

● DBMS 개요와 MySQL 소개

- SECTION 01 DBMS 개요

- 1.1 데이터베이스의 정의와 특징

- 1.2 데이터베이스의 발전

- 1.3 DBMS 분류

- 1.4 SQL 개요

- SECTION 02 MySQL 소개

- 2.1 MySQL의 개요와 변천사

- SECTION 03 MySQL의 에디션 및 기능 비교

DBMS 개요와 MySQL 소개

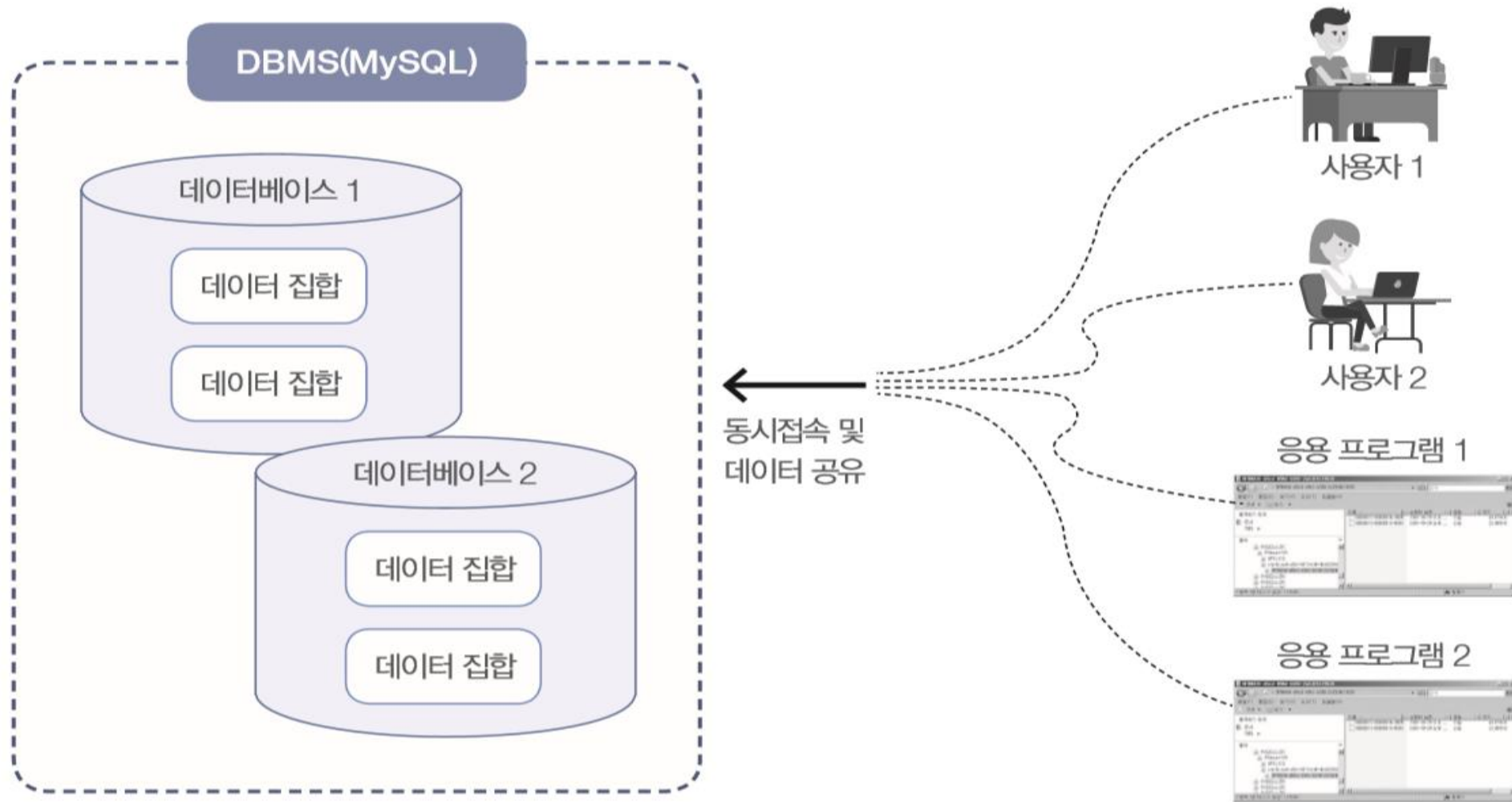
SECTION 01 DBMS 개요

데이터베이스의 정의와 특징

- 데이터베이스
 - '데이터의 집합'
 - 여러 명의 사용자나 응용프로그램이 공유하는 데이터들
 - 동시에 접근 가능해야
 - 데이터의 저장 공간' 자체
- DBMS
 - 데이터베이스를 관리·운영하는 역할

