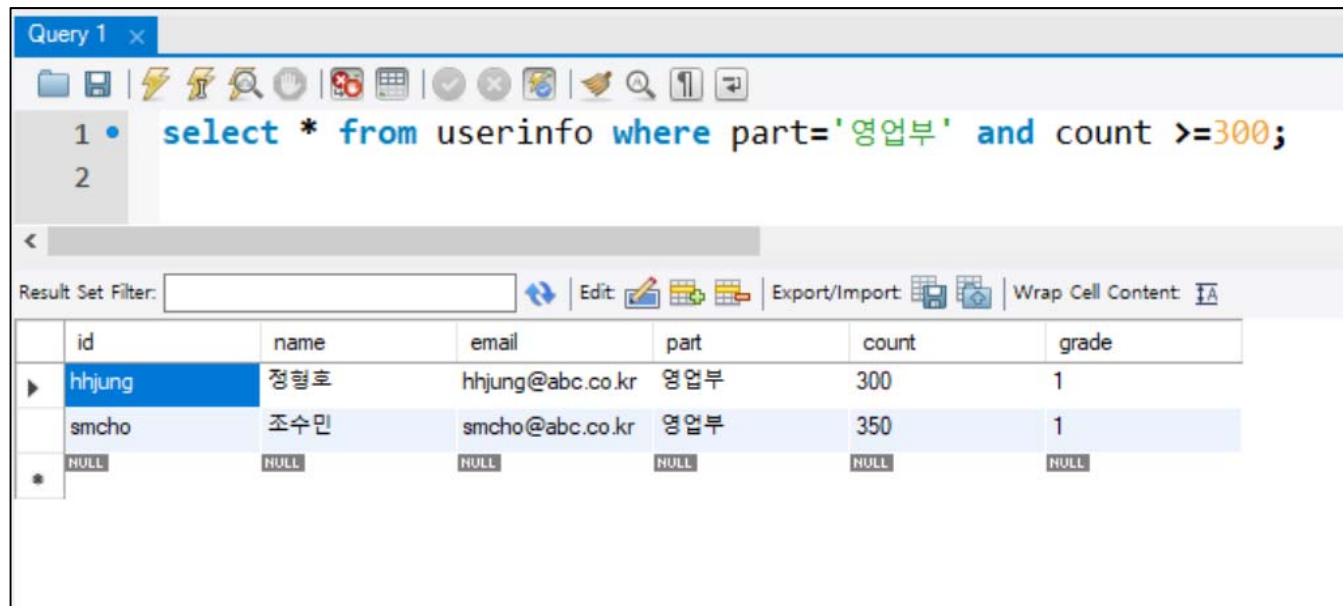
The screenshot shows a database query tool interface. At the top, a tab labeled 'Query 1' is active. Below it is a toolbar with various icons. The main area contains a SQL query: `select * from userinfo;`. Below the query editor is a 'Result Set Filter' input field. The results are displayed in a table with 7 columns: id, name, email, part, count, and grade. The first five rows contain user data, and the last row is a summary row with NULL values. The first row is highlighted in blue.

	id	name	email	part	count	grade
▶	hhjung	정형호	hhjung@abc.co.kr	영업부	300	1
	mkkang	강민구	mkkang@abc.co...	관리부	250	2
	nylee	이나영	nylee@abc.co.kr	영업부	150	3
	slcho	조승래	slcho@abc.co.kr	관리부	220	2
	smcho	조수민	smcho@abc.co.kr	영업부	350	1
*	NULL	NULL	NULL	NULL	NULL	NULL

레코드의 검색

■ where 구문

- **지정한 조건에 만족하는 레코드를 출력**
 - ▶ 하나 이상의 검색 조건을 포함할 수 있으며, 검색 조건이 두 개 이상일 경우 AND를 사용해 연결
- **[실습]**
 - ▶ part가 영업부이고 count가 300 이상인 레코드의 모든 필드를 출력



