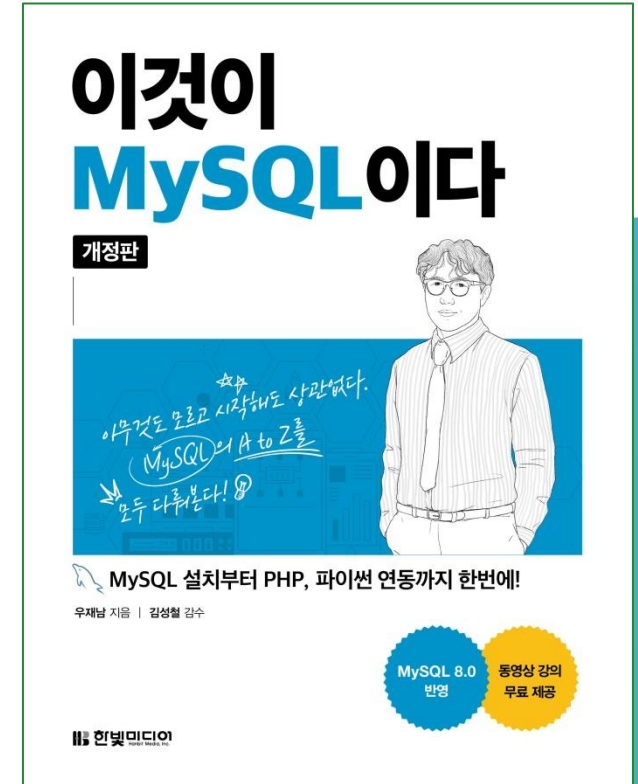


이것이 MySQL이다



저자 : 우재남

Contents

- CHAPTER 08 테이블과 뷰
 - SECTION 01 테이블
 - 1.1 테이블 만들기
 - 1.2 제약 조건
 - 1.3 테이블 압축
 - 1.4 임시 테이블
 - 1.5 테이블 삭제
 - 1.6 테이블 수정

Contents

- CHAPTER 08 테이블과 뷰
 - SECTION 02 뷰
 - 2.1 뷰의 개념
 - 2.2 뷰의 장점
 - SECTION 03 테이블스페이스
 - 3.1 테이블스페이스의 개념
 - 3.2 성능 향상을 위한 테이블스페이스 추가

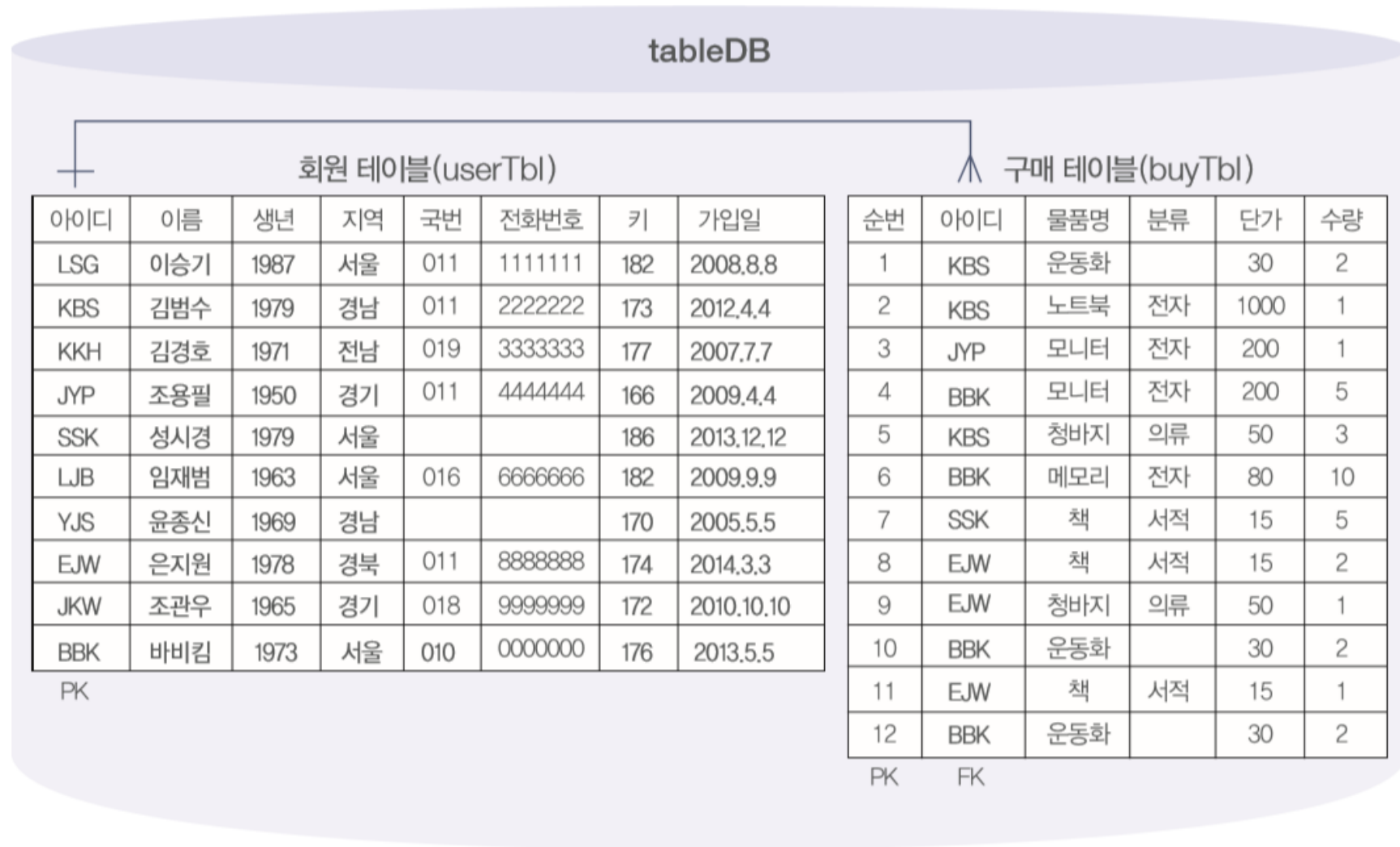


CHAPTER 08 테이블과 뷰

데이터베이스의 핵심 개체인 테이블과 가상의 테이블인 뷰에 대해서도 알아본다.

SECTION 01 테이블

테이블 만들기

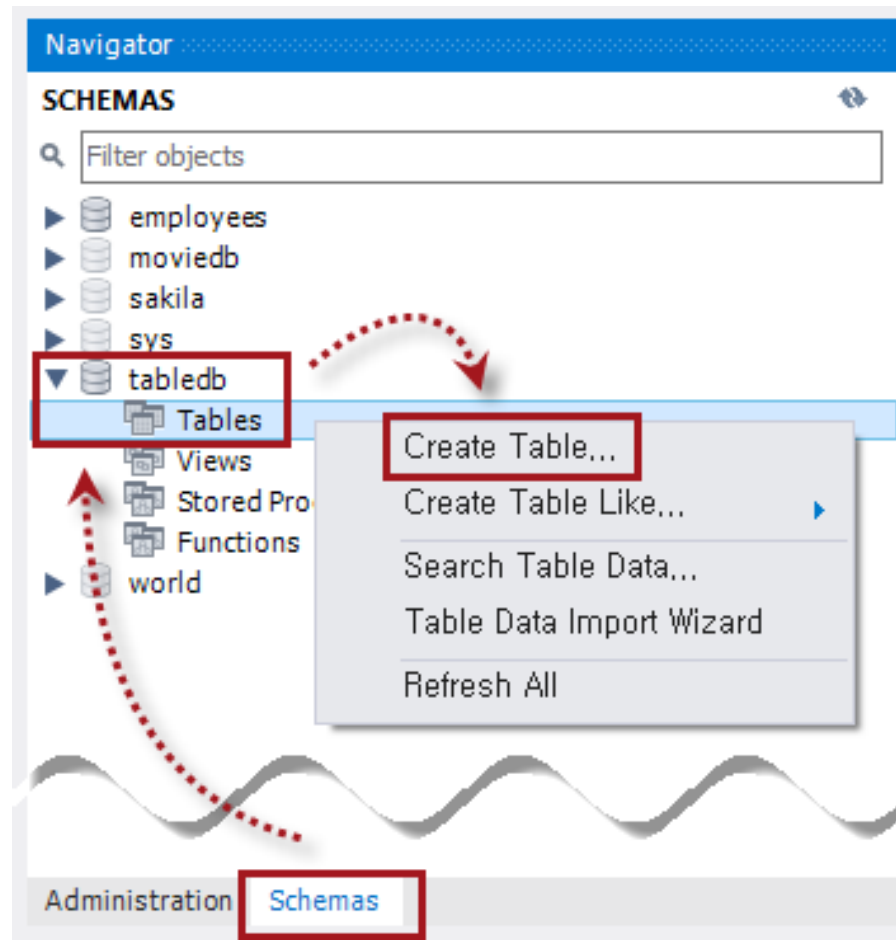


[그림 8-1] 샘플로 사용할 tabledb

SECTION 01 테이블

테이블 만들기

- MySQL Workbench에서 테이블 생성
 - Navigator – [Schemas] 클릭 – 'tabledb' 확장 – 'Tables' 마우스 오른쪽 버튼 – [Create Table] 선택



