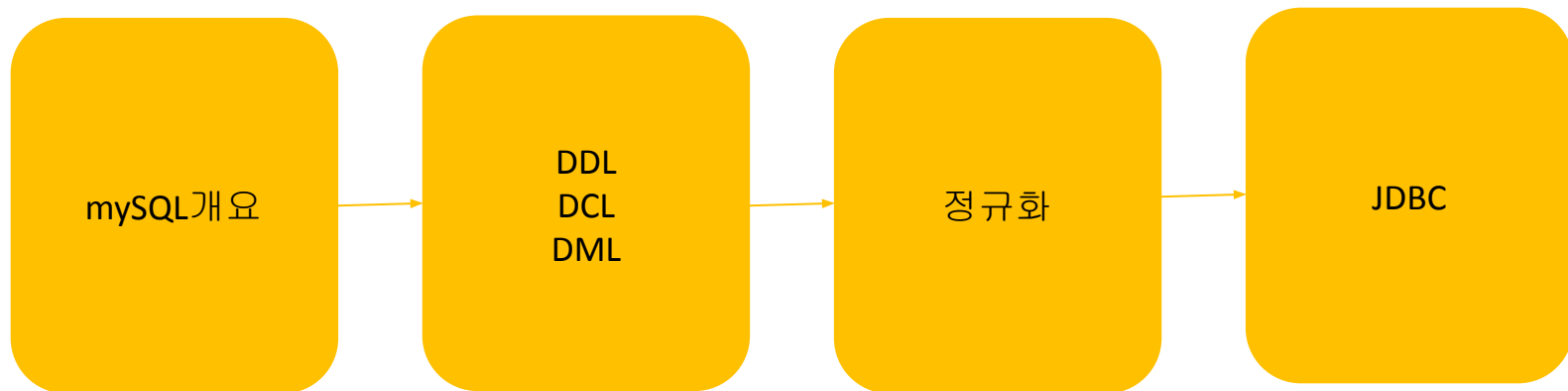


2024년 상반기 K-디지털 트레이닝

DML 심화(함수)

[KB] IT's Your Life





*SCOTT DEPT EMP								
Properties Data 엔터티 관계도								
EMP Enter a SQL expression to filter results (use Ctrl+Space)								
	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	7,369	SMITH	CLERK	7,902	2-17 00:00:00	800	[NULL]	20
2	7,499	ALLEN	SALESM	7,698	1-20 00:00:00	1,600	300	30
3	7,521	WARD	SALESM	7,698	1-22 00:00:00	1,250	500	30
4	7,566	JONES	MANAGI	7,839	1-02 00:00:00	2,975	[NULL]	20
5	7,654	MARTIN	SALESM	7,698	1-28 00:00:00	1,250	1,400	30
6	7,698	BLAKE	MANAGI	7,839	5-01 00:00:00	2,850	[NULL]	30
7	7,782	CLARK	MANAGI	7,839	1-09 00:00:00	2,450	[NULL]	10
8	7,788	SCOTT	ANALYS	7,566	1-09 00:00:00	3,000	[NULL]	20
9	7,839	KING	PRESIDE	[NULL]	1-17 00:00:00	5,000	[NULL]	10
10	7,844	TURNER	SALESM	7,698	1-08 00:00:00	1,500	0	30
11	7,876	ADAMS	CLERK	7,788	1-12 00:00:00	1,100	[NULL]	20
12	7,900	JAMES	CLERK	7,698	1-03 00:00:00	950	[NULL]	30
13	7,902	FORD	ANALYS	7,566	1-03 00:00:00	3,000	[NULL]	20
14	7,934	MILLER	CLERK	7,782	1-23 00:00:00	1,300	[NULL]	10

EMP	
123	EMPNO
ABC	ENAME
ABC	JOB
123	MGR
🕒	HIREDATE
123	SAL
123	COMM
123	DEPTNO

DEPT	
123	DEPTNO
ABC	DNAME
ABC	LOC

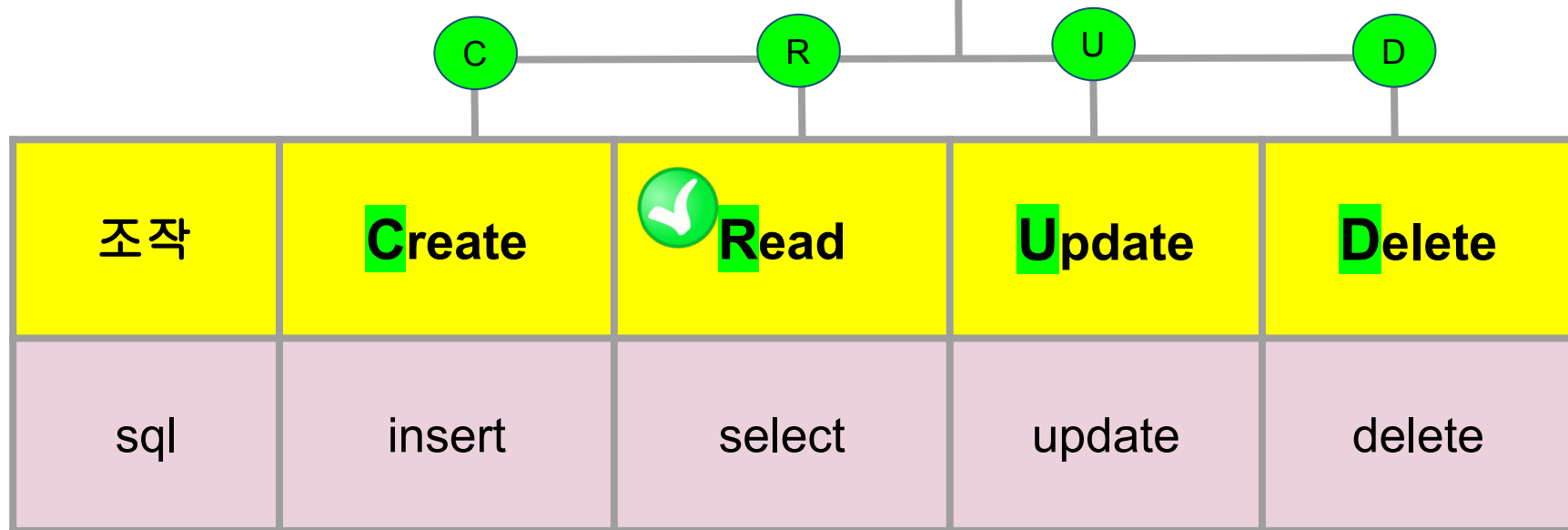


- DML
 - insert: unique, pk, fk, sysdate(mysql now())
 - select:
 - where
 - is null, is not null
 - and, or, like(%, _)
 - as
 - order by, desc

DML 심화 개요

용어	Data Definition Language (DDL)	Data Manipulation Language (DML)	Data Control Language (DCL)	Transaction Control Language (TCL)
역할	데이터 항목 정의	데이터 조작	DBMS 제어 (계정승인, 권한부여/회수)	트랜잭션 제어
SQL 명령어	CREATE, ALTER, DROP, TRUNCATE	INSERT, SELECT, UPDATE, DELETE	GRANT, REVOKE	COMMIT, ROLLBACK

조작 4가지



주요 함수

내장 함수(built-in)



dbms 설치시
함께 들어있었던 명령어(함수)
→ 기능 처리 단위
(count, sum, avg 등)

사용자 정의 함수(user-defined)