The screenshot shows a database query tool interface. At the top, a tab labeled 'Query 1' is active. Below it is a toolbar with various icons. The main area displays a SQL query: `select * from userinfo where part='영업부' and count >=300;`. Below the query editor is a 'Result Set Filter' field. The results are displayed in a table with columns: id, name, email, part, count, and grade. The table contains three rows: one for 'hhjung' (count 300), one for 'smcho' (count 350), and a row of NULL values. The first two rows are highlighted in blue.

id	name	email	part	count	grade
hhjung	정형호	hhjung@abc.co.kr	영업부	300	1
smcho	조수민	smcho@abc.co.kr	영업부	350	1
NULL	NULL	NULL	NULL	NULL	NULL

레코드의 검색

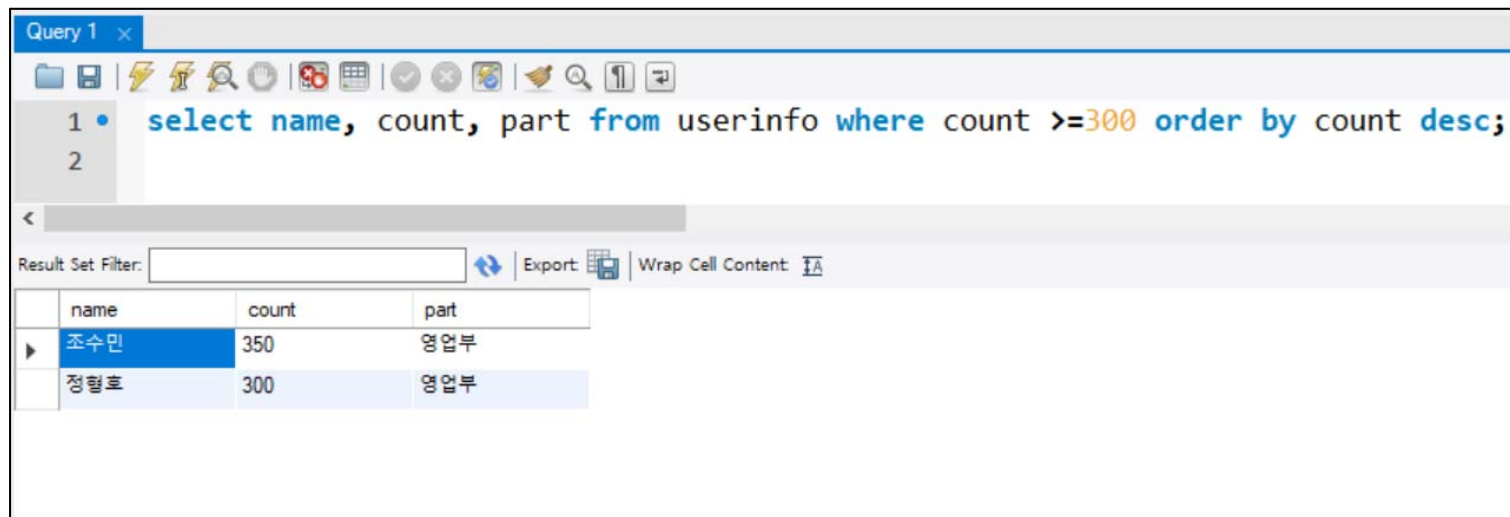
■ order by 구문

- 레코드를 정렬하여 검색할 때 사용되는 구문

- ▶ 오름차순 정렬을 수행하고자 할 경우 asc를 사용
- ▶ 내림차순 정렬을 사용하고자 할 경우 desc를 사용

- [실습]

- ▶ count가 300 이상인 사람의 name, count, part 필드를 출력하되 count를 기준으로 내림차순으로 정렬



The screenshot shows a SQL query editor window titled "Query 1". The query text is: `select name, count, part from userinfo where count >=300 order by count desc;`. Below the query editor, there is a "Result Set Filter" field and an "Export" button. The results are displayed in a table with three columns: name, count, and part. The table contains two rows: one for "조수민" with a count of 350 and part "영업부", and another for "정형호" with a count of 300 and part "영업부".