SECTION 01 테이블

테이블 만들기

- MySQL Workbench에서 테이블 생성
 - usertbl 생성
 - userID열을 기본 키(Primary Key)로 설정

The screenshot shows the MySQL Workbench Table Designer for a table named 'usertbl' in the 'tabledb' schema. The table is using the InnoDB engine. The columns are defined as follows:

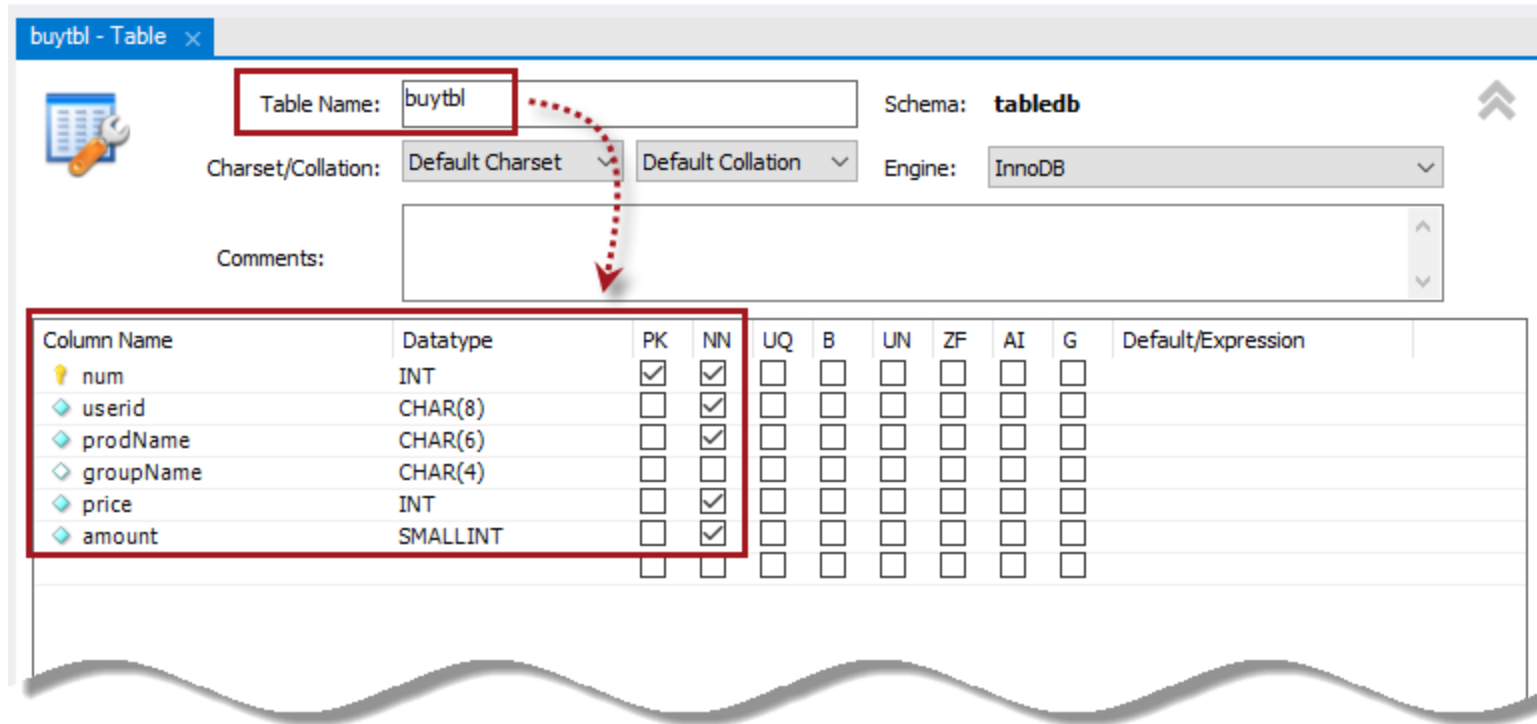
Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
userID	CHAR(8)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
name	VARCHAR(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
birthYear	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
addr	CHAR(2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
mobile1	CHAR(3)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
mobile2	CHAR(8)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
height	SMALLINT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
mDate	DATE	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

A red dashed arrow points from the 'Table Name' field to the 'PK' column in the table definition grid.

SECTION 01 테이블

테이블 만들기

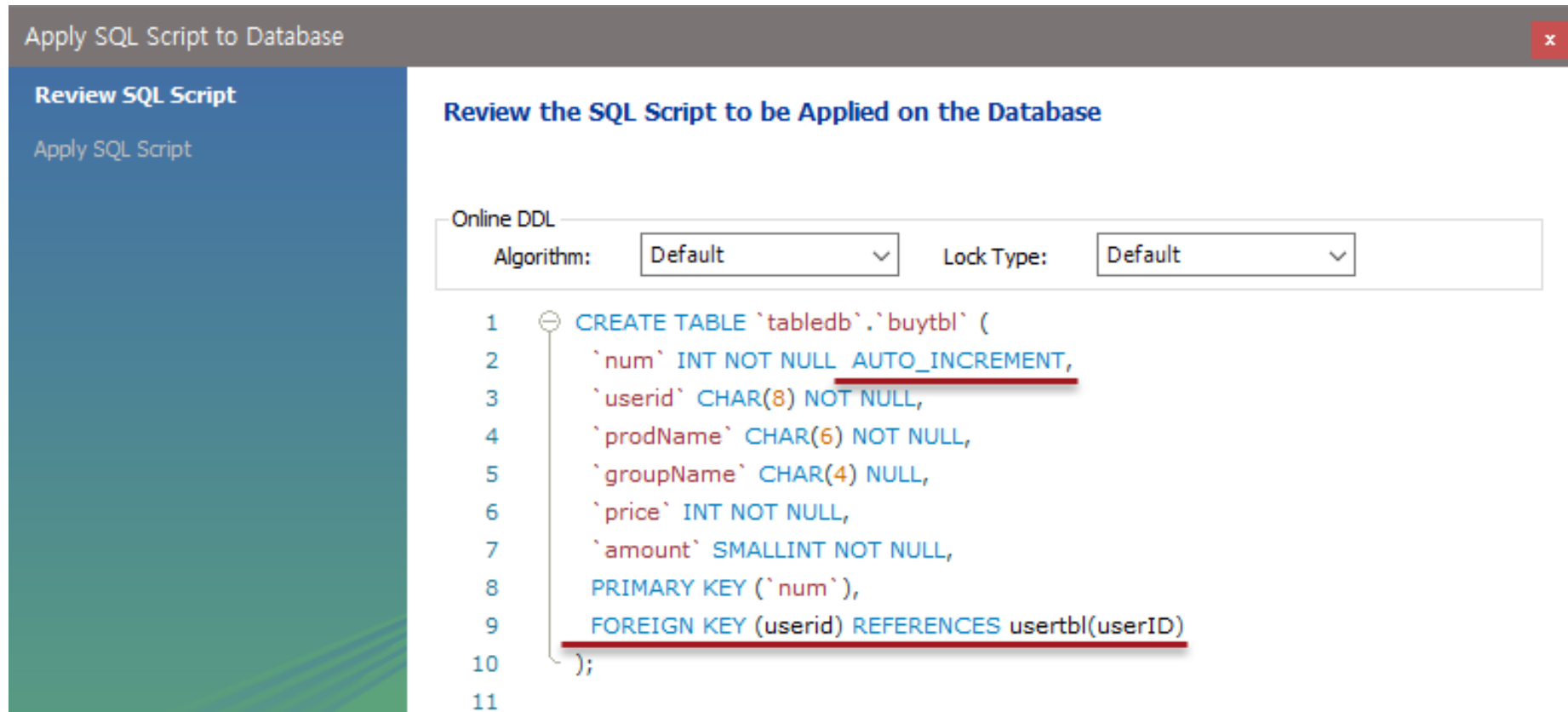
- MySQL Workbench에서 테이블 생성
 - buytbl 생성
 - num열을 기본 키(Primary Key)로 설정