사용자가 필요에 의해
기능 처리 단위로 만드는 명령어

내장함수의 종류-용어 안 중요함. 개념만

- 단일행 함수: 하나의 행씩 처리되어 하나의 행씩 결과가 나오는 함수
- 다중행 함수: 여러 개의 행에 대해 처리하여 최종 하나의 행으로 결과가 나오는 함수

단일행 함수

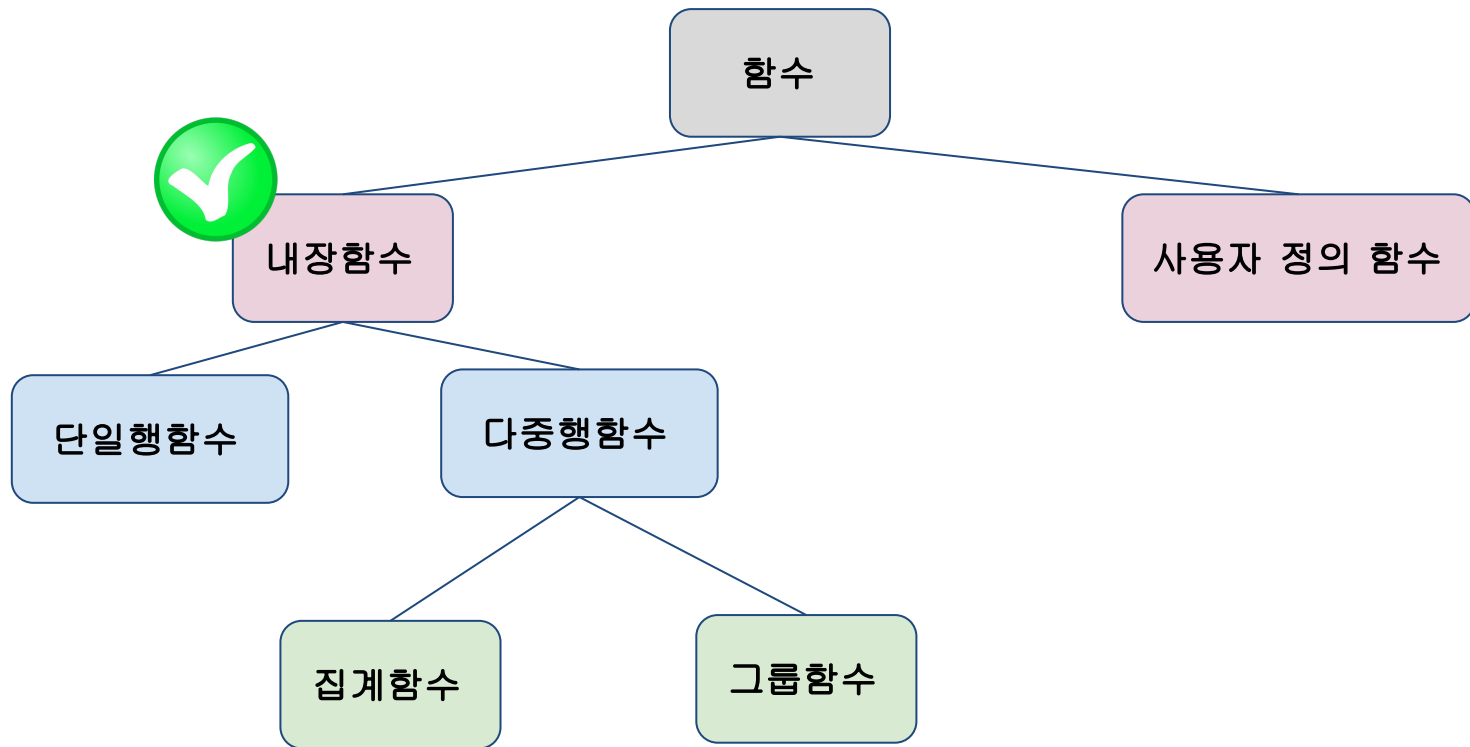
열 1	열 2	...	열 N		열 1	열 2	...	열 N
행 1				→	행 1			
행 2				→	행 2			
...				→	...			
행 N				→	행 N			

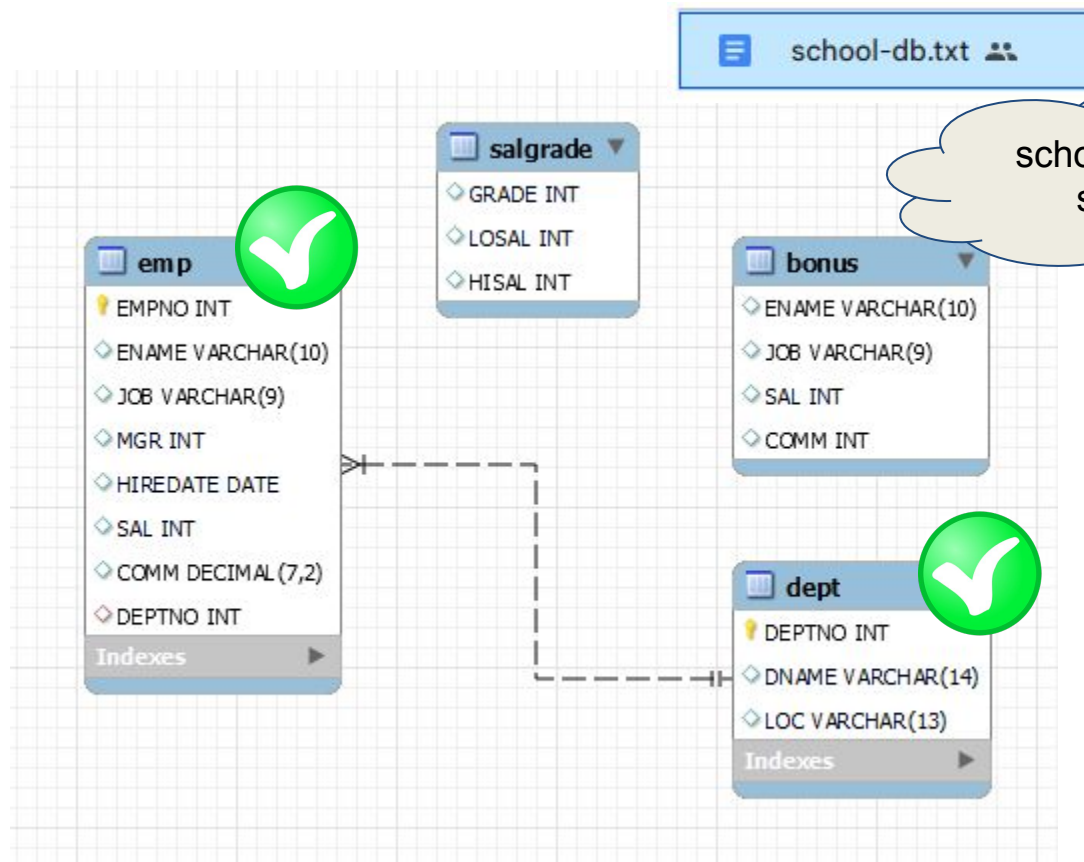
다중행 함수

열 1	열 2	...	열 N		열 1	열 2	...	열 N
행 1]	행 1			
행 2								
...								
행 N								

예)
전체 한 행의
특정컬럼의 값을
모두 대문자로 변경

예)
다중행의 특정한
컬럼의 합, 평균





```
CREATE TABLE DEPT
(  
    DEPTNO int(2),  
    DNAME VARCHAR(14),  
    LOC VARCHAR(13)  
);
```

```
CREATE TABLE EMP(  
    EMPNO int(11),  
    ENAME VARCHAR(10),  
    JOB VARCHAR(9),  
    MGR int(11),  
    HIREDATE DATE,  
    SAL int(11),  
    COMM decimal(7,2),  
    DEPTNO int(2)  
);
```

```
CREATE TABLE SALGRADE  
(  
    GRADE int(11),  
    LOSAL int(11),  
    HISAL int(11)  
);
```

```
CREATE TABLE BONUS  
(  
    ENAME VARCHAR(10),  
    JOB VARCHAR(9),  
    SAL int(11),  
    COMM int(11)  
);
```

```
ALTER TABLE DEPT ADD (  
CONSTRAINT PK_DEPT  
PRIMARY KEY(DEPTNO));
```

```
ALTER TABLE EMP ADD (  
CONSTRAINT PK_EMP  
PRIMARY KEY(EMPNO));
```

```
ALTER TABLE EMP ADD (  
CONSTRAINT FK_DEPTNO  
FOREIGN KEY (DEPTNO)  
REFERENCES DEPT (DEPTNO));
```