name	count	part
조수민	350	영업부
정형호	300	영업부

레코드의 검색

■ group by 구문

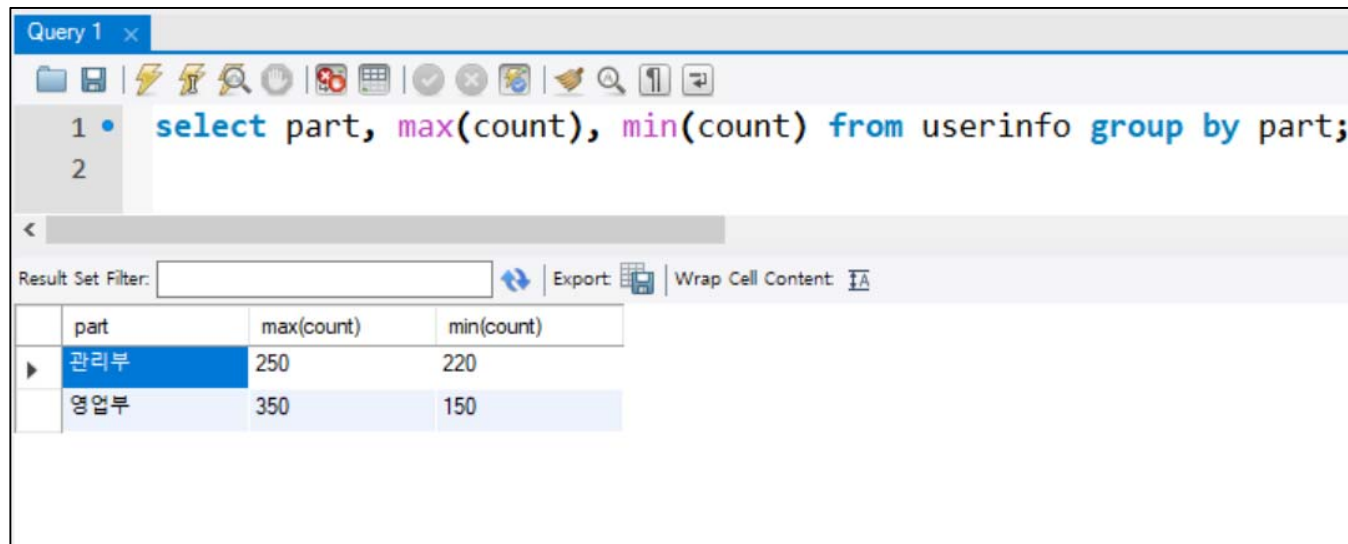
- 테이블에 존재하는 레코드들을 특정 필드를 기준으로 그룹을 생성하고자 할 때 사용
 - ▶ [예] part 필드가 가지는 두 개의 필드 값인 [영업부]와 [관리부]를 기준으로 그룹을 지을 때 사용 가능
- GROUP BY 구문에서 사용하는 함수

그룹 함수명	설명
count(필드명)	지정된 필드에 데이터를 가지고 있는 레코드들의 수를 반환
sum(필드명)	지정된 필드에 저장되어 있는 데이터들의 합계를 반환
min(필드명)	지정된 필드에 저장되어 있는 데이터들 중 가장 작은 값을 반환
max(필드명)	지정된 필드에 저장되어 있는 데이터들 중 가장 큰 값을 반환
avg(필드명)	지정된 필드에 저장되어 있는 데이터들의 평균 값을 반환

레코드의 검색

- [실습]