SECTION 01 DBMS 개요

● DBMS 개념도



[그림 1-1] DBMS 개념도

SECTION 01 DBMS 개요

DB/DBMS의 특징

- 데이터의 무결성 (Integrity)
 - 데이터베이스 안의 데이터는 오류가 없어야
 - 제약 조건(Constrain)이라는 특성을 가짐
- 데이터의 독립성
 - 데이터베이스 크기 변경하거나 데이터 파일의 저장소 변경시 기존에 작성된 응용프로그램은 전혀 영향을 받지 않아야
- 보안
 - 데이터베이스 안의 데이터에 데이터를 소유한 사람이나 데이터에 접근이 허가된 사람만 접근할 수 있어야
 - 접근할 때도 사용자의 계정에 따라서 다른 권한 가짐

SECTION 01 DBMS 개요

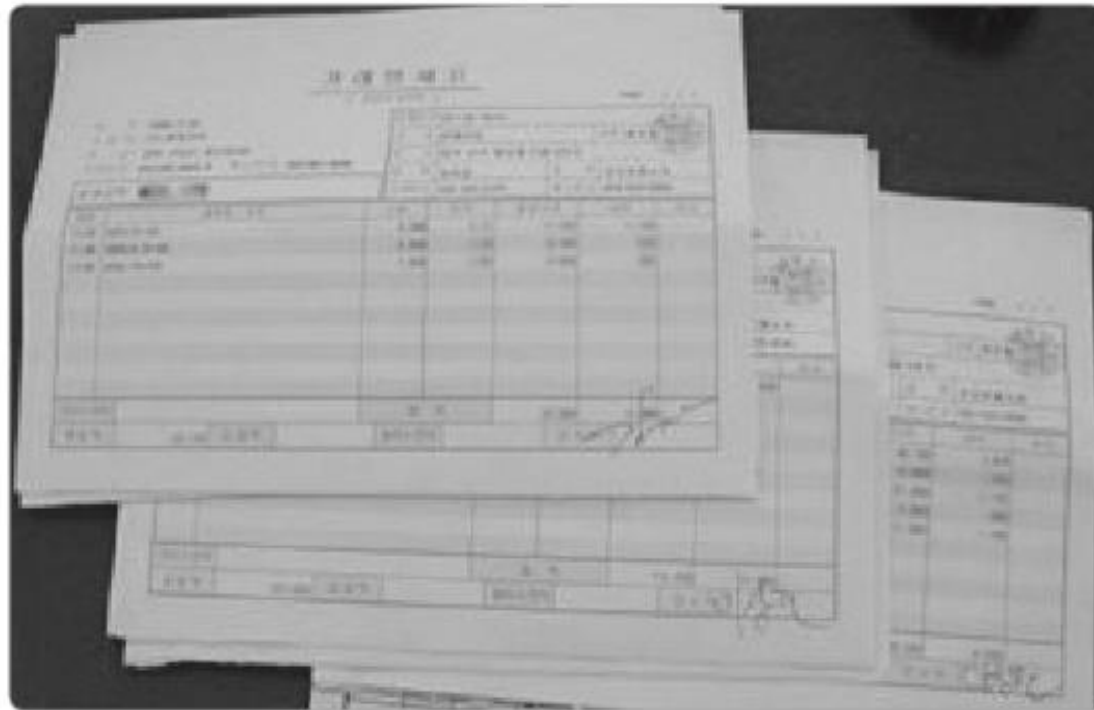
DB/DBMS의 특징

- 데이터 중복의 최소화
 - 동일한 데이터가 여러 개 중복되어 저장되는 것 방지
- 응용프로그램 제작 및 수정이 쉬워짐
 - 통일된 방식으로 응용프로그램 작성 가능
 - 유지보수 또한 쉬워짐
- 데이터의 안전성 향상
 - 대부분의 DBMS가 제공하는 백업·복원 기능 이용
 - 데이터가 깨지는 문제가 발생할 경우 원상으로 복원 , 복구하는 방법이 명확해짐