EMP_DATA.csv

SALGRADE_DATA.csv

DEPT_DATA.csv

data import

1. SELECT * FROM school.salgrade;

GRADE	LOSAL	HISAL
1	700	1200
2	1201	1400
3	1401	2000
4	2001	3000
5	3001	9999

1. SELECT * FROM school.emp;

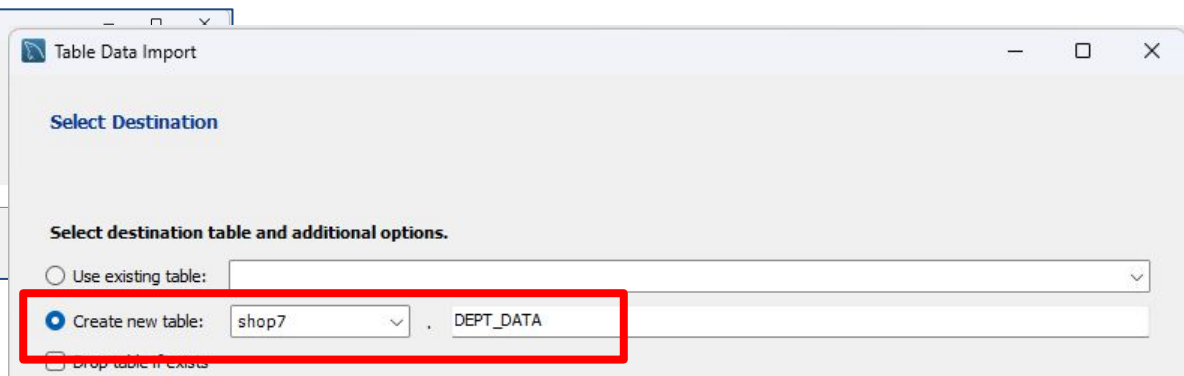
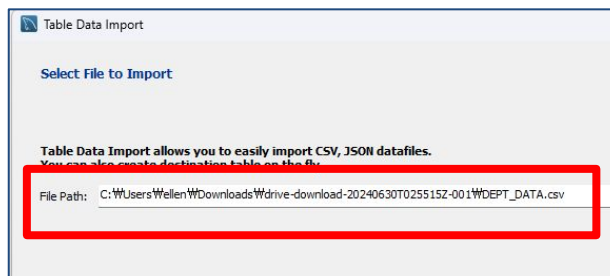
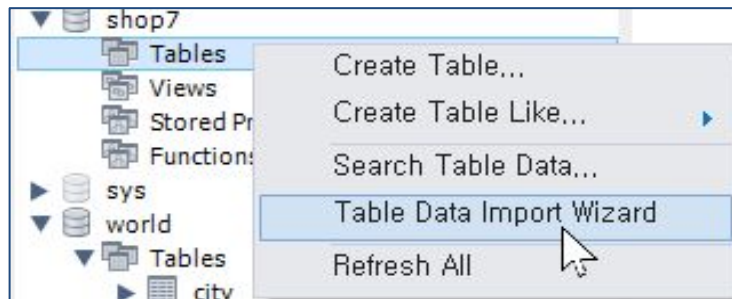
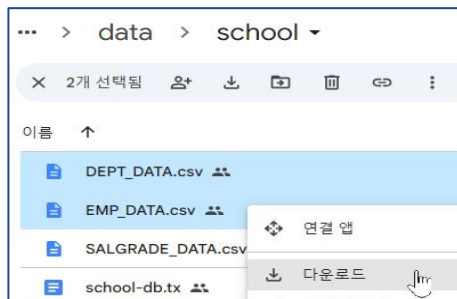
EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7499	ALLEN	SALESMAN	7698	1981-02-20	1600	300.00	30
7521	WARD	SALESMAN	7698	1981-02-22	1250	500.00	30
7654	MARTIN	SALESMAN	7698	1981-09-28	1250	1400.00	30
7844	TURNER	SALESMAN	7698	1981-09-08	1500	0.00	30

1. SELECT * FROM school.dept;

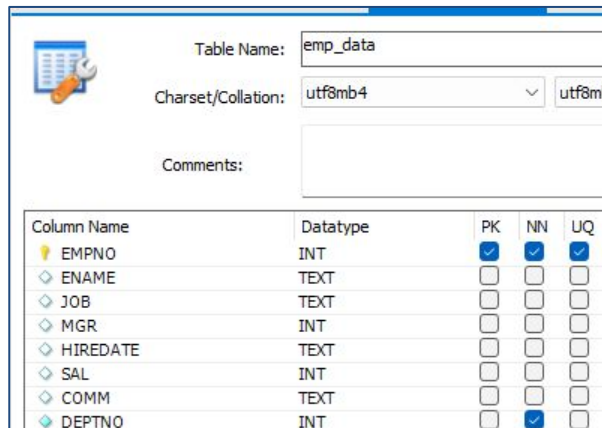
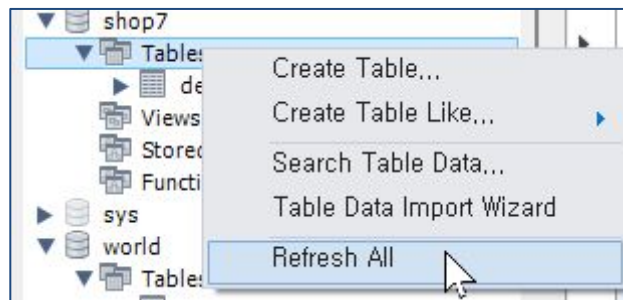
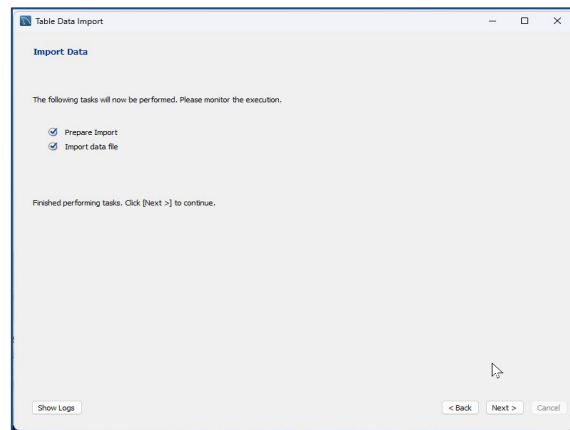
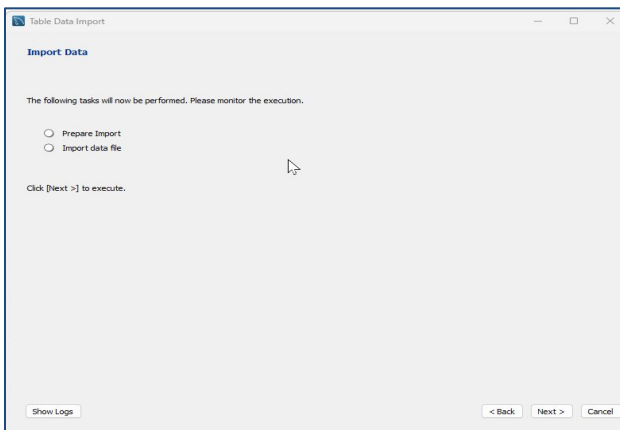
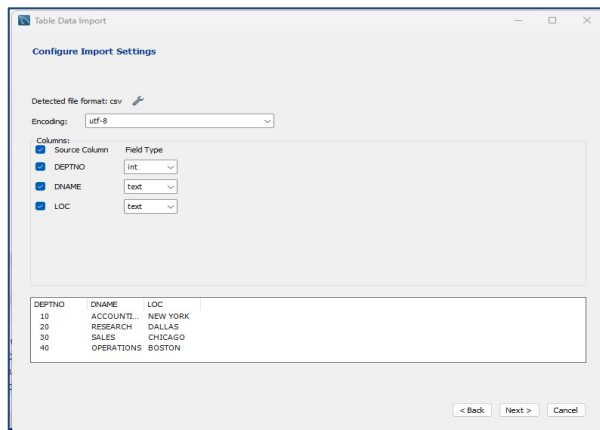
DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