SECTION 01 테이블

테이블 만들기

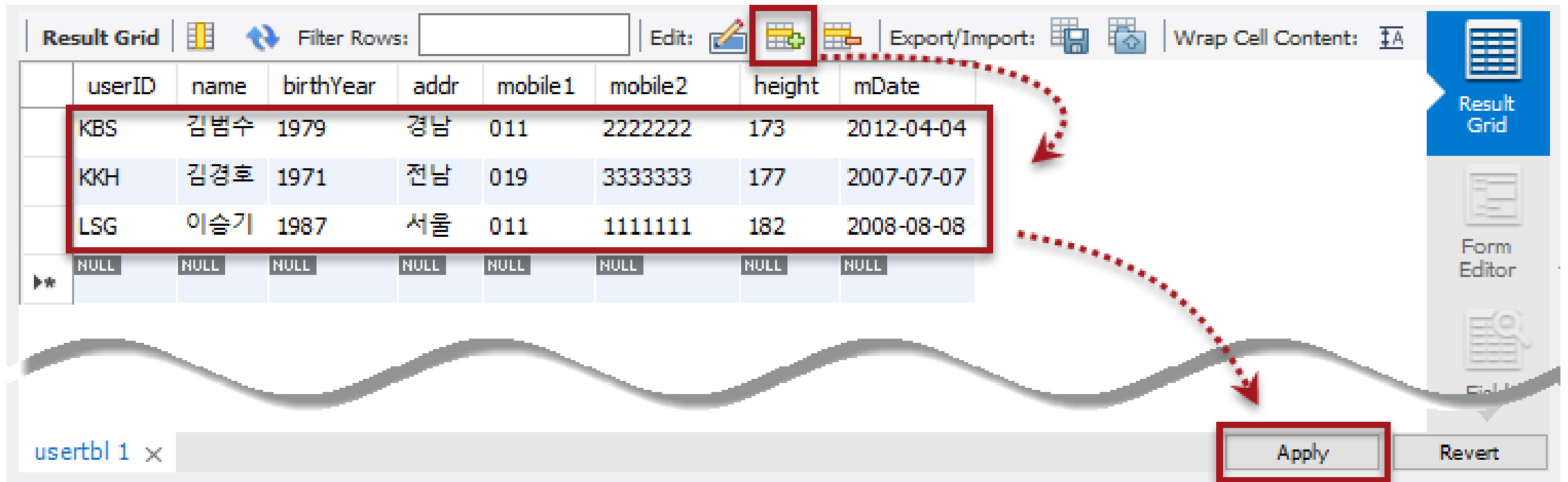
- MySQL Workbench에서 테이블 생성
 - buytbl 생성
 - num열에 AUTO_INCREMENT, FOREIGN KEY 추가



SECTION 01 테이블

테이블 만들기

- MySQL Workbench에서 데이터 입력
 - Navigator에서 usertbl 선택 – 마우스 오른쪽 버튼 클릭 – [Select Rows – Limit 1000] 선택
 - <Insert new row> 아이콘 클릭한 후, 3개 행 입력 - <Apply> 클릭 - <Finish> 클릭



SECTION 01 테이블

테이블 만들기

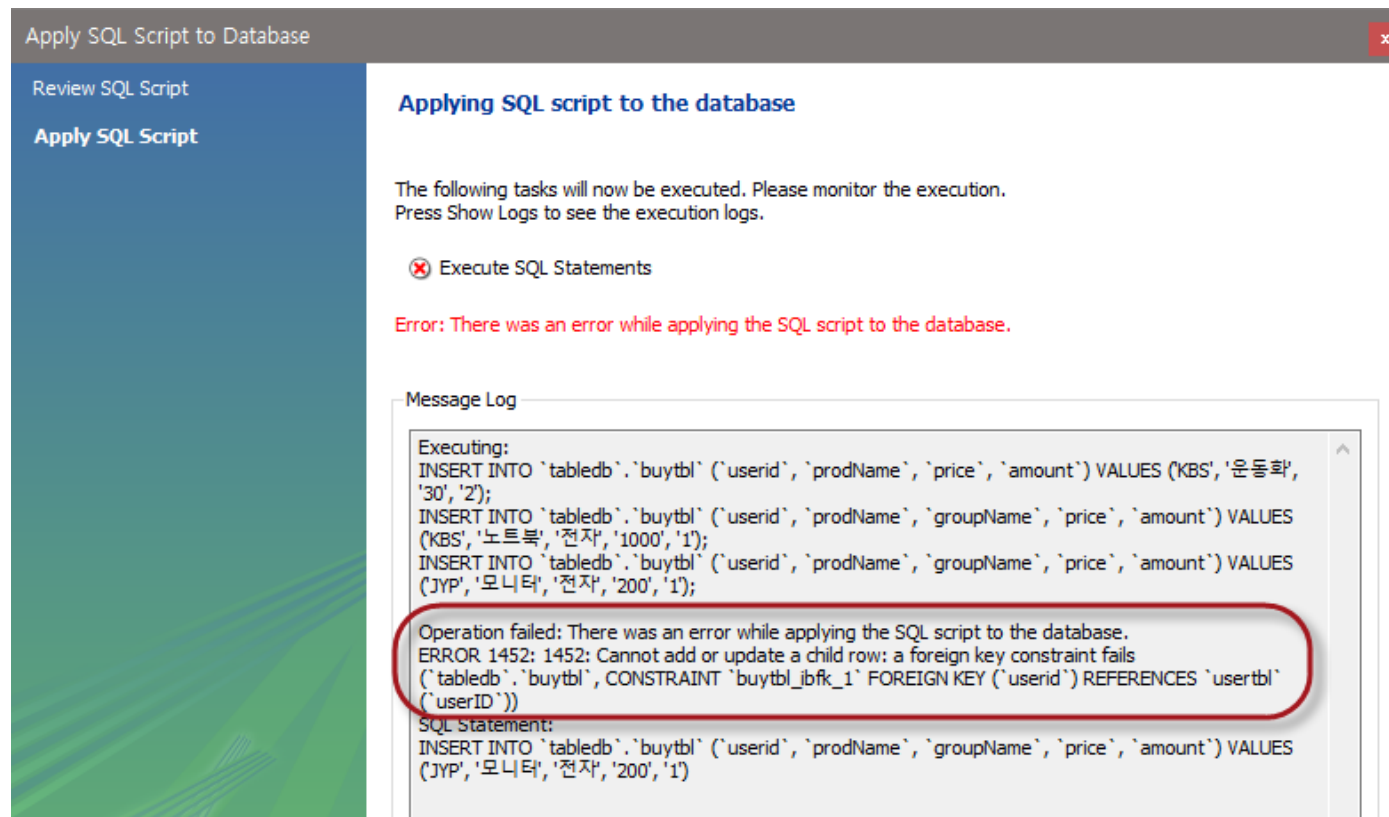
- MySQL Workbench에서 데이터 입력
 - buytbl 선택 – 마우스 오른쪽 버튼 클릭 – [Select Rows – Limit 1000] 선택
 - <Insert new row> 아이콘 클릭한 후, 3개 행 입력
 - num열은 자동 입력되니 NULL 값은 그대로 둬 - <Apply> 클릭

	num	userid	prodName	groupName	price	amount
	NULL	KBS	운동화	NULL	30	2
	NULL	KBS	노트북	전자	1000	1
▶	NULL	JYP	모니터	전자	200	1
✱	NULL	NULL	NULL	NULL	NULL	NULL

SECTION 01 테이블

테이블 만들기

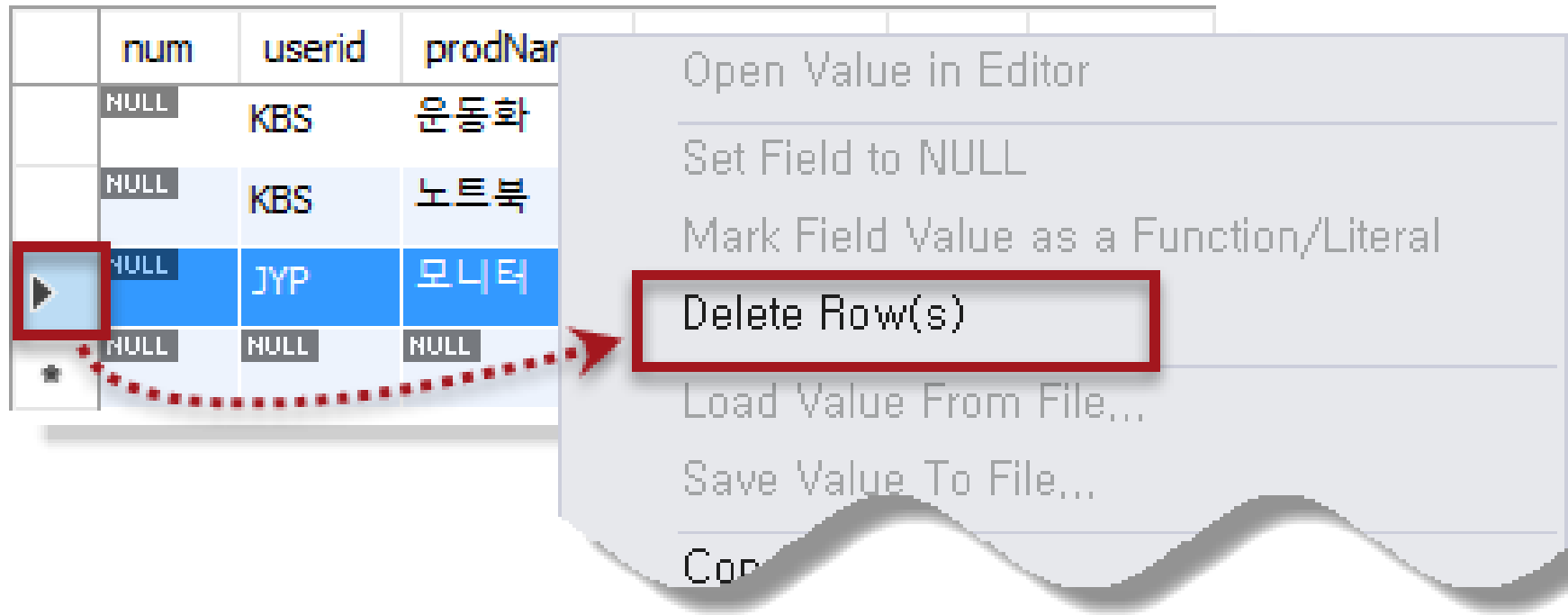
- MySQL Workbench에서 데이터 입력
 - 오류 메시지 이해
 - ex) 구매 테이블의 외래 키로 설정된 userid에 데이터가 입력되기 위해서는 입력될 값이 회원 테이블의 userid열에 존재해야 한다는 사항 이해