SECTION 01 DBMS 개요

데이터베이스의 발전

- 오프라인 관리
 - 종이에 연필로 기록해 장부로 관리



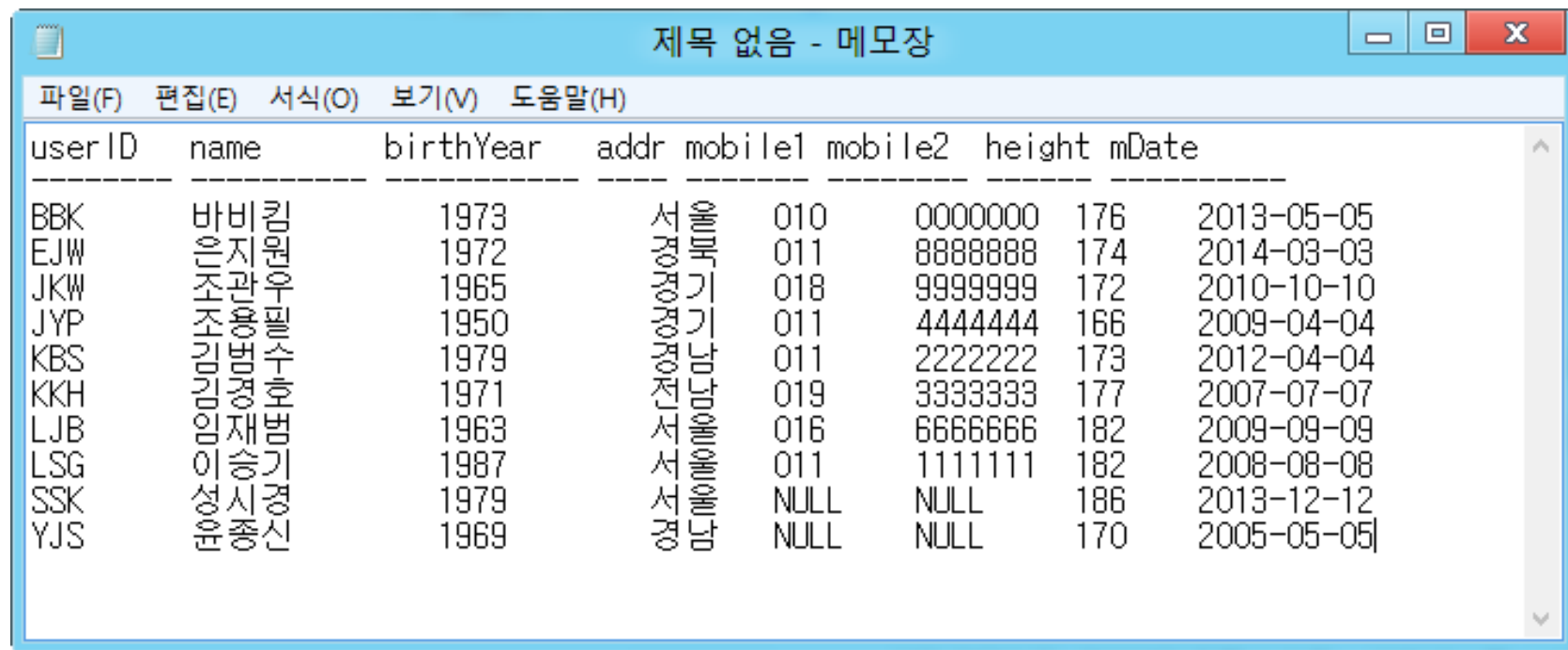
[그림 1-2] 종이 장부

SECTION 01 DBMS 개요

데이터베이스의 발전

◦ 파일시스템 사용

- 컴퓨터 파일에 기록/저장 - 메모장, 엑셀 활용
- 컴퓨터에 저장된 파일의 내용은 읽고, 쓰기가 편한 약속된 형태의 구조 사용
- 데이터의 양이 많아지면 데이터 중복으로 인한 불일치 위험



The image shows a screenshot of a Windows Notepad window titled "제목 없음 - 메모장". The window contains a table with 8 columns: userID, name, birthYear, addr, mobile1, mobile2, height, and mDate. The table lists 10 rows of user data. The text is formatted with dashed lines under the column headers.

userID	name	birthYear	addr	mobile1	mobile2	height	mDate
BBK	바비킴	1973	서울	010	0000000	176	2013-05-05
EJW	은지원	1972	경북	011	8888888	174	2014-03-03
JKW	조관우	1965	경기	018	9999999	172	2010-10-10
JYP	조용필	1950	경기	011	4444444	166	2009-04-04
KBS	김범수	1979	경남	011	2222222	173	2012-04-04
KKH	김경호	1971	전남	019	3333333	177	2007-07-07
LJB	임재범	1963	서울	016	6666666	182	2009-09-09
LSG	이승기	1987	서울	011	1111111	182	2008-08-08
SSK	성시경	1979	서울	NULL	NULL	186	2013-12-12
YJS	윤종신	1969	경남	NULL	NULL	170	2005-05-05