- ▶ part를 기준으로 그룹을 형성하고, part의 이름과 각 part에 대한 최대 count 값과 최소 count 값을 출력



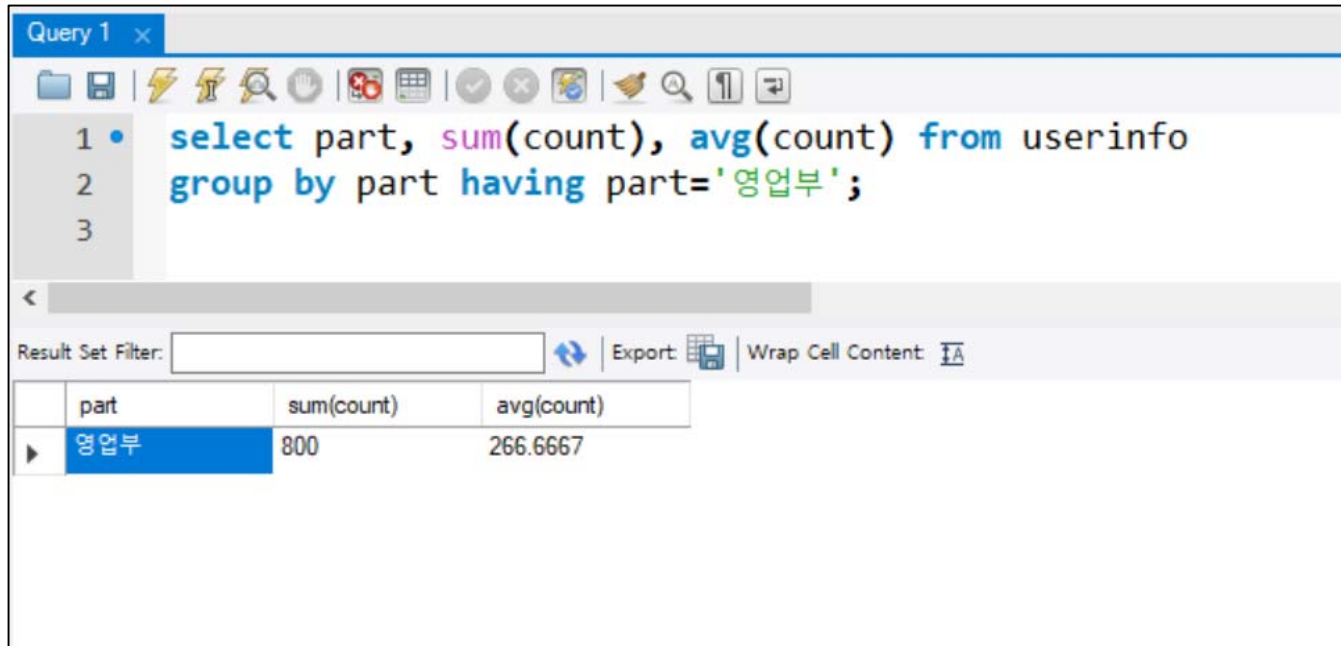
The screenshot shows a SQL query editor window titled "Query 1". The query is: `select part, max(count), min(count) from userinfo group by part;`. Below the query, there is a "Result Set Filter" field and buttons for "Export" and "Wrap Cell Content". The results are displayed in a table with three columns: "part", "max(count)", and "min(count)".

part	max(count)	min(count)
관리부	250	220
영업부	350	150

레코드의 검색

■ having 구문

- group by 구문으로 지정한 필드에 추가로 조건을 지정할 때 사용되는 구문
 - ▶ [예] GROUP BY 구문의 예에서는 출력된 [영업부]와 [관리부] 중에서 하나만 출력하고자 할 경우
- [실습]
 - ▶ part가 관리부인 레코드의 count에 대한 합계와 평균 만을 출력



The screenshot shows a SQL query editor window titled "Query 1". The query text is:

```
1 • select part, sum(count), avg(count) from userinfo
2 group by part having part='영업부';
3
```