import data가 모두 되지 않는 경우

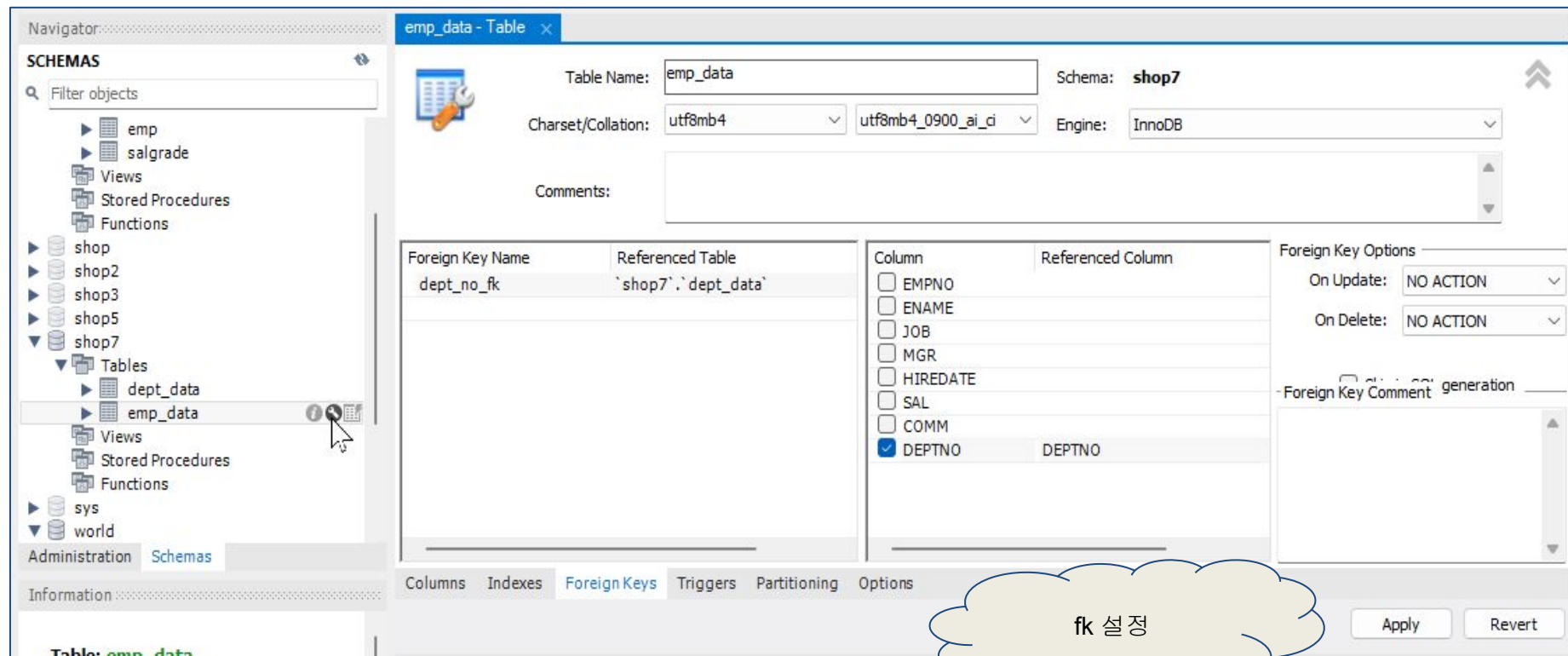
- 1) 외래키 삭제 2) 기존 emp테이블을 삭제 후, 3)다음과 같이 import하여 테이블 명 수정



import data가 모두 되지 않는 경우



- 외래키 설정 (emp_data table의 deptno와 dept_data table의 deptno)





The screenshot shows the MySQL Workbench interface with the 'emp_data' table selected in the 'shop7' schema. The 'Foreign Keys' tab is active, displaying the following configuration:

Foreign Key Name	Referenced Table	Column	Referenced Column
dept_no_fk	`shop7`.`dept_data`	<input checked="" type="checkbox"/> DEPTNO	DEPTNO

Additional settings shown include:

- Table Name: emp_data
- Schema: shop7
- Charset/Collation: utf8mb4, utf8mb4_0900_ai_ci
- Engine: InnoDB
- Foreign Key Options: On Update: NO ACTION, On Delete: NO ACTION
- Foreign Key Comment: generation

A cloud-shaped callout with the text 'fk 설정' (FK Setting) is positioned over the bottom right of the interface.

함수명	사용목적	사용법
UPPER/LOWER	대문자/소문자/첫글자만 대문자로 변환	select UPPER(ENAME) from EMP; // WILL
 LENGTH	글자수	select LENGTH(ENAME) from EMP; // 5
SUBSTR/INSTR	문자열 추출/문자열의 위치	select SUBSTR(ENAME, 2) from EMP; // ILL (2번째 글자부터 끝까지) select SUBSTR(ENAME, 1, 2) from EMP; // WI (1번째 글자부터 2개) select SUBSTR(ENAME, -2) from EMP; // LL (-2번째 글자부터 끝까지)
REPLACE	특정문자를 다른 문자로 대체	select REPLACE(ENAME, 'L', 'N') from EMP; // WINN
LPAD/RPAD	빈 공간을 특정문자로 대체	select LPAD(ENAME, 6, '#') from EMP; // ##WILL (6개의 나머지 앞자리를 #으로 채우기,채울 문자를 넣지 않은 경우 공백으로 채워짐)
 CONCAT	문자열을 결합	select CONCAT(EMPNO, ENAME) from EMP; // 78WILL
TRIM/LTRIM/RTRIM	특정문자나 공백을 삭제 (앞뒤/앞만/뒷만)	select TRIM('W' from ENAME) from EMP; // ILL select TRIM(' hong ') from EMP; // hong