SECTION 01 DBMS 개요

데이터베이스의 발전

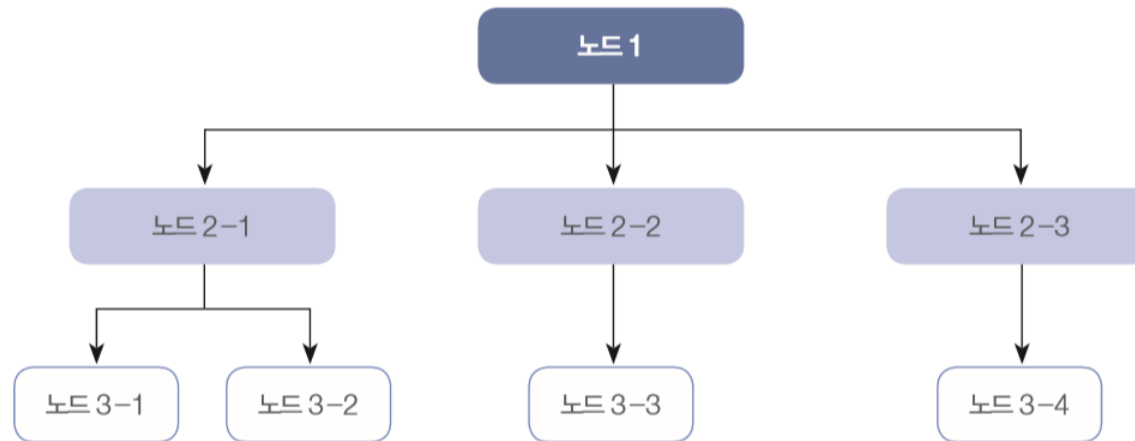
- 데이터베이스 관리시스템
 - 파일시스템의 단점 보완
 - 대량의 데이터를 보다 효율적으로 관리하고 운영하기 위해 사용
 - DBMS - DataBase Management System
 - 데이터의 집합인 '데이터베이스' 를 잘 관리하고 운영하기 위한 시스템 또는 소프트웨어
- SQL(Structured Query Language)
 - DBMS에 데이터 구축/관리/활용 위해서 사용되는 언어
 - DBMS를 통해 중요한 정보들을 입력, 관리, 추출

SECTION 01 DBMS 개요

DBMS 분류

◦ 계층형 DBMS

- 처음으로 나온 DBMS 개념 - 1960년대에 시작
- 각 계층은 트리Tree 형태, 1:N 관계
- 문제점
 - 처음 구축한 이후 그 구조를 변경하기가 상당히 까다로움
 - 주어진 상태에서의 검색은 상당히 빠름
 - 접근 유연성 부족해서 임의의 검색에는 어려움



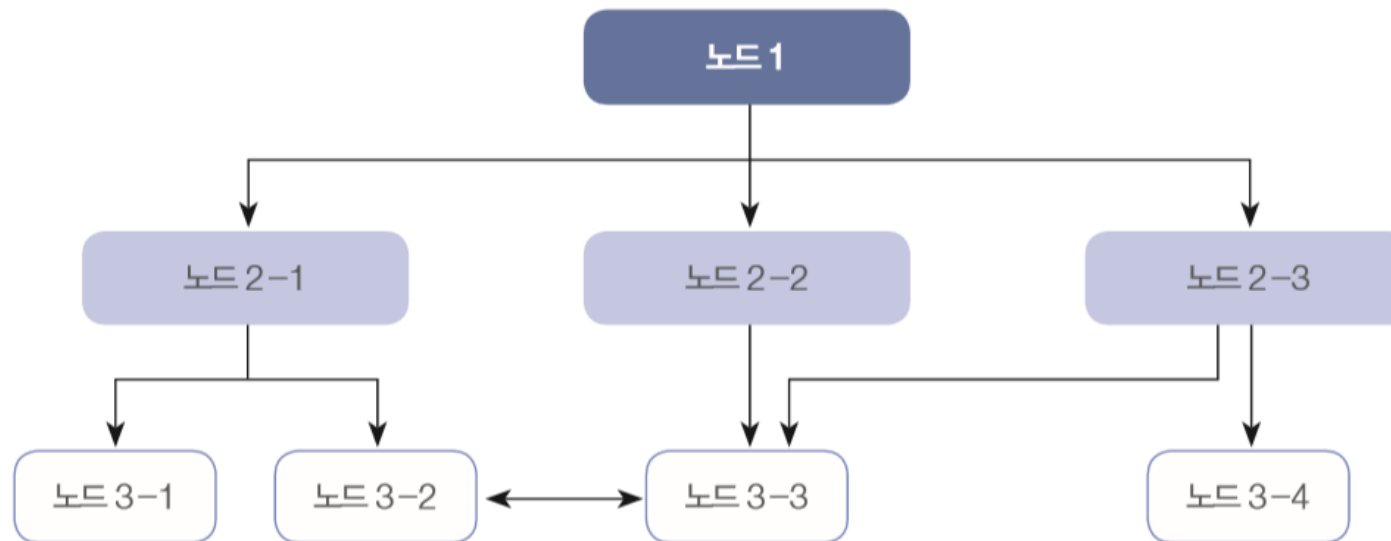
[그림 1-4] 계층형 구조

SECTION 01 DBMS 개요

DBMS 분류

◦ 망형 DBMS

- 계층형 DBMS의 문제점을 개선하기 위해 1970년대에 시작
- 1:1, 1:N, N:M(다대다) 관계 지원 - 효과적이고 빠른 데이터 추출
- 복잡한 내부 포인터 사용
 - 프로그래머가 이 모든 구조를 이해해야만 프로그램의 작성 가능