SECTION 01 테이블

테이블 만들기

- MySQL Workbench에서 데이터 입력
 - JYP 열 선택 - 마우스 오른쪽 버튼 - [Delete Row(s)] 선택
 - <Apply> 클릭 - <Finish> 클릭
 - 문제없이 입력 됨



SECTION 01 테이블

테이블 만들기

- SQL로 테이블 생성
 - 열린 창을 모두 닫고 쿼리 창을 연다.
 - 앞의 실습에서 사용한 tabledb를 삭제하고 다시 생성
 - DROP DATABASE tabledb;
 - CREATE DATABASE tabledb;

SECTION 01 테이블

테이블 만들기

- SQL로 테이블 생성
 - usertbl 생성

```
CREATE TABLE buytbl
(  num INT AUTO_INCREMENT NOT NULL PRIMARY KEY,
   userid CHAR(8) NOT NULL ,
   prodName CHAR(6) NOT NULL,
   groupName CHAR(4) NULL ,
   price      INT NOT NULL,
   amount     SMALLINT NOT NULL
);
```

SECTION 01 테이블

테이블 만들기

- SQL로 테이블 생성
 - buytbl 생성

```
CREATE TABLE buytbl
(  num INT AUTO_INCREMENT NOT NULL PRIMARY KEY,
   userid CHAR(8) NOT NULL ,
   prodName CHAR(6) NOT NULL,
   groupName CHAR(4) NULL ,
   price      INT NOT NULL,
   amount     SMALLINT NOT NULL
   , FOREIGN KEY(userid) REFERENCES usertbl(userID)
);
```


SECTION 01 테이블

테이블 만들기

- SQL로 테이블 생성

- 회원 테이블 데이터 입력

- INSERT INTO usertbl VALUES('LSG', '이승기', 1987, '서울', '011', '1111111', 182, '2008-8-8');
 - INSERT INTO usertbl VALUES('KBS', '김범수', 1979, '경남', '011', '2222222', 173, '2012-4-4');
 - INSERT INTO usertbl VALUES('KKH', '김경호', 1971, '전남', '019', '3333333', 177, '2007-7-7');

- 구매 테이블 데이터 입력

- INSERT INTO buytbl VALUES(NULL, 'KBS', '운동화', NULL, 30, 2);
 - INSERT INTO buytbl VALUES(NULL, 'KBS', '노트북', '전자', 1000, 1);
 - INSERT INTO buytbl VALUES(NULL, 'JYP', '모니터', '전자', 200, 1);

- 구매 테이블 데이터 입력시 3번째 행은 앞과 같이 에러 발생하므로 삭제하고 입력

SECTION 01 테이블

제약 조건

- 제약 조건(Constraint)이란?
 - 데이터의 무결성을 지키기 위한 제한된 조건 의미
 - 특정 데이터를 입력 시 어떠한 조건을 만족했을 때에 입력되도록 제약
 - ex) 동일한 아이디로 다시 회원 가입이 안되는 것
 - 데이터 무결성을 위한 제약조건
 - PRIMARY KEY 제약 조건
 - FOREIGN KEY 제약 조건
 - UNIQUE 제약 조건
 - CHECK 제약 조건(MySQL 8.0.16부터 지원)
 - DEFAULT 정의
 - NULL 값 허용

SECTION 01 테이블

데이터 무결성을 위한 제약 조건

- 기본 키(Primary Key) 제약 조건
 - 기본 키(Primary Key) 란?
 - 테이블에 존재하는 많은 행의 데이터를 구분할 수 있는 식별자
 - 중복이나 NULL값이 입력될 수 없음
 - ex) 회원 테이블의 회원 아이디, 학생 테이블이 학번
 - 기본 키로 생성한 것은 자동으로 클러스터형 인덱스 생성
 - 테이블에서는 기본 키를 하나 이상 열에 설정 가능
 - 기본 키 생성 방법

```
DROP TABLE IF EXISTS userTBL;  
CREATE TABLE userTBL  
( userID CHAR(8) NOT NULL,  
  name VARCHAR(10) NOT NULL,  
  birthYear INT NOT NULL,  
  CONSTRAINT PRIMARY KEY PK_userTBL_userID (userID)  
);
```

SECTION 01 테이블

데이터 무결성을 위한 제약 조건

- 기본 키(Primary Key) 제약 조건
 - ex) 제품 테이블
 - 기본 키 = 제품코드 + 제품일련번호

제품 코드	제품 일련 번호	제조일자	현 상태
AAA	0001	2023.10.10	판매완료
AAA	0002	2023.10.11	매장진열
BBB	0001	2023.10.12	재고창고
CCC	0001	2023.10.13	판매완료
CCC	0002	2023.10.14	매장진열

SECTION 01 테이블

데이터 무결성을 위한 제약 조건

- 기본 키(Primary Key) 제약 조건

- ex) 제품 테이블

```
DROP TABLE IF EXISTS prodTbl;  
CREATE TABLE prodTbl  
( prodCode CHAR(3) NOT NULL,  
  prodID   CHAR(4)  NOT NULL,  
  prodDate DATETIME NOT NULL,  
  prodCur CHAR(10) NULL,  
  CONSTRAINT PK_prodTbl_proCode_prodID  
    PRIMARY KEY (prodCode, prodID)  
);
```

- SHOW INDEX FROM prodTbl;

	Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
▶	prodtbl	0	PRIMARY	1	prodCode	A	0	NULL	NULL		BTREE		
	prodtbl	0	PRIMARY	2	prodID	A	0	NULL	NULL		BTREE		

SECTION 01 테이블

데이터 무결성을 위한 제약 조건

- 외래 키(Foreign Key) 제약 조건
 - 두 테이블 사이의 관계 선언하여 데이터의 무결성 보장해주는 역할
 - 외래 키 관계를 설정하면 하나의 테이블이 다른 테이블에 의존
 - 외래 키 테이블이 참조하는 기준 테이블의 열은 반드시 **Primary Key**이거나 **Unique 제약 조건**이 설정되어 있어야 함
 - 외래 키의 옵션 중 ON DELETE CASCADE 또는 ON UPDATE CASCADE
 - 기준 테이블의 데이터가 변경되었을 때 외래 키 테이블도 자동으로 적용되도록 설정

SECTION 01 테이블

데이터 무결성을 위한 제약 조건

- 외래 키(Foreign Key) 제약 조건
 - 외래 키 생성 방법 1
 - CREATE TABLE 끝에 FOREIGN KEY 키워드로 설정

```
DROP TABLE IF EXISTS buyTBL, userTBL;
CREATE TABLE userTBL
( userID CHAR(8) NOT NULL PRIMARY KEY,
  name   VARCHAR(10) NOT NULL,
  birthYear INT NOT NULL
);
CREATE TABLE buyTBL
( num INT AUTO_INCREMENT NOT NULL PRIMARY KEY ,
  userID CHAR(8) NOT NULL,
  prodName CHAR(6) NOT NULL,
  FOREIGN KEY(userID) REFERENCES userTBL(userID)
);
```

SECTION 01 테이블

데이터 무결성을 위한 제약 조건

- 외래 키(Foreign Key) 제약 조건
 - 외래 키 생성 방법 2
 - ALTER TABLE 구문 이용

```
DROP TABLE IF EXISTS buyTBL;
CREATE TABLE buyTBL
( num INT AUTO_INCREMENT NOT NULL PRIMARY KEY,
  userID CHAR(8) NOT NULL,
  prodName CHAR(6) NOT NULL
);
ALTER TABLE buyTBL
  ADD CONSTRAINT FK_userTBL_buyTBL
  FOREIGN KEY (userID)
  REFERENCES userTBL(userID);
```


SECTION 01 테이블

데이터 무결성을 위한 제약 조건

- UNIQUE 제약 조건
 - '중복되지 않는 유일한 값'을 입력해야 하는 조건
 - PRIMARY KEY와 비슷하나 UNIQUE는 **NULL 값 허용**
 - NULL은 여러 개가 입력되어도 상관 없음
 - ex) 회원 테이블 Email 주소 Unique로 설정

```
USE tableDB;
DROP TABLE IF EXISTS buyTBL, userTBL;
CREATE TABLE userTBL
( userID CHAR(8) NOT NULL PRIMARY KEY,
  name VARCHAR(10) NOT NULL,
  birthYear INT NOT NULL,
  email CHAR(30) NULL UNIQUE
);
```

SECTION 01 테이블

데이터 무결성을 위한 제약 조건

◦ CHECK 제약 조건

- 입력되는 데이터를 점검하는 기능
 - ex) 키(Height) 제한 - 마이너스 값이 들어올수 없도록,
 - 출생년도 제한 - 1900년 이후이고 현재시점 이전
- ALTER TABLE문으로 제약 조건 추가 가능

```
-- 출생년도가 1900년 이후 그리고 2023년 이전, 이름은 반드시 넣어야 함.  
DROP TABLE IF EXISTS userTBL;  
CREATE TABLE userTBL  
( userID CHAR(8) PRIMARY KEY,  
  name VARCHAR(10) ,  
  birthYear INT CHECK (birthYear >= 1900 AND birthYear <= 2023),  
  mobile1 char(3) NULL,  
  CONSTRAINT CK_name CHECK ( name IS NOT NULL)  
);
```