10. `SELECT concat(no, ":", title) as result from bbs;`

result Grid | Filter Rows: | Export: | Wrap Cell Content: IA

result

100:fun1

101:fun2

102:fun3

103:fun4

104:fun5

105:fun6

106:fun7

`SELECT concat(CONCAT(empno, ':'), ENAME) AS concat_result FROM emp`

Results 1

`SELECT concat(CONCAT(empno, ':'), ENAME) AS concat_re` Enter a SQL expression to filter results (use Ctrl+Space)

	CONCAT_RESULT
1	7369: SMITH
2	7499: ALLEN a
3	7521:AWARDA
4	7566:JONES
5	7654:MARTIN

mysql8 length, char_length 비교

- **LENGTH(string)**: 문자열의 바이트 길이를 반환. 각 문자가 여러 바이트로 인코딩될 수 있는 멀티바이트 문자셋을 사용하는 경우, 이 함수는 문자열의 실제 바이트 수를 반환
- **CHAR_LENGTH(string)**: 문자열의 문자 개수를 반환. 각 문자가 몇 바이트인지와 상관없이 문자열 내의 문자의 총 수를 반환

40 • **SELECT** LENGTH('abc'); -- 결과: 3

41 • **SELECT** LENGTH('가나다'); -- 결과: 9 (UTF-8 인코딩 기준, 한글 문자는 각각 3 바이트)

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
LENGTH('가나다')			
9			

43 • **SELECT** CHAR_LENGTH('abc'); -- 결과: 3

44 • **SELECT** CHAR_LENGTH('가나다'); -- 결과: 3 (각 한글 문자가 3 바이트이지만 문자의 개수는 3)

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
CHAR_LENGTH('가나다')			
3			

추가 문자열 함수

- **LEFT()**
 - SELECT LEFT('Hello', 2); -- 'He'
- **RIGHT()**
 - SELECT RIGHT('Hello', 2); -- 'lo'
- **MID()**
 - SELECT MID('Hello', 2, 3); -- 'ell'
- **REVERSE()**
 - SELECT REVERSE('Hello'); -- 'olleH'
- **LOCATE() == POSITION()**
 - SELECT LOCATE('l', 'Hello'); -- 3
 - SELECT POSITION('l' IN 'Hello'); -- 3
- **FIELD() :** 목록 내에서 문자열의 위치를 반환
 - SELECT FIELD('B', 'A', 'B', 'C'); -- 2
- **FIND_IN_SET() :** 쉼표로 구분된 문자열 목록에서 문자열의 위치를 반환
 - SELECT FIND_IN_SET('B', 'A,B,C'); -- 2

숫자 함수/날짜 함수/자료형 변환 함수

```
select FLOOR(1234.56);
```

함수명	사용목적	사용법
ROUND/TRUNCATE/CEIL/FLOOR/MOD	반올림/버림/올림/내림/나머지	ROUND(1234.567, 1) → 1234.6, ROUND(1234.567, -1) → 1230 TRUNCATE(1234.567, 1) → 1234.5 CEIL(1234.56) → 1235 FLOOR(1234.56) → 1234
ADDDATE	몇 개월 이후 날짜	SELECT ADDDATE('2025-01-01', INTERVAL 3 MONTH); → 3개월 이후 날짜
TIMESTAMPDIFF	두 날짜 데이터 간의 달 차이	TIMESTAMPDIFF (DAY, '2024-02-01', now()) → 재직일수
LAST_DAY	특정날짜에 해당하는 달의 마지막 날짜	LAST_DAY(now()) → 2022-11-30, select current_date(); -- date
DATE_FORMAT()	숫자, 날짜 데이터를 문자로 변환/문자 데이터를 숫자로 변환	date_format(now(), '%Y%m%d%H%i%s'), 소문자는 짧게 표현
STR_TO_DATE	문자 데이터를 날짜 데이터로 변환	select STR_TO_DATE(now(), '%Y-%m-%d');

TIMESTAMPDIFF

- 초 차이 계산: `TIMESTAMPDIFF(SECOND, '2024-02-01', NOW())`
- 분 차이 계산: `TIMESTAMPDIFF(MINUTE, '2024-02-01', NOW())`
- 시간 차이 계산: `TIMESTAMPDIFF(HOUR, '2024-02-01', NOW())`
- 일 차이 계산: `TIMESTAMPDIFF(DAY, '2024-02-01', NOW())`
- 주 차이 계산: `TIMESTAMPDIFF(WEEK, '2024-02-01', NOW())`
- 월 차이 계산: `TIMESTAMPDIFF(MONTH, '2024-02-01', NOW())`
- 분기 차이 계산: `TIMESTAMPDIFF(QUARTER, '2024-02-01', NOW())`
- 년 차이 계산: `TIMESTAMPDIFF(YEAR, '2024-02-01', NOW())`



1) 날짜에 1 DAY ADD하면 내일

2) 날짜를 원하는 형식으로 변환하여 추출

```
SELECT ADDDATE('2025-01-01', INTERVAL 31 DAY), ADDDATE('2025-01-01', INTERVAL  
1 MONTH);  
SELECT SUBDATE('2025-01-01', INTERVAL 31 DAY), SUBDATE('2025-01-01', INTERVAL  
1 MONTH);
```

31일 후, 1달 후

31일 전, 1달 전

```
SELECT ADDTIME('2025-01-01 23:59:59', '1:1:1'), ADDTIME('15:00:00', '2:10:10');  
SELECT SUBTIME('2025-01-01 23:59:59', '1:1:1'), SUBTIME('15:00:00', '2:10:10');
```

1시간 1분 1초 후

1시간 1분 1초 전

```
SELECT YEAR(CURDATE()), MONTH(CURDATE()), DAYOFMONTH(CURDATE());  
SELECT HOUR(CURTIME()), MINUTE(CURRENT_TIME()), SECOND(CURRENT_TIME),  
MICROSECOND(CURRENT_TIME);
```

현재 중 년/월/일
현재 중 시/분/일
DATE(now()), TIME(now()) 날짜,
시간 추출

3) 시각을 원하는 포맷으로 변환하여 추출

날짜와 시간
차이구해줌

```
SELECT DATEDIFF('2025-01-01', NOW()), TIMEDIFF('23:23:59', '12:11:10');
```

```
SELECT QUARTER('2025-07-07');
```

해당 분기

```
SELECT MAKEDATE(2025, 32);
```

2025년 32일 지난 날짜

2025년 1월에서 11개월
지난 월

2025년 1월과 2023년
12월 사이의 월수

```
SELECT PERIOD_ADD(202501, 11), PERIOD_DIFF(202501, 202312);
```

- **ADDDATE**: 날짜에 일수를 더하는 함수
 - SELECT ADDDATE('2025-01-01', INTERVAL 31 DAY); -- 31일 후
 - SELECT ADDDATE('2025-01-01', INTERVAL 1 MONTH); -- 1달 후
- **SUBDATE**: 날짜에서 일수를 빼는 함수
 - SELECT SUBDATE('2025-01-01', INTERVAL 31 DAY); -- 31일 전
 - SELECT SUBDATE('2025-01-01', INTERVAL 1 MONTH); -- 1달 전
- **ADDTIME**: 시간에 시간을 더하는 함수
 - SELECT ADDTIME('2025-01-01 23:59:59', '1:1:1'); -- 1시간 1분 1초 후
 - SELECT ADDTIME('15:00:00', '2:10:10'); -- 2시간 10분 10초 후
- **SUBTIME**: 시간에서 시간을 빼는 함수
 - SELECT SUBTIME('2025-01-01 23:59:59', '1:1:1'); -- 1시간 1분 1초 전
 - SELECT SUBTIME('15:00:00', '2:10:10'); -- 2시간 10분 10초 전