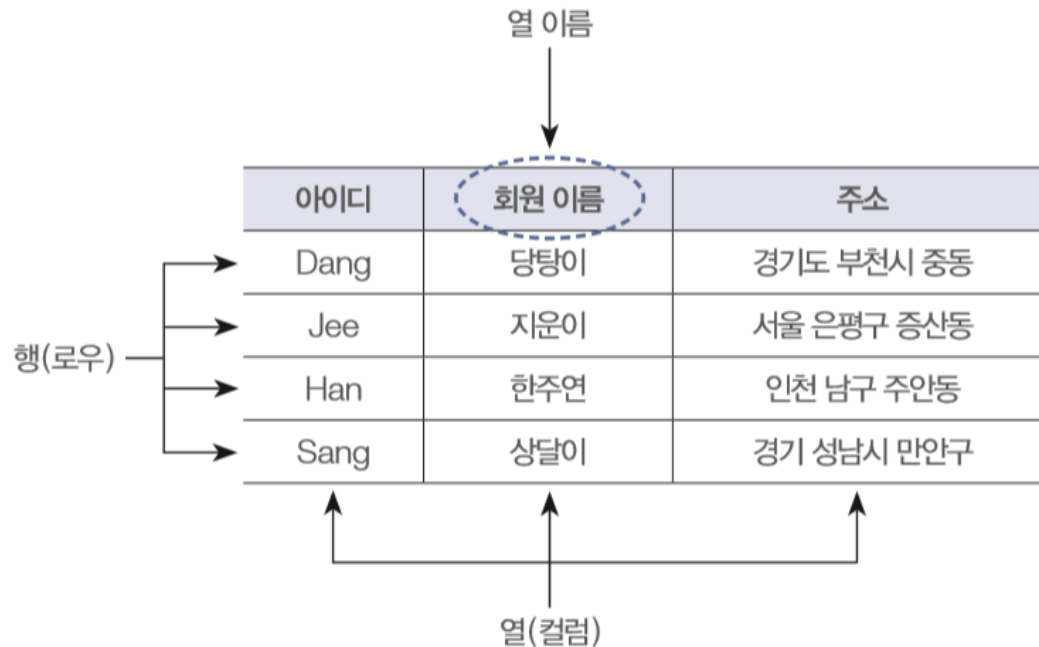


[그림 1-5] 망형 구조

SECTION 01 DBMS 개요

DBMS 분류

- 관계형 DBMS (Relational DBMS)
 - 1969년 E.F.Codd라는 학자가 수학 모델에 근거해 고안
 - 데이터베이스는 테이블Table이라 불리는 최소 단위로 구성
 - 이 테이블은 하나 이상의 열로 구성



[그림 1-6] 관계형 구조

SECTION 01 DBMS 개요

관계형 DBMS (Relational DBMS)의 장단점

◦ 장점

- 다른 DBMS에 비해 업무가 변화될 경우 쉽게 변화에 순응
- 유지보수 측면에서도 편리
- 대용량 데이터의 관리와 데이터 무결성Integration보장