SECTION 01 테이블

데이터 무결성을 위한 제약 조건

◦ DEFAULT 정의

- 값 입력하지 않았을 때 자동으로 입력되는 기본 값 정의하는 방법
- ALTER TABLE 사용 시에 열에 DEFAULT를 지정하기 위해서 ALTER COLUMN문 사용

```
DROP TABLE IF EXISTS userTBL;
CREATE TABLE userTBL
( userID      CHAR(8) NOT NULL PRIMARY KEY,
  name        VARCHAR(10) NOT NULL,
  birthYear   INT NOT NULL DEFAULT -1,
  addr        CHAR(2) NOT NULL DEFAULT '서울',
  mobile1     CHAR(3) NULL,
  mobile2     CHAR(8) NULL,
  height      SMALLINT NULL DEFAULT 170,
  mDate       DATE NULL
);
```

SECTION 01 테이블

데이터 무결성을 위한 제약 조건

- DEFAULT 정의
 - 디폴트 설정된 열에는 다음과 같은 방법으로 데이터 입력

```
-- default문은 DEFAULT로 설정된 값을 자동 입력한다.  
INSERT INTO usertbl VALUES ('LHL', '이혜리', default, default, '011', '1234567',  
    default, '2023.12.12');  
-- 열 이름이 명시되지 않으면 DEFAULT로 설정된 값을 자동 입력한다.  
INSERT INTO usertbl(userID, name) VALUES('KAY', '김아영');  
-- 값이 직접 명기되면 DEFAULT로 설정된 값은 무시된다.  
INSERT INTO usertbl VALUES ('WB', '원빈', 1982, '대전', '019', '9876543', 176,  
    '2020.5.5');  
SELECT * FROM usertbl;
```

	userID	name	birthYear	addr	mobile1	mobile2	height	mDate
▶	KAY	김아영	-1	서울	NULL	NULL	170	NULL
	LHL	이혜리	-1	서울	011	1234567	170	2023-12-12
	WB	원빈	1982	대전	019	9876543	176	2020-05-05

SECTION 01 테이블

데이터 무결성을 위한 제약 조건

- Null 값 허용

- NULL 값을 허용하려면 NULL을, 허용하지 않으려면 NOT NULL을 사용
- PRIMARY KEY가 설정된 열에는 생략하면 자동으로 NOT NULL
- NULL 값은 '아무 것도 없다'라는 의미, 공백(' ') 이나 0과 다름

SECTION 01 테이블

테이블 압축

- 압축 기능은 대용량 테이블의 공간 절약하는 효과
- MySQL 5.0부터 자체적으로 테이블 압축 기능 제공
- MySQL 8.0에서 내부적인 기능이 더욱 강화
- MySQL이 허용하는 최대 용량의 데이터도 오류 없이 압축 가능
-

SECTION 01 테이블

테이블 압축

- 예제 실습(P. 337~338)
 - 테스트용 DB 생성 – 동일한 열을 지닌 간단한 두 테이블 생성
 - 하나는 열 뒤에 ROW_FORMAT=COMPRESSED문을 붙여서 압축되도록 설정

```
CREATE DATABASE IF NOT EXISTS compressDB;  
USE compressDB;  
CREATE TABLE normalTBL( emp_no int , first_name VARCHAR(14));  
CREATE TABLE compressTBL( emp_no int , first_name VARCHAR(14))  
    ROW_FORMAT=COMPRESSED ;
```

SECTION 01 테이블

테이블 압축

- 예제 실습(P. 337~338)
 - 두 테이블에 데이터 30만 건 입력

```
INSERT INTO normalTbl
SELECT emp_no, first_name FROM employees.employees;
INSERT INTO compressTBL
SELECT emp_no, first_name FROM employees.employees;
```

- 쿼리 실행 결과

✓	289	22:14:54	INSERT INTO <u>normalTbl</u> SELECT emp_no, first_nam...	300024 row(s) affected Records: 300024 Duplicates:...	2.235 sec
✓	290	22:15:05	INSERT INTO <u>compressTBL</u> SELECT emp_no, first_...	300024 row(s) affected Records: 300024 Duplicates:...	5.796 sec

SECTION 01 테이블

테이블 압축

- 예제 실습(P. 337~338)
 - 두 테이블 상태 확인
 - SHOW TABLE STATUS FROM compressDB;

	Name	Engine	Version	Row_format	Rows	Avg_row_length	Data_length	Max_data_length	Index_length	Data_free	Auto_increment
▶	compresstbl	InnoDB	10	Compressed	299582	25	7626752	0	0	2097152	NULL
	normaltbl	InnoDB	10	Dynamic	299570	40	12075008	0	0	4194304	NULL

- 실습한 DB 제거
 - DROP DATABASE IF EXISTS compressDB;

SECTION 01 테이블

임시 테이블

- 임시로 잠깐 사용되는 테이블

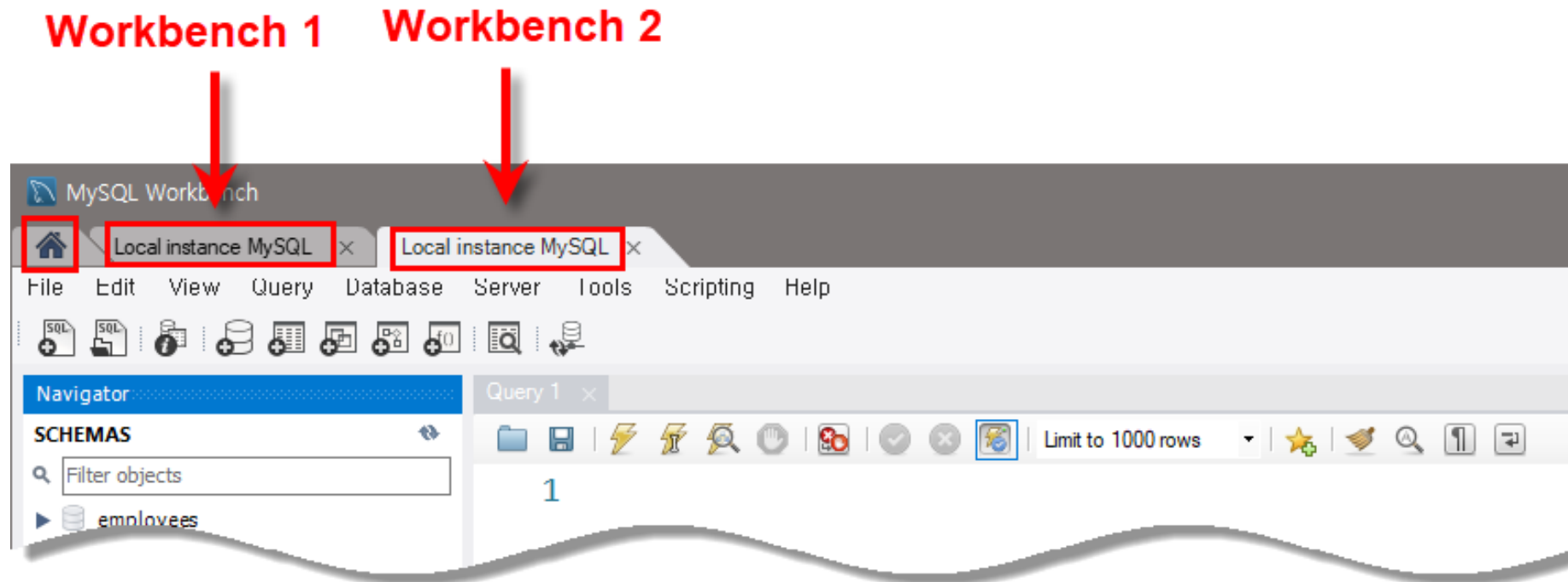
```
CREATE TEMPORARY TABLE [IF NOT EXISTS] 테이블이름  
( 열 정의 ... )
```

- 세션(Session) 내에서만 존재
 - 세션이 닫히면 자동 삭제
- 생성한 클라이언트에서만 접근 가능
 - 다른 클라이언트에는 접근 불가
- 임시 테이블 삭제 시점
 - 사용자가 DROP TABLE로 직접 삭제
 - Workbench를 종료하거나 mysql 클라이언트를 종료하면 삭제됨
 - MySQL 서비스가 재시작되면 삭제됨

SECTION 01 테이블

임시 테이블

- 예제 실습 (P. 340 ~ 341)
 - Workbench 실행 – [Local instance MySQL] 접속
 - 왼쪽 상단 [Home] 탭 클릭 - [Local instance MySQL] 접속



SECTION 01 테이블

임시 테이블

- 예제 실습 (P. 340 ~ 341)
 - 임시 테이블 2개 생성
 - 두 번째는 기존의 employees 테이블과 동일한 이름으로 생성

USE employees;

```
CREATE TEMPORARY TABLE IF NOT EXISTS temptbl (id INT, name CHAR(5));
```

```
CREATE TEMPORARY TABLE IF NOT EXISTS employees (id INT, name CHAR(5));
```

```
DESCRIBE temptbl;
```

```
DESCRIBE employees;
```