- 현재 연도, 월, 일, 시간, 분, 초 추출
 - `SELECT YEAR(CURDATE()), MONTH(CURDATE()), DAYOFMONTH(CURDATE());` -- 현재 연/월/일
 - `SELECT HOUR(CURTIME()), MINUTE(CURRENT_TIME), SECOND(CURRENT_TIME), MICROSECOND(CURRENT_TIME);` -- 현재 시/분/초/마이크로초
- DATE, TIME 추출
 - `SELECT DATE(NOW()), TIME(NOW());` -- 현재 날짜, 시간 추출

- 날짜와 시간 차이
 - `SELECT DATEDIFF('2025-01-01', NOW()), TIMEDIFF('23:23:59', '12:11:10');`
- 해당 분기
 - `SELECT QUARTER('2025-07-07');`
- 2025년 32일 지난 날짜
 - `SELECT MAKEDATE(2025, 32);`
- 2025년 1월에서 11개월 지난 월, 2025년 1월과 2023년 12월 사이의 월수
 - `SELECT PERIOD_ADD(202501, 11), PERIOD_DIFF(202501, 202312);`

NULL처리 함수(null인 경우, 0으로 채움)

- UPDATE EMP SET COMM = 0 WHERE COMM is null;(선택)

3. UPDATE school.EMP SET COMM = 0 WHERE COMM is null;

Result Grid

Filter Rows:

Edit:

Export/Import:

Wrap Cell Content:

	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
▶	7499	ALLEN	SALESMAN	7698	1981-02-20	1600	300.00	30
	7521	WARD	SALESMAN	7698	1981-02-22	1250	500.00	30
	7654	MARTIN	SALESMAN	7698	1981-09-28	1250	1400.00	30
	7844	TURNER	SALESMAN	7698	1981-09-08	1500	0.00	30

1. SELECT * FROM school.emp;
 2.
 3. UPDATE EMP SET COMM = 0 WHERE COMM = '';

Result Grid

Filter Rows:

Edit:

Export/Import:

Wrap Cell Content:

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPT
7369	SMITH	CLERK	7902	1980-12-17 00:0...	800	0	20

우리 예제는
import시 "공백으로
들어감.

CASE 함수

- 값의 결과에 따라 다르게 처리하여 추출 - mysql)



```

02 SELECT EMPNO, ENAME, JOB, SAL,
03     CASE JOB
04         WHEN 'MANAGER' THEN SAL*1.1
05         WHEN 'SALESMAN' THEN SAL*1.05
06         WHEN 'ANALYST' THEN SAL
07         ELSE SAL*1.03
08     END AS UPSAL
09 FROM EMP;

```

:: 결과 화면

EMPNO	ENAME	JOB	SAL	UPSAL
7369	SMITH	CLERK	800	824
7499	ALLEN	SALESMAN	1600	1680
7521	WARD	SALESMAN	1250	1312.5
7566	JONES	MANAGER	2975	3272.5
7654	MARTIN	SALESMAN	1250	1312.5
7698	BLAKE	MANAGER	2850	3135
7782	CLARK	MANAGER	2450	2695
7788	SCOTT	ANALYST	3000	3000
7839	KING	PRESIDENT	5000	5150
7844	TURNER	SALESMAN	1500	1575
7876	ADAMS	CLERK	1100	1133
7900	JAMES	CLERK	950	978.5
7902	FORD	ANALYST	3000	3000
7934	MILLER	CLERK	1300	1339

```

SELECT EMPNO, ENAME, JOB, SAL,
CASE JOB
    WHEN 'MANAGER' THEN SAL * 1.1
    WHEN 'SALESMAN' THEN SAL * 1.05
    WHEN 'ANALYST' THEN SAL
    ELSE SAL * 1.03
END AS U_SAL
FROM EMP;

```

CASE 함수

- 값의 결과에 따라 다르게 처리하여 추출 - mysql)

:: 결과 화면

EMPNO	ENAME	COMM	COMM_TEXT
7369	SMITH		해당사항 없음
7499	ALLEN	300	수당 : 300
7521	WARD	500	수당 : 500
7566	JONES		해당사항 없음
7654	MARTIN	1400	수당 : 1400
7698	BLAKE		해당사항 없음
7782	CLARK		해당사항 없음
7788	SCOTT		해당사항 없음
7839	KING		해당사항 없음
7844	TURNER	0	수당없음
7876	ADAMS		해당사항 없음
7900	JAMES		해당사항 없음
7902	FORD		해당사항 없음
7934	MILLER		해당사항 없음

```
SELECT EMPNO, ENAME, COMM,
```

```
  CASE
```

```
    WHEN COMM IS NULL THEN '해당사항 없음'
```

```
    WHEN COMM = 0 THEN '수당없음'
```

```
    WHEN COMM > 0 THEN concat('수당 : ', COMM)
```

```
  END AS COMM_TEXT
```


```
FROM EMP;
```


집계 함수 / 그룹 함수

집계/그룹함수

- (함수를 외우는 것이 문제가 아니라, 사용처가 문제다!)

SELECT count(SAL), sum(SAL), AVG(sal)		
FROM EMP		
WHERE JOB = 'SALESMAN'		
<small>SQL> SELECT count(SAL), sum(SAL), AVG(sal) FROM EMP WHERE JOB = 'SALESMAN';</small>		
COUNT(SAL)	SUM(SAL)	AVG(SAL)
4	5,600	1,400

함수명	사용목적	사용법
SUM/COUNT/AVG	합/개수/평균 (** distinct, 중복제외)	select COUNT(SAL) from EMP; select COUNT(distinct SAL) from EMP;
MIN/MAX	최소/최대	select MIN(SAL), MAX(SAL) from EMP;
 GROUP BY	그룹별로 묶을 때	select AVG(SAL) from EMP GROUP BY DEPTNO ; //DEPTNO별로 SAL의 평균 select AVG(SAL) from EMP GROUP BY DEPTNO ORDER BY DEPTNO desc ; //그룹이 먼저, 정렬이 나중 select AVG(SAL) from EMP GROUP BY DEPTNO HAVING AVG(SAL) >= 2000 //조건은 HAVING으로 써야함 ! ORDER BY DEPTNO desc ;

-- 집계함수

```
SELECT count(SAL), sum(SAL), AVG(sal), MIN(SAL), MAX(SAL)
FROM EMP
WHERE JOB = 'SALESMAN'
```

results 1

COUNT(SAL)	SUM(SAL)	AVG(SAL)	MIN(SAL)	MAX(SAL)
4	5,600	1,400	1,250	1,600

```
SELECT job, count(SAL), sum(SAL), round(AVG(sal)), MIN(SAL), MAX(SAL)
FROM EMP
GROUP BY JOB
```

results 1

JOB	COUNT(SAL)	SUM(SAL)	ROUND(AVG(SAL))	MIN(SAL)	MAX(SAL)
1 CLERK	4	4,150	1,038	800	1,300
2 SALESMAN	4	5,600	1,400	1,250	1,600
3 PRESIDENT	1	5,000	5,000	5,000	5,000
4 MANAGER	3	8,275	2,758	2,450	2,975
5 ANALYST	2	6,000	3,000	3,000	3,000

```
SELECT job, count(SAL), sum(SAL), round(AVG(sal)), MIN(SAL), MAX(SAL)
FROM EMP
GROUP BY JOB
ORDER BY JOB DESC
```