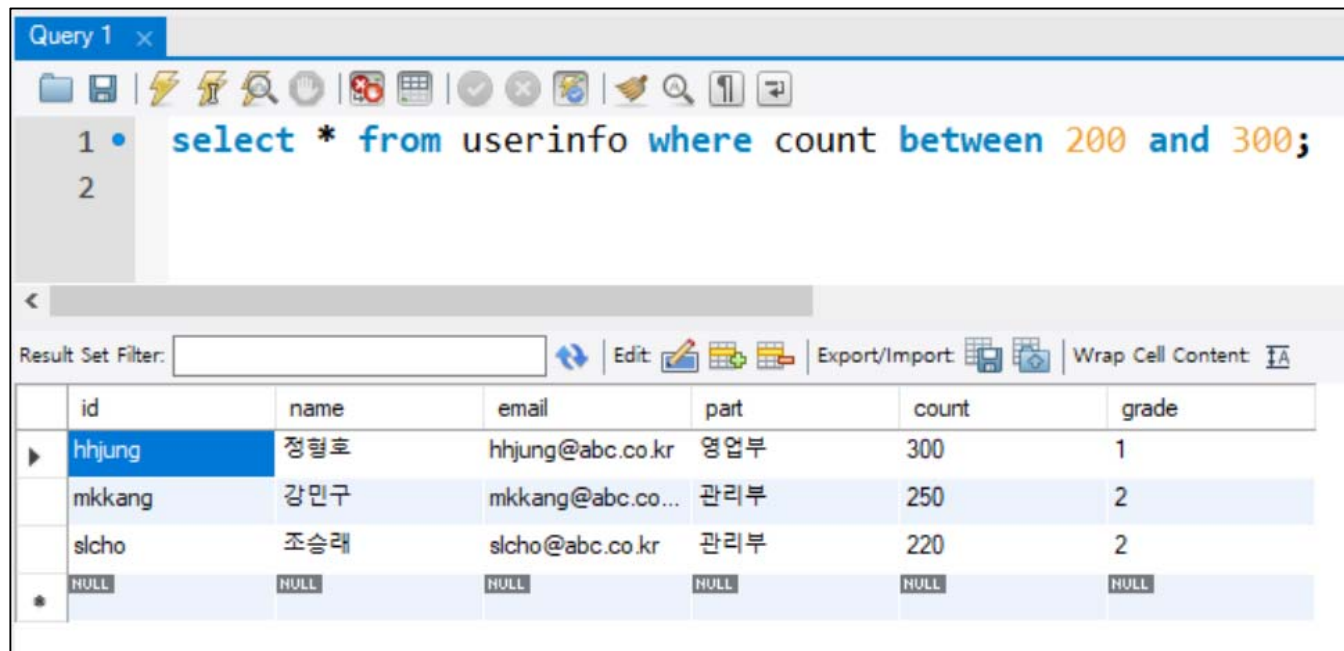
Below the query editor, there is a "Result Set Filter:" field and buttons for "Export" and "Wrap Cell Content". The result set is displayed in a table with the following data:

	part	sum(count)	avg(count)
▶	영업부	800	266.6667

레코드의 검색

■ between 구문

- 지정된 필드의 데이터 중 특정 범위 내에 존재하는 레코드를 검색하고자 할 때 사용
 - ▶ where 구문과 함께 사용됨
- [실습]
 - ▶ count의 값이 200에서 300 사이에 있는 레코드의 모든 필드를 출력



The screenshot shows a database query editor window titled "Query 1". The SQL query entered is: `select * from userinfo where count between 200 and 300;`. Below the query editor, there is a "Result Set Filter" field and a toolbar with icons for Edit, Export/Import, and Wrap Cell Content. The results are displayed in a table with the following columns: id, name, email, part, count, and grade. The table contains three rows of data, with the first row highlighted in blue.

	id	name	email	part	count	grade
▶	hhjung	정형호	hhjung@abc.co.kr	영업부	300	1
	mkkang	강민구	mkkang@abc.co...	관리부	250	2
	slcho	조승래	slcho@abc.co.kr	관리부	220	2
*	NULL	NULL	NULL	NULL	NULL	NULL

레코드의 검색

■ LIKE 구문

- 지정된 필드 내에 특정 문자열이 존재하는지 조사하는 구문
 - ▶ where 구문과 함께 사용
 - ▶ 와일드 카드(%)와 함께 사용 가능
- 와일드 카드 사용의 예
 - ▶ %조%
 - 문자열 내에 '조'라는 문자만 존재하는지 조사
 - ▶ %조
 - '%조'는 '조'의 앞에는 어떠한 문자가 와도 상관없지만 반드시 '조'로 끝나야 한다는 것을 의미
 - ▶ 조%
 - '조'의 뒤에는 어떠한 문자가 와도 상관없지만 반드시 '조'로 시작되어야 한다는 것을 의미

레코드의 검색

- [실습]