	Field	Type	Null	Key	Default	Extra
▶	id	int(11)	YES		NULL	
	name	char(5)	YES		NULL	

SECTION 01 테이블

임시 테이블

- 예제 실습 (P. 340 ~ 341)
 - 데이터 입력하고 확인

```
INSERT INTO temptbl VALUES (1, 'This');  
INSERT INTO employees VALUES (2, 'MySQL');  
SELECT * FROM temptbl;  
SELECT * FROM employees;
```

	id	name
▶	1	This

	id	name
▶	2	MySQL

SECTION 01 테이블

임시 테이블

- 예제 실습 (P. 340 ~ 341)
 - (Workbench 2) Workbench 1에서 생성한 테이블에 접근

```
USE employees;  
SELECT * FROM temptbl;  
SELECT * FROM employees;
```

- tempTBL은 아예 그런 테이블이 없다는 오류 메시지 나옴,
- 세션이 다르면 임시 테이블에 접근할 수 없음
- employees 테이블은 기존의 employees 테이블이 접근해옴
- 임시 테이블 employees 접근할 수 없음

SECTION 01 테이블

임시 테이블

- 예제 실습 (P. 340 ~ 341)
 - (Workbench 1) 임시 테이블 삭제
 - DROP TABLE temptbl;
 - Workbench를 종료 후 다시 접속
 - 다음 쿼리로 확인 시 임시 테이블이 아닌 기존의 테이블 조회 됨
 - USE employees;
 - SELECT * FROM employees;

SECTION 01 테이블

테이블 삭제

```
DROP TABLE 테이블이름 ;
```

- 외래 키 제약 조건의 기준 테이블은 삭제할 수가 없음
 - 먼저 외래 키가 생성된 외래 키 테이블을 삭제해야 함
 - 구매 테이블이 존재하는데 회원 테이블을 삭제 할 수 없음, 구매 테이블 삭제가 선행 되어야 함
- 동시에 여러 테이블 삭제도 가능
 - DROP TABLE 테이블1, 테이블2, 테이블3;

SECTION 01 테이블

테이블 수정

- ALTER TABLE문 사용
 - 테이블에 무엇인가 추가/변경/수정/삭제 모두 ALTER TABLE문 사용
- 열의 추가
 - 기본적으로 가장 뒤에 추가
 - 순서를 지정하려면 제일 뒤에 'FIRST' 또는 'ALTER 열 이름' 지정
 - ex) 회원 테이블(usertbl)에 회원 홈페이지 주소 추가

```
USE tabledb;  
ALTER TABLE usertbl  
    ADD homepage VARCHAR(30)  -- 열 추가  
                                DEFAULT 'http://www.hanbit.co.kr' -- 디폴트 값  
                                NULL; -- Null 허용함
```

SECTION 01 테이블

테이블 수정

- 열의 삭제

```
ALTER TABLE usertbl  
  DROP COLUMN mobile1;
```

- 제약 조건이 걸린 열을 삭제할 경우 제약 조건을 먼저 삭제한 후에 열을 삭제해야 함

- 열의 이름 및 데이터 형식 변경

- ex) 회원 이름(name)의 열 이름을 uName으로 변경하고 데이터 형식을 VARCHAR(20)으로 변경, NULL 값도 허용하는 경우

```
ALTER TABLE usertbl  
  CHANGE COLUMN name uName VARCHAR(20) NULL ;
```

SECTION 01 테이블

테이블 수정

- 열의 제약 조건 추가 및 삭제
 - ex) 기본 키를 삭제 하는 경우

```
ALTER TABLE usertbl  
    DROP PRIMARY KEY;
```

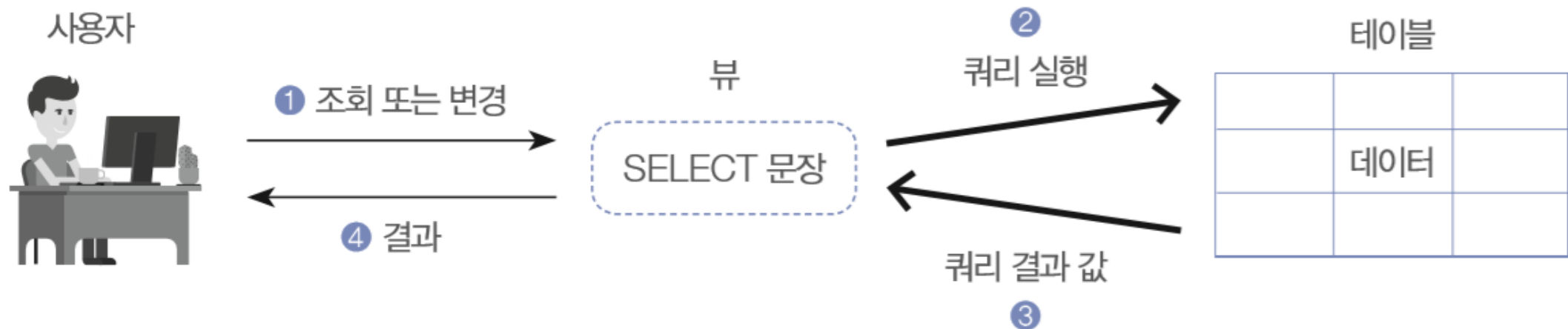
- 오류가 발생
 - usertbl의 기본 키인 userID열은 buytbl에 외래 키로 연결되어 있기 때문에 외래 키를 제거한 후에 다시 기본 키를 제거해야 함

SECTION 02 뷰

뷰의 개념

- 일반 사용자 입장에서 테이블과 동일하게 사용하는 개체
 - 뷰를 생성한 후에는 테이블처럼 접근 가능하여 동일한 결과 얻을수 있음

뷰의 작동 방식



[그림 8-27] 뷰의 작동 방식

SECTION 02 뷰

뷰의 개념

- 뷰 생성 구문

```
USE tabledb;  
CREATE VIEW v_usertbl  
AS  
    SELECT userid, name, addr FROM usertbl;
```

```
SELECT * FROM v_usertbl; -- 뷰를 테이블이라고 생각해도 무방
```

	userid	name	addr
▶	BBK	바비킴	서울
	EJW	은지원	경북
	JKW	조관우	경기
	JYP	조용필	경기
	KBS	김범수	경남
	KKH	김경호	전남
	LJB	임재범	서울
	LSG	이승기	서울
	SSK	성시경	서울
	YJS	윤종신	경남

SECTION 02 뷰

뷰의 장점

- 보안에 도움
 - 사용자가 중요한 정보에 바로 접근하지 못함
- 복잡한 쿼리 단순화
 - 긴 쿼리를 뷰로 작성, 뷰를 테이블처럼 사용 가능

```
CREATE VIEW v_userbuytbl
AS
SELECT U.userid, U.name, B.prodName, U.addr, CONCAT(U.mobile1, U.mobile2) AS '연락처'
FROM usertbl U
      INNER JOIN buytbl B
      ON U.userid = B.userid ;
```

```
SELECT * FROM v_userbuytbl WHERE name = '김범수';
```

	userid	name	prodName	addr	연락처
▶	KBS	김범수	운동화	경남	0112222222
	KBS	김범수	노트북	경남	0112222222
	KBS	김범수	청바지	경남	0112222222

SECTION 03 테이블스페이스

테이블스페이스의 개념

- 물리적인 공간을 뜻함
- 데이터베이스는 논리적 공간
- 테이블스페이스를 지정하지 않은 경우
 - 시스템 테이블스페이스(System Tablespace)에 테이블 저장됨
- 시스템 변수 `innodb_data_file_path`에 관련 내용 저장됨

```
SHOW VARIABLES LIKE 'innodb_data_file_path';
```