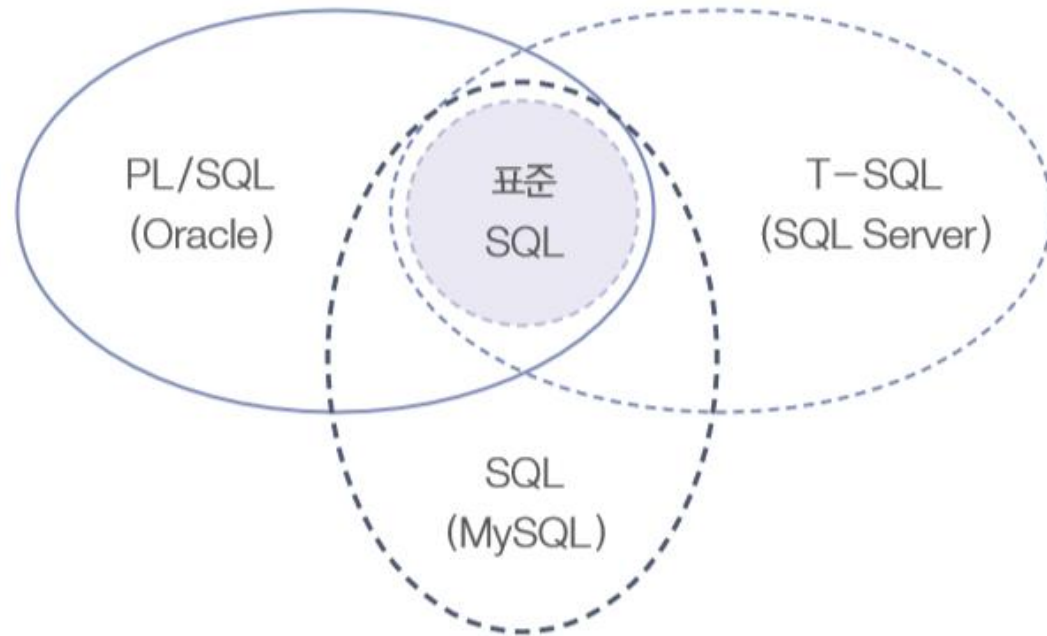
◦ 단점

- 시스템 자원을 많이 차지해 시스템이 전반적으로 느려지는 것
- 하드웨어 발전되어 해결

SECTION 01 DBMS 개요

SQL 개요

- SQL (Structured Query Language)
 - 관계형 데이터베이스에서 사용되는 언어, '에스큐엘' 또는 '시퀄'
 - DBMS 제작 회사와 독립적
 - 다른 시스템으로 이식성이 좋음
 - 표준이 계속 발전중
 - 대화식 언어
 - 분산형 클라이언트/서버 구조



[그림 1-7] 표준 SQL과 각 회사의 SQL

SECTION 02 MySQL 소개

MySQL의 개요와 변천사

- Oracle사에서 제작한 DBMS 소프트웨어
 - 대량의 데이터를 관리해주는 소프트웨어
- 오픈 소스 (Open Source) 로 제공
- 우리는 8.0 사용하여 학습

SECTION 03 MySQL의 에디션 및 기능 비교

상용 에디션

- Standard, Enterprise, Cluster CGE
- 비용이나 기능 면 비교
 - Standard < Enterprise < Cluster CGE

무료 에디션

- Community
- Enterprise 버전과 기능상 차이는 거의 없음
- 사용 허가에 대한 라이선스 차이

Contents

● CHAPTER 02 MySQL 설치

- SECTION 01 MySQL 설치 전 준비사항
 - 1.1 소프트웨어 요구사항
- SECTION 02 MySQL 설치
- SECTION 03 샘플 데이터베이스 설치
- SECTION 04 설치 후에 확인할 사항
- SECTION 05 MySQL 제거
- SECTION 06 Linux에 MySQL 설치하기

CHAPTER 02 MySQL 설치

MySQL 설치시 준비사항과 설치 후에 확인할 사항에 대하여 알아본다.

SECTION 01 MySQL 설치 전 준비사항

소프트웨어 요구사항

- MySQL Community 8.0 설치 위한 하드웨어
 - Windows가 설치된 머신
- MySQL Community 8.0 설치 위한 소프트웨어
 - Windows 10(64bit) 이상, Windows Server 2012 R2 이상의 버전에서만 설치 가능
 - 부가적인 기능을 사용하기 위해서 추가 소프트웨어 설치 (p.21~22)

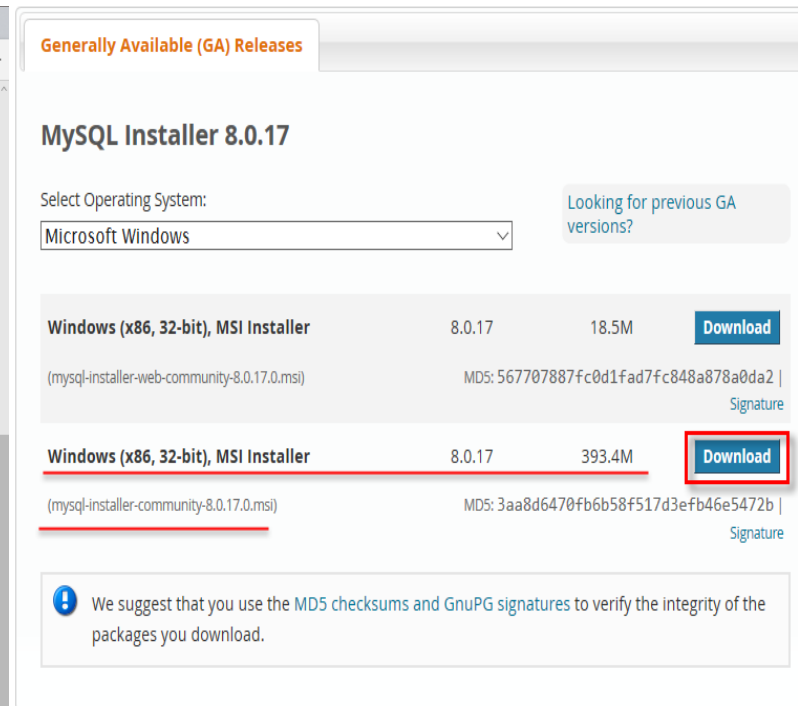
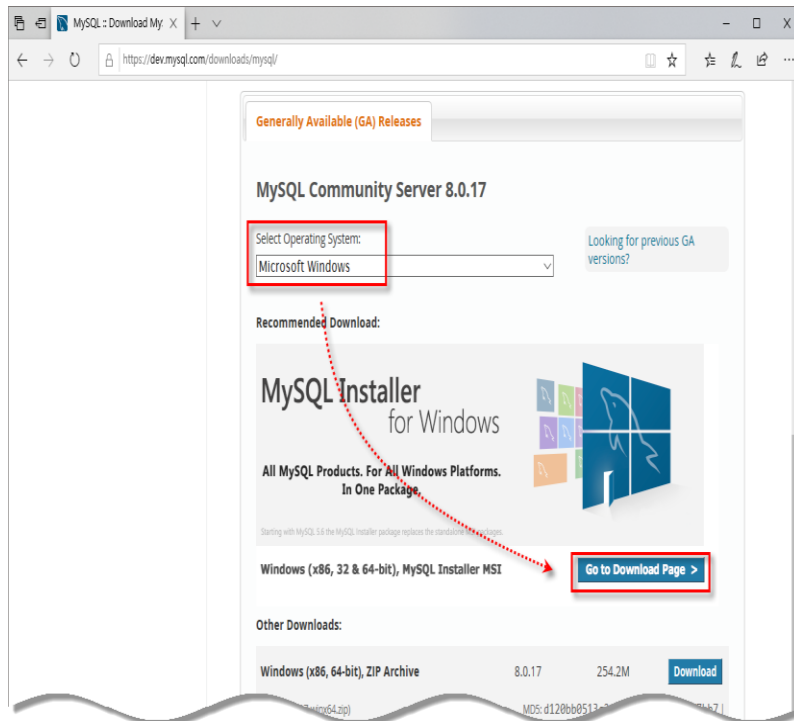
서버 운영 체제(x64)	PC 운영 체제(x64)
Windows Server 2019	Windows 10
Windows Server 2016	
Windows Server 2012 R2	