- ▶ 'name' 필드에 '조'라는 문자가 들어있는 레코드의 모든 필드를 출력

The screenshot shows a SQL query editor window titled "Query 1". The query text is: `select * from userinfo where name like '%조%';`. Below the query editor, there is a "Result Set Filter" field and a toolbar with icons for "Edit", "Export/Import", and "Wrap Cell Content". The result set is displayed as a table with the following columns: id, name, email, part, count, and grade. The table contains three rows: one for 'slcho' (조승래), one for 'smcho' (조수민), and a row of NULL values. The first row is highlighted in blue.

id	name	email	part	count	grade
slcho	조승래	slcho@abc.co.kr	관리부	220	2
smcho	조수민	smcho@abc.co.kr	영업부	350	1
NULL	NULL	NULL	NULL	NULL	NULL

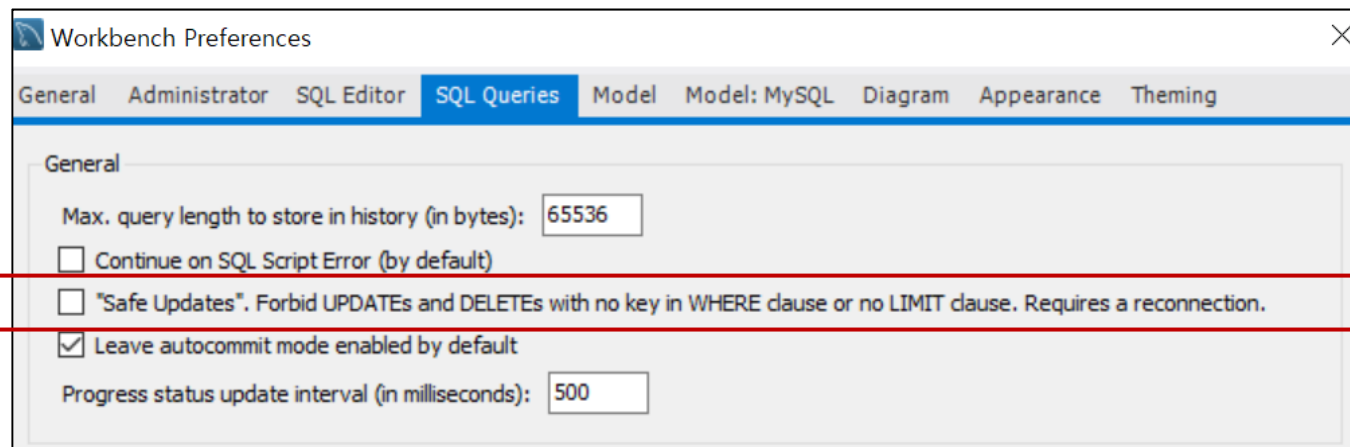
레코드의 변경

■ 레코드의 변경

update 테이블이름 **set** 필드명1=값1 ,... , 필드명n=값n [where 조건]