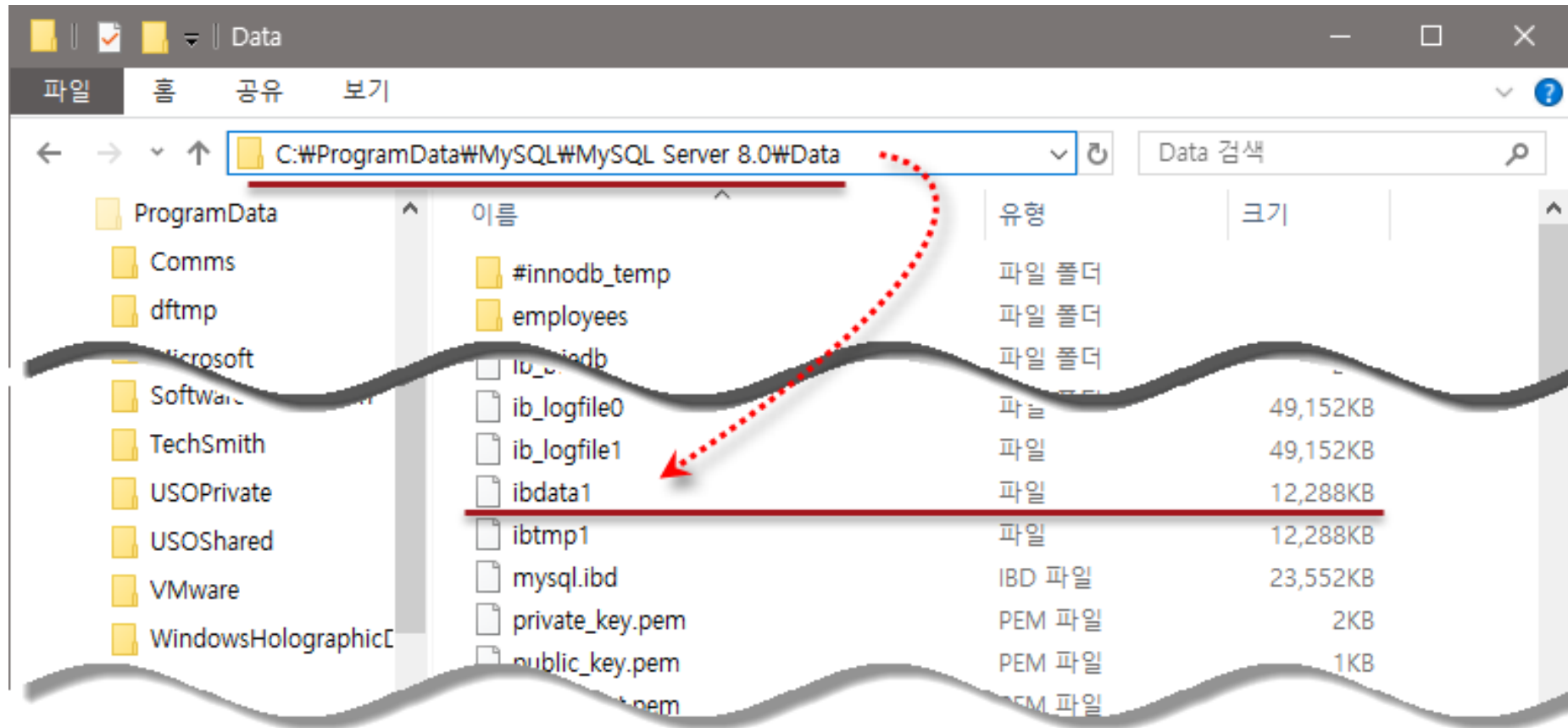
	Variable_name	Value
▶	<code>innodb_data_file_path</code>	<u><code>ibdata1:12M:autoextend</code></u>

SECTION 03 테이블스페이스

테이블스페이스의 개념

- 시스템 테이블 스페이스 파일 확인

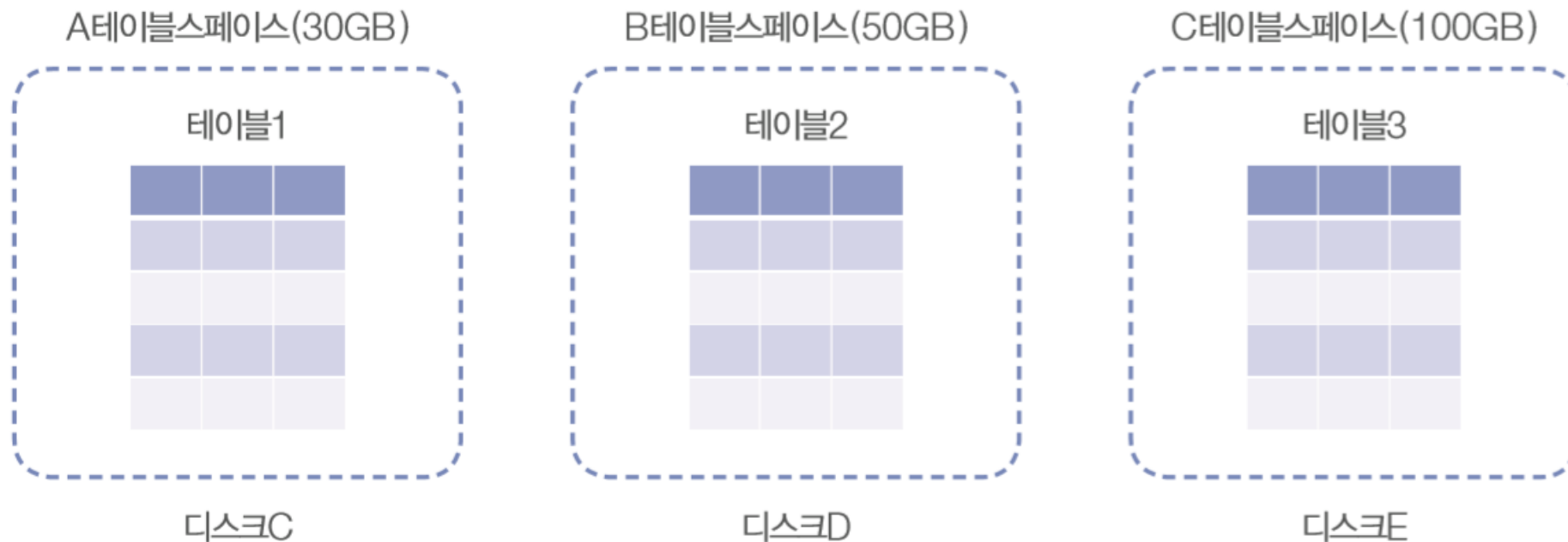
- MySQL 8.0에서 테이블스페이스 파일은 기본적으로 'C:\ProgramData\MySQL\MySQL Server 8.0\Data' 폴더에 저장 되어 있음



SECTION 03 테이블스페이스

성능 향상을 위한 테이블스페이스 추가

- 소용량의 데이터를 사용하는 경우에는 테이블스페이스 고려하지 않아도 되나 대용량의 데이터를 운영할 경우에는 성능 향상을 위해 테이블스페이스의 분리를 적극 고려



[그림 8-38] 테이블스페이스가 분리된 테이블

SECTION 03 테이블스페이스

성능 향상을 위한 테이블스페이스 추가

- 테이블스페이스 실습 (P.366 ~ 369)
 - 각 테이블이 별도의 테이블스페이스에 저장되도록 시스템 변수 `innodb_file_per_table`이 ON으로 설정되어야 함
 - 확인 방법
 - `SHOW VARIABLES LIKE 'innodb_file_per_table';`

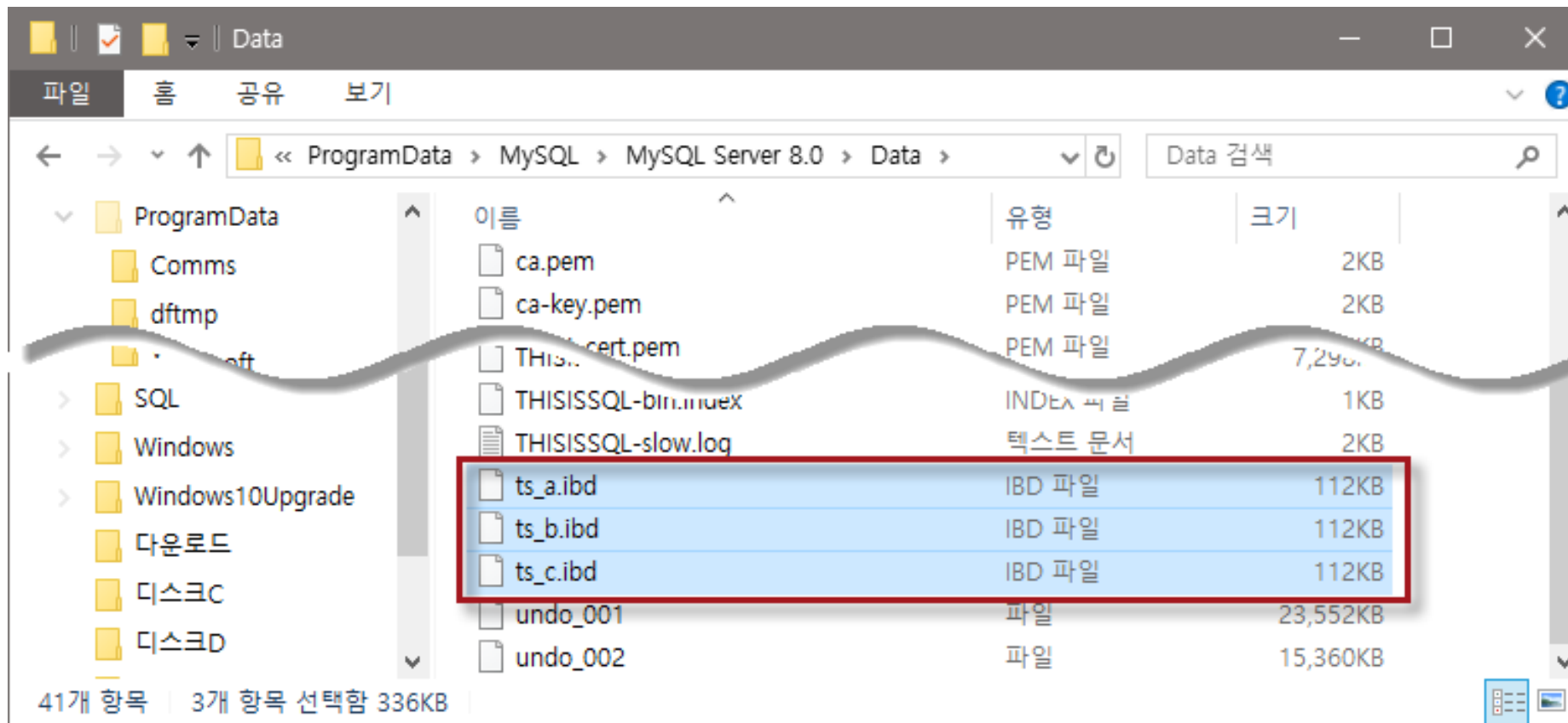
Variable_name	Value
innodb_file_per_table	ON