[표 2-1] MySQL Community 8.0을 설치하기 위한 운영 체제

SECTION 02 MySQL 설치

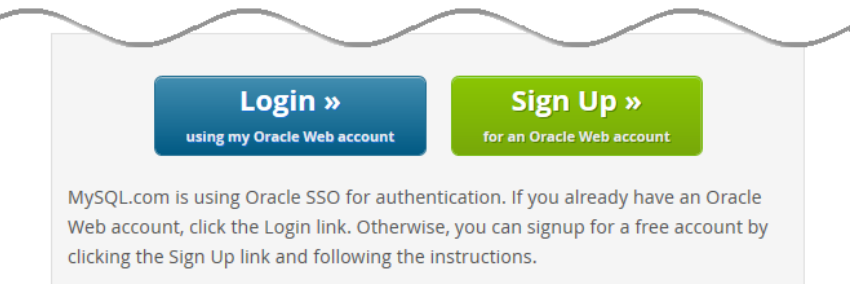
MySQL Community 8.0.17설치

- MySQL Community 8.0 다운로드
 - <http://dev.mysql.com/downloads/mysql/> 에 접속
 - 플랫폼 'Microsoft Windows'로 선택 (64 bit 용 따로 없음)
 - [Windows(x86, 32&64-bit), MySQL Installer MSI] 다운로드