1	JOB	COUNT(SAL)	SUM(SAL)	ROUND(AVG(SAL))	MIN(SAL)	MAX(SAL)
1	SALESMAN	4	5,600	1,400	1,250	1,600
2	PRESIDENT	1	5,000	5,000	5,000	5,000
3	MANAGER	3	8,275	2,758	2,450	2,975
4	CLERK	4	4,150	1,038	800	1,300
5	ANALYST	2	6,000	3,000	3,000	3,000

```
SELECT job, count(SAL), sum(SAL), round(AVG(sal)), MIN(SAL), MAX(SAL)
FROM EMP
GROUP BY JOB
HAVING COUNT(SAL) >= 4
ORDER BY JOB DESC
```

1	JOB	COUNT(SAL)	SUM(SAL)	ROUND(AVG(SAL))	MIN(SAL)	MAX(SAL)
1	SALESMAN	4	5,600	1,400	1,250	1,600
2	CLERK	4	4,150	1,038	800	1,300

EMP *<xe> Script-34

```

SELECT job, count(SAL), sum(SAL), round(AVG(sal)), MIN(SAL), MAX(SAL)
FROM EMP
GROUP BY JOB
HAVING COUNT(SAL) >= 4 -- 그룹을 지어서 검색 후, 필터링을 하고자 하는 경우
ORDER BY JOB DESC -- 이미 모든 검색이 다 끝난 후, 그 결과를 정렬하고자 하는 경우, 맨 끝

```

Results 1

SELECT job, count(SAL), sum(SAL), round(AVG(sal)), MIN(S.
Enter a SQL expression to filter results (use Ctrl+Space)

	JOB	COUNT(SAL)	SUM(SAL)	ROUND(AVG(SAL))	MIN(SAL)	MAX(SAL)
1	SALESMAN	4	5,600	1,400	1,250	1,600
2	CLERK	4	4,150	1,038	800	1,300

추가 집계/그룹 함수

- **GROUP_CONCAT()**
 - SELECT DEPTNO, GROUP_CONCAT(ENAME) FROM EMP GROUP BY DEPTNO;
- **VARIANCE()** : 표본 분산을 계산
 - SELECT VARIANCE(SAL) FROM EMP;
- **STDDEV()** : 표본 표준 편차를 계산
 - SELECT STDDEV(SAL) FROM EMP;

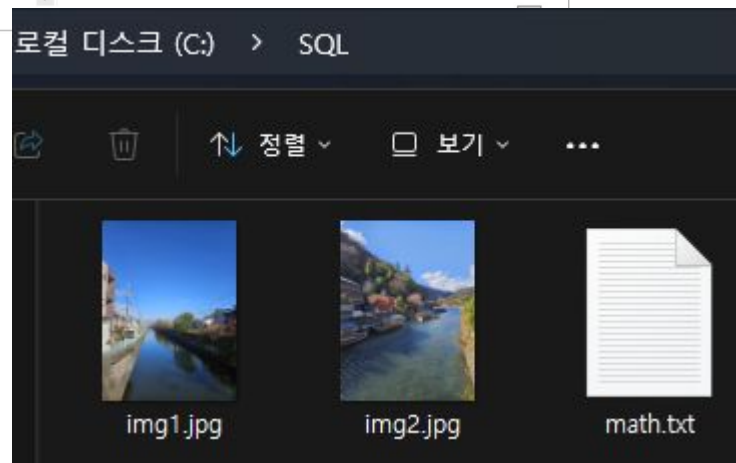
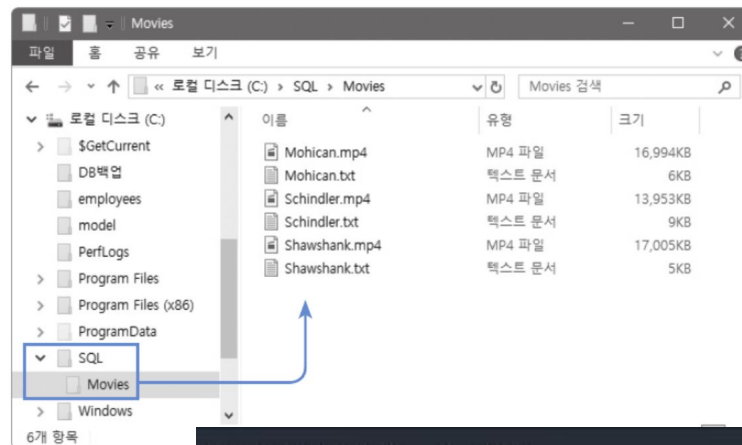
대용량 데이터

```
CREATE DATABASE moviedb;
```

```
USE moviedb;
```

```
CREATE TABLE movietbl
```

```
(  
    movie_id INT,  
    movie_title VARCHAR(30),  
    movie_director VARCHAR(20),  
    movie_star VARCHAR(20),  
    movie_script LONGTEXT,  
    movie_film LONGBLOB  
) DEFAULT CHARSET=utf8mb4;
```




```
INSERT INTO movietbl VALUES ( 1, '썬들러 리스트', '스필버그', '리암 니슨',  
    LOAD_FILE('C:/SQL/math.txt'), LOAD_FILE('C:/SQL/img1.jpg')  
);
```

18 • **SELECT** * **FROM** movietbl;

19

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

movie_id	movie_title	movie_director	movie_star	movie_script	movie_film
1	썬들러 리스트	스필버그	리암 니슨	NULL	NULL

```
SHOW variables LIKE 'max_allowed_packet';
```

	Variable_name	Value
▶	max_allowed_packet	4194304

```
SHOW variables LIKE 'secure_file_priv';
```

Variable_name	Value
secure_file_priv	C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\

설정 파일
변경하는
경우는
workbench stop
→ restart필요

```
my.ini
파일 편집 보기
innodb_checksum_algorithm=0

# If this is set to a nonzero value, all tables are closed every flush_time seconds.
# synchronize unflushed data to disk.
# This option is best used only on systems with minimal resources.
flush_time=0

# The minimum size of the buffer that is used for plain index scans,
# indexes and thus perform full table scans.
join_buffer_size=256K

# The maximum size of one packet or any generated or intermediate state
# mysql_stmt_send_long_data() C API function.
max_allowed_packet=1024M

# If more than this many successive connection requests from a host are
# connection,
# the server blocks that host from performing further connections.
max_connect_errors=100

# The number of file descriptors available to mysqld from the operating system.
# Try increasing the value of this option if mysqld gives the error
open_files_limit=8191
```

```
my.ini
파일 편집 보기
# This variable is used to limit the effect of data import and export
# those performed by the LOAD DATA and SELECT ... INTO OUTFILE state
# LOAD_FILE() function. These operations are permitted only to users
# secure-file-priv="C:/ProgramData/MySQL/MySQL Server 8.0/Uploads"
secure-file-priv="C:/SQL/Movies"
```