- 테이블 스페이스 3개 생성
 - `CREATE TABLESPACE ts_a ADD DATAFILE 'ts_a.ibd';`
 - `CREATE TABLESPACE ts_b ADD DATAFILE 'ts_b.ibd';`
 - `CREATE TABLESPACE ts_c ADD DATAFILE 'ts_c.ibd';`

SECTION 03 테이블스페이스

성능 향상을 위한 테이블스페이스 추가

- 테이블스페이스 실습 (P.366 ~ 369)
 - 파일 탐색기에서 'C:\Programdata\MySQL\MySQL Server 8.0\Data' 폴더 확인



SECTION 03 테이블스페이스

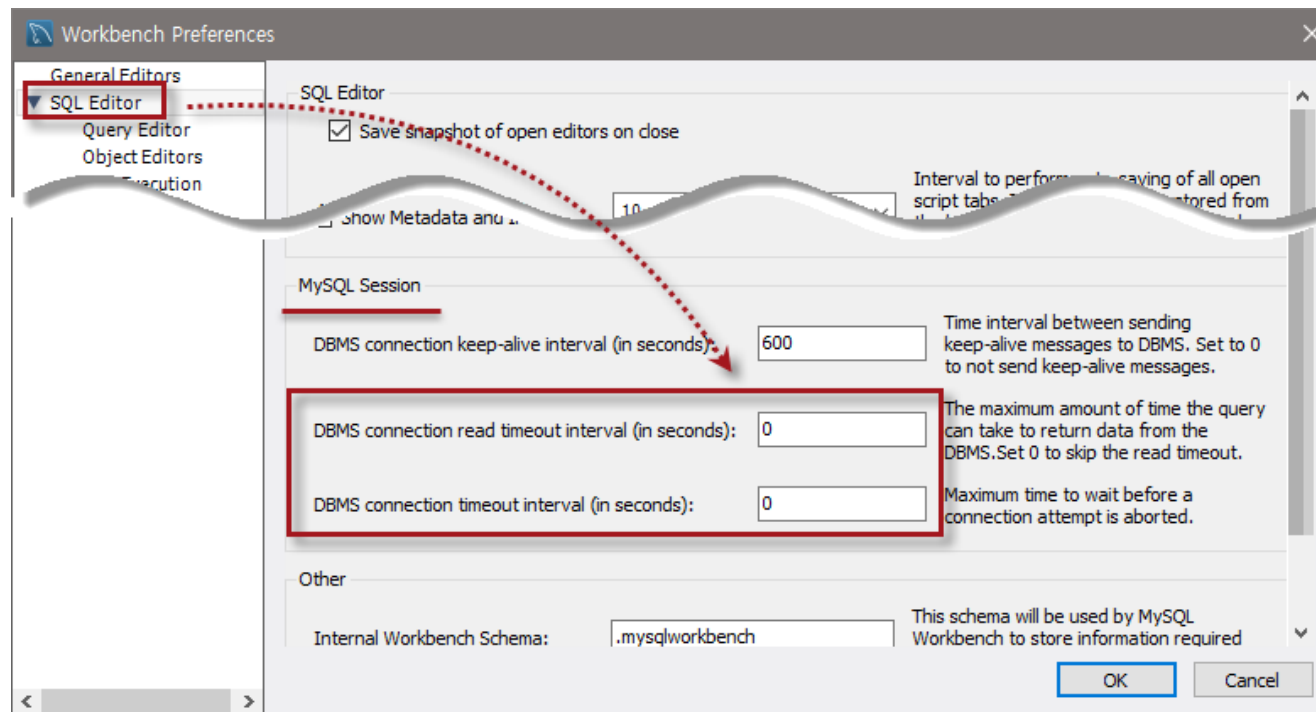
성능 향상을 위한 테이블스페이스 추가

- 테이블스페이스 실습 (P.366 ~ 369)
 - 각 테이블스페이스에 파일 생성
 - USE sqldb;
 - CREATE TABLE table_a (id INT) TABLESPACE ts_a;
 - 테이블을 만든 후에 ALTER TABLE문으로 테이블스페이스 변경 가능
 - CREATE TABLE table_b (id INT)
 - ALTER TABLE table_b TABLESPACE ts_b;

SECTION 03 테이블스페이스

성능 향상을 위한 테이블스페이스 추가

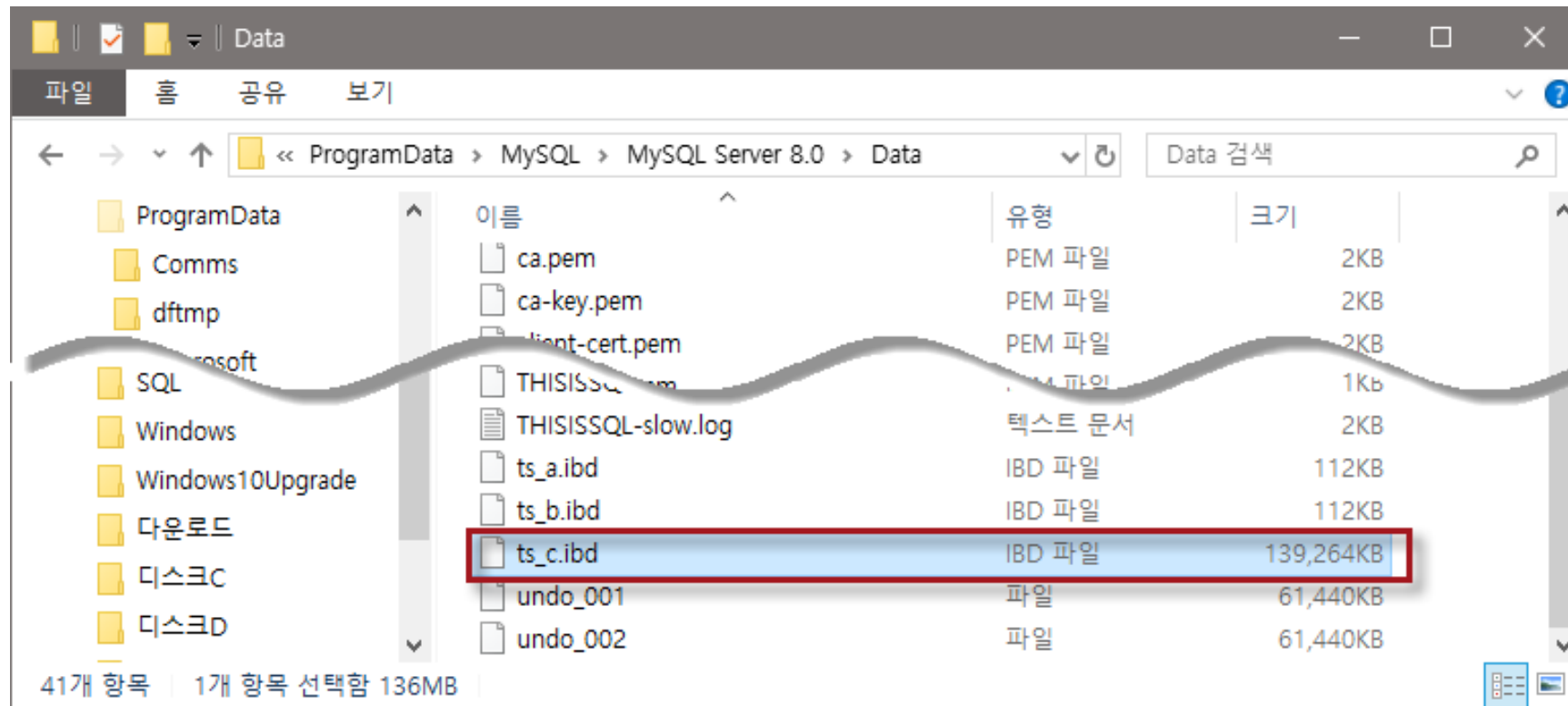
- 테이블스페이스 실습 (P.366 ~ 369)
 - 쿼리 응답 시간 제한 없애기
 - Workbench 메뉴의 [Edit] >> [Preferences]를 선택 - 왼쪽에서 [SQL Editor]를 선택
 - [MySQL Session] 부분의 'DBMS connection read timeout interval'와
 - 'DBMS connection timeout interval' 두 개를 0으로 설정 - <OK> 클릭



SECTION 03 테이블스페이스

성능 향상을 위한 테이블스페이스 추가

- 테이블스페이스 실습 (P.366 ~ 369)
 - 대용량의 테이블을 복사한 후 테이블 스페이스 지정
 - CREATE TABLE table_c (SELECT * FROM employees.salaries); ALTER TABLE table_c TABLESPACE ts_c;



▶ 이것이 MySQL 이다

Thank You!