- 안전 모드를 해제해야 테이블의 수정과 삭제가 가능함

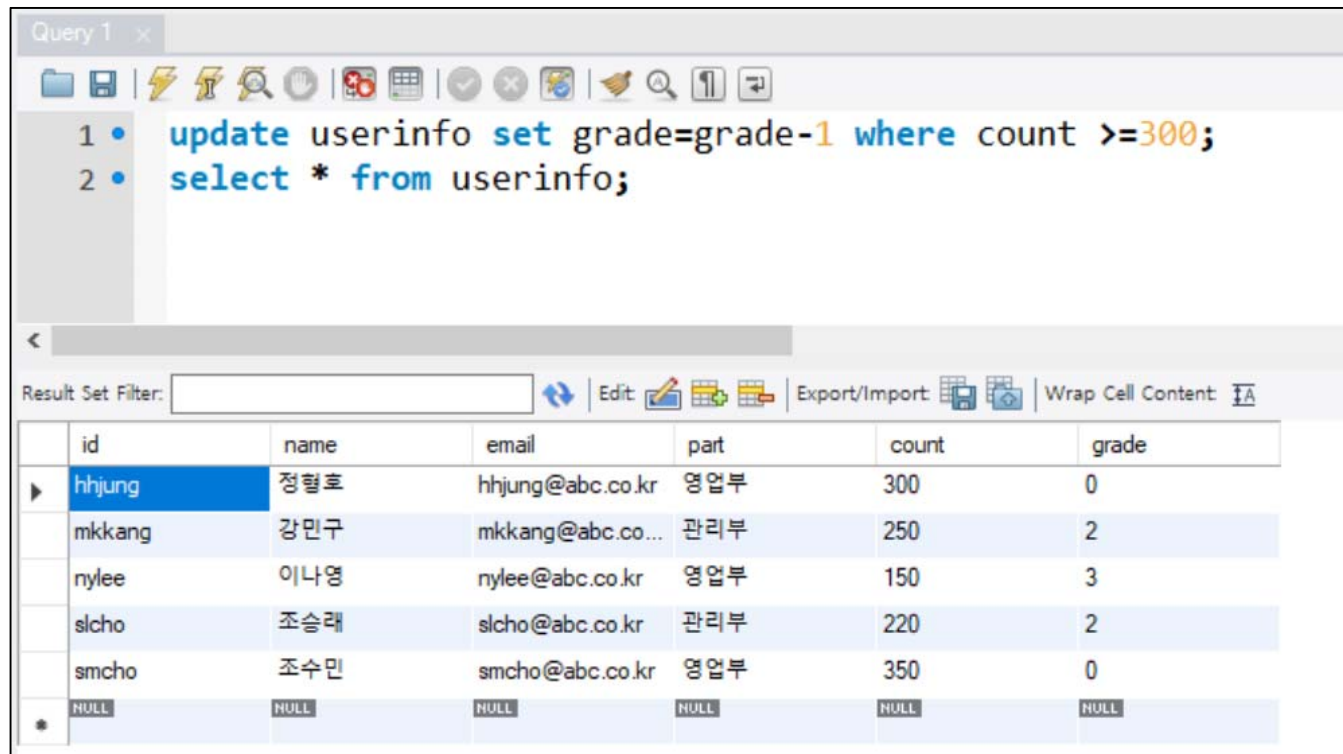
- ▶ MySQL 워크벤치의 [Edit]/ [Peferences]를 실행
- ▶ [SQL Query] 탭에서 'safe updates'를 해제
- ▶ MySQL 워크벤치를 재시작해야 함



레코드의 변경

- [실습]

- ▶ count 값이 300 이상인 사람의 grade 값을 1만큼 감소시킨 다음 userinfo 테이블의 모든 레코드와 필드를 출력



The screenshot shows a SQL query editor window titled 'Query 1'. It contains two SQL statements: an update statement to decrease the grade by 1 for users with a count of 300 or more, and a select statement to retrieve all data from the userinfo table. Below the editor, a result set table is displayed with columns: id, name, email, part, count, and grade. The table contains six rows of data, with the first row (hhjung) highlighted. The last row shows NULL values for all columns.

```
1 • update userinfo set grade=grade-1 where count >=300;
2 • select * from userinfo;
```

	id	name	email	part	count	grade
▶	hhjung	정철호	hhjung@abc.co.kr	영업부	300	0
	mkkang	강민구	mkkang@abc.co...	관리부	250	2
	nylee	이나영	nylee@abc.co.kr	영업부	150	3
	slcho	조승래	slcho@abc.co.kr	관리부	220	2
	smcho	조수민	smcho@abc.co.kr	영업부	350	0
*	NULL	NULL	NULL	NULL	NULL	NULL