경로를 변경하거나, 해당 위치에
넣거나 선택 → 해당 위치에 파일을
 옮겨 넣을 예정이라 재시작 불필요

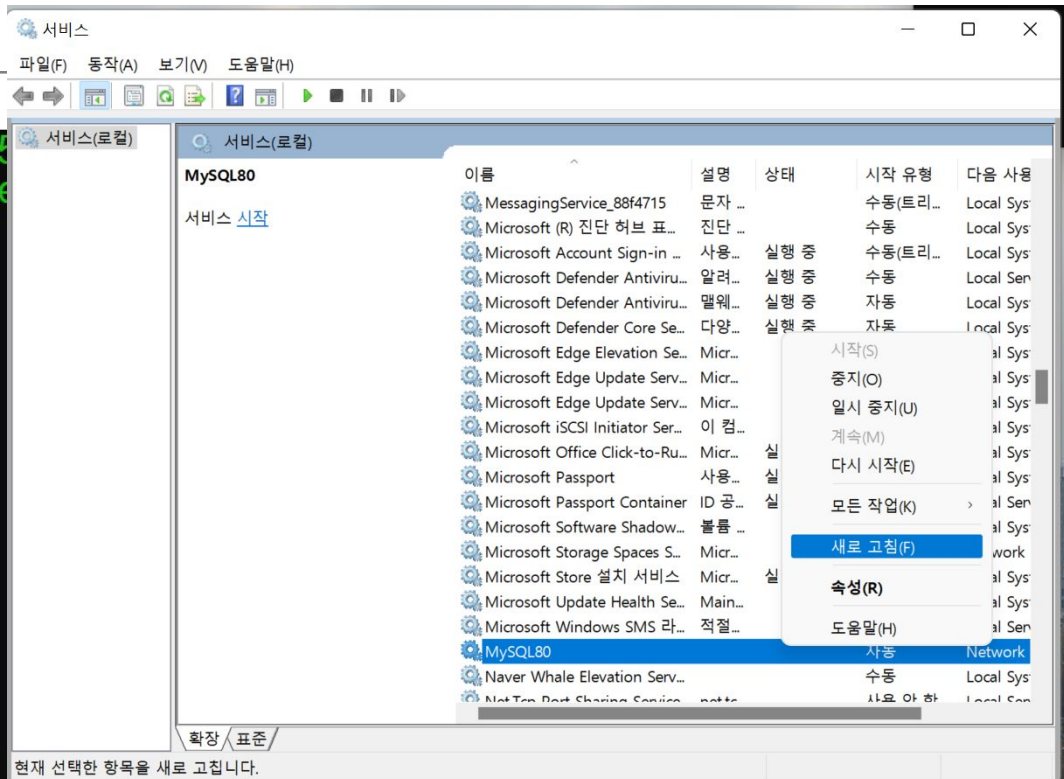
관리자: 명령 프롬프트

Microsoft Windows [Version 10.0.22000.2839]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>net stop mysql80
MySQL80 서비스를 멈춥니다...
MySQL80 서비스를 잘 멈추었습니다.

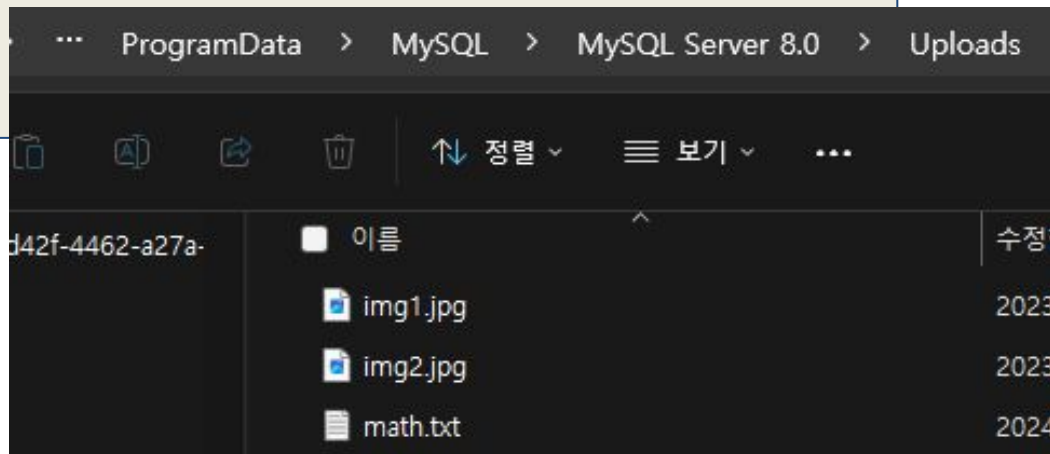
C:\Windows\system32>net start mysql80
MySQL80 서비스를 시작합니다..
MySQL80 서비스가 잘 시작되었습니다.

C:\Windows\system32>

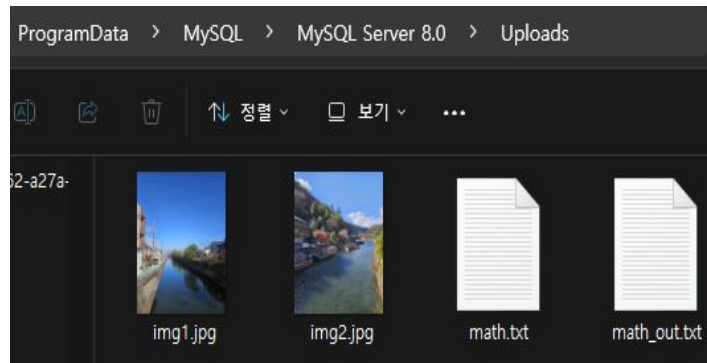
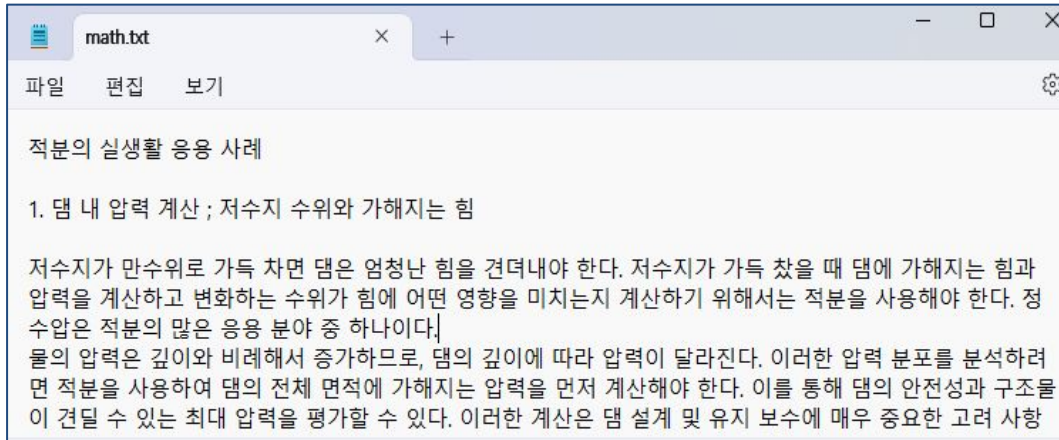


```
INSERT INTO movietbl VALUES ( 1, '썬들러 리스트', '스필버그', '리암 니슨',  
    LOAD_FILE('C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/math.txt'),  
    LOAD_FILE('C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/img1.jpg')  
);
```

```
SELECT * FROM movietbl;
```



Result Grid						
Filter Rows:						
Export:						
Wrap Cell Content: I A						
	movie_id	movie_title	movie_director	movie_star	movie_script	movie_film
▶	1	썬들러 리스트	스필버그	리암 니슨	NULL	NULL
	1	썬들러 리스트	스필버그	리암 니슨	적분의 실생활 응용 ...	BLOB



```
SELECT movie_script FROM movietbl WHERE movie_id=1  
INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/math_out.txt'  
LINES TERMINATED BY '\n';
```



```
SELECT movie_film FROM movietbl WHERE movie_id=3  
INTO DUMPFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/img1_out.jpg';
```

- 주요함수
 - 문자, 숫자, 날짜
 - 형식변환
- 그룹함수
 - `sum()`, `avg()`, `min()`, `max()`, `count()`
 - `group by`, `having`
- 대용량 데이터 다루기