레코드의 삭제

■ 레코드의 삭제

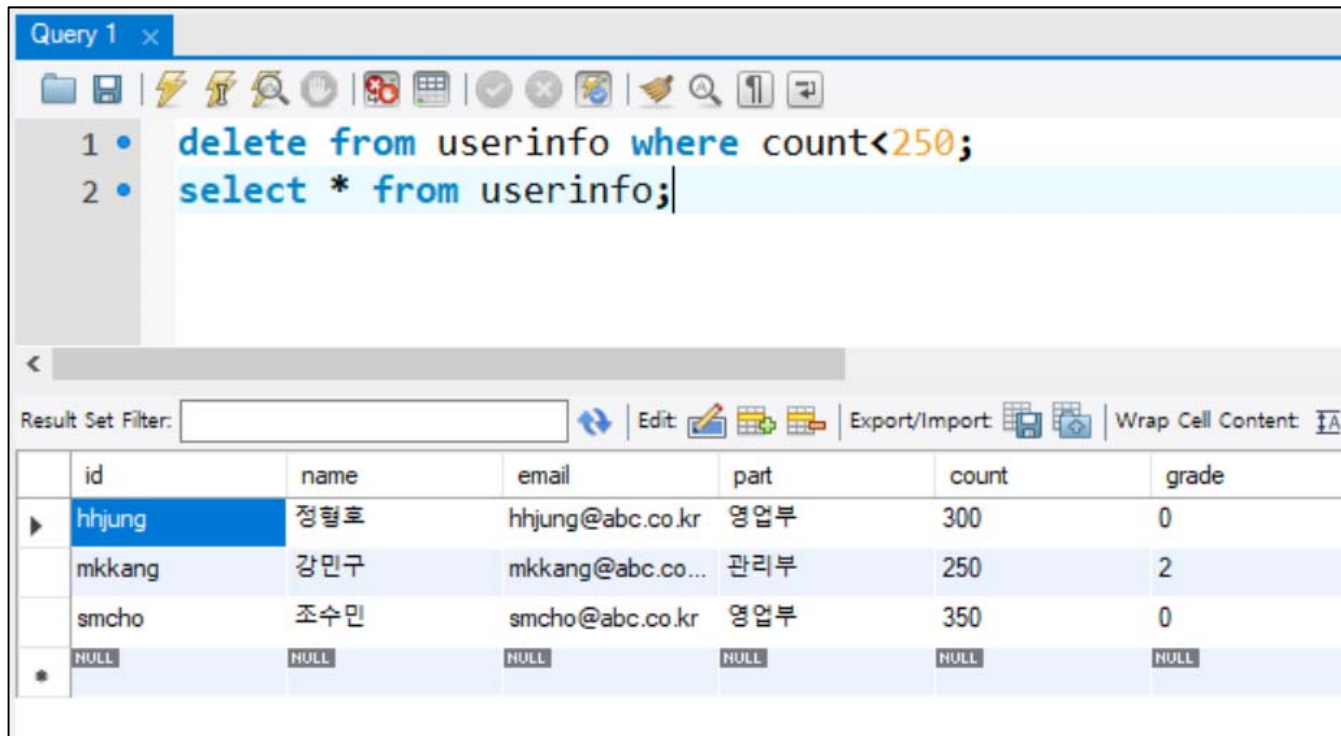
```
delete from 테이블 이름 [ where 조건 ]
```

- DELETE 절의어는 테이블에 있는 모든 레코드를 삭제할 수 있음
 - WHERE 절을 생략할 경우 테이블에 존재하는 모든 레코드를 제거
 - 제거된 레코드는 복원 불가능
- 조건에 부합하는 레코드만을 선별하여 삭제할 수 있음
 - 제거하고자 하는 조건을 포함하는 where 구문과 함께 사용

레코드의 삭제

- [실습]

- ▶ count 값이 250보다 작은 레코드를 모두 삭제한 다음 모든 레코드와 필드를 출력



The screenshot shows a SQL query editor window titled "Query 1". The query text is as follows:

```
1 • delete from userinfo where count<250;
2 • select * from userinfo;
```

Below the query editor, there is a "Result Set Filter:" field and a toolbar with icons for "Edit", "Export/Import", and "Wrap Cell Content". The result set is displayed in a table with the following columns: id, name, email, part, count, and grade.

	id	name	email	part	count	grade
▶	hhjung	정형호	hhjung@abc.co.kr	영업부	300	0
	mkkang	강민구	mkkang@abc.co...	관리부	250	2
	smcho	조수민	smcho@abc.co.kr	영업부	350	0
*	NULL	NULL	NULL	NULL	NULL	NULL

테이블 제거

■ 테이블 제거

```
drop table 테이블이름;
```

- DROP TABLE이 수행되면 테이블 내의 모든 레코드는 테이블과 함께 제거됨
 - 한번 제거된 테이블은 복구할 수 없으므로 주의해야 함
- userinfo 테이블은 차후에 사용할 것이므로 제거해서는 안됨

수고하셨습니다