Begin Your Download

mysql-installer-community-8.0.17.0.msi

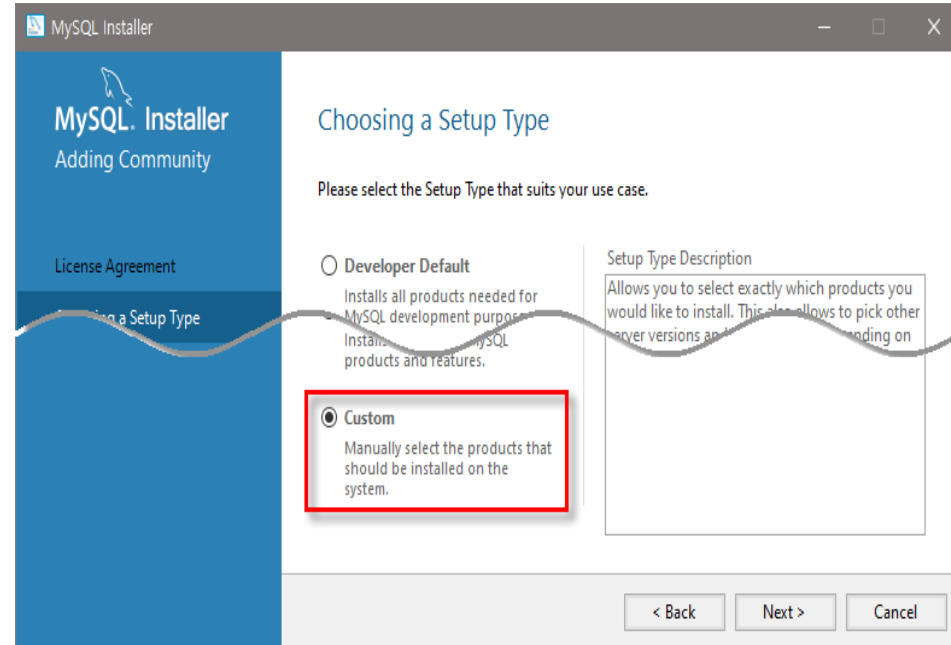
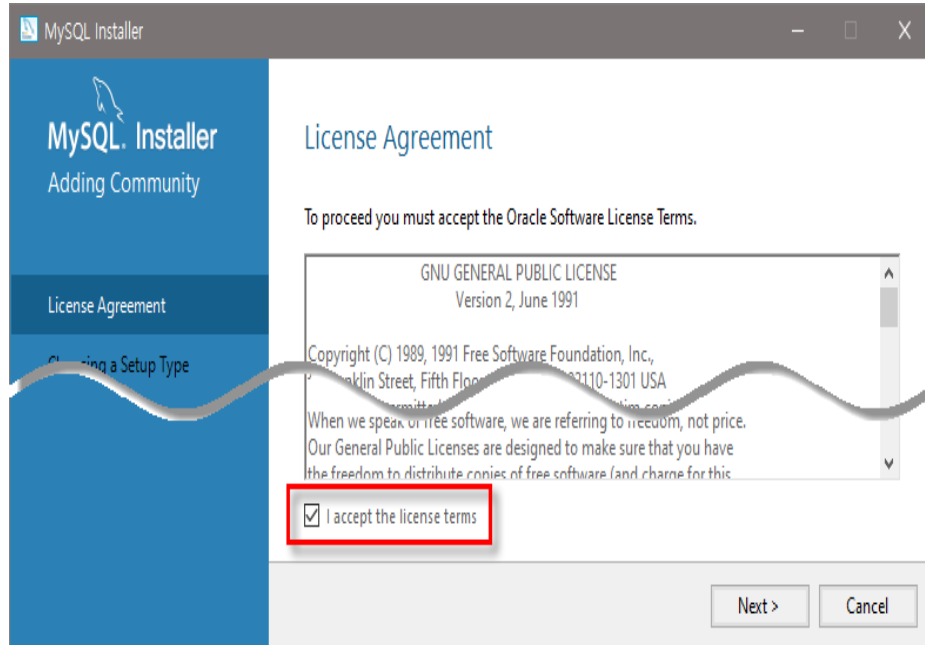


No thanks, just start my download.

SECTION 02 MySQL 설치

MySQL Community 8.0.17설치

- 다운로드 받은 MySQL 설치
 - 'mysql-installer-community-8.0.17.0' 더블 클릭해 설치
 - 'Installer'의 버전은 중요 X
 - [License Agreement] → <I accept the license terms> → <Next>
 - [Choosing a Setup Type] → <Custom> → <Next>



SECTION 02 MySQL 설치

MySQL Community 8.0.17설치

- 설치 이후 MySQL Server 8.0.17 환경 설정
 - [Product Configuration]에서 2개 항목의 환경 설정 필요 → <Next>
 - [High Availability] → 'Standard MySQL Server~~' 선택 → <Next>
 - [Type and Networking] → [Config Type] 'Development Computer' 선택
 - [TCP/IP]가 체크된 상태 - 포트 번호 3306
 - [Open Firewall port for network access]도 체크 → <Next>
 - [Authentication Method] → 'Use Strong Password~~' 선택 → <Next>
 - 관리자(root) 비밀번호 등록 → <Next>
 - 윈도우 서비스 등록 - 이름은 MySQL
 - [Apply Server Configuration] → <Execute> 선택하여 설정 완료

● CHAPTER 03 MySQL 전체 운영 실습

- SECTION 01 요구사항 분석과 시스템 설계 그리고 모델링

- 1.1 정보시스템 구축 절차 요약

- 1.2 데이터베이스 모델링과 필수 용어

- SECTION 02 MySQL을 이용한 데이터베이스 구축 절차

- 2.1 데이터베이스 생성

- 2.2 테이블 생성

- 2.3 데이터 입력

- 2.4 데이터 활용

● CHAPTER 03 MySQL 전체 운영 실습

- SECTION 03 테이블 외에 데이터베이스 개체의 활용

- 3.1 인덱스

- 3.2 뷰

- 3.3 스토어드 프로시저

- 3.4 트리거

- SECTION 04 데이터베이스 백업 및 관리

- 4.1 백업과 복원

- SECTION 05 MySQL과 응용 프로그램의 연결

CHAPTER 03 MySQL 전체 운영 실습

실무에서 발생하는 상황과 비슷한 설정을 하여 응용프로그램과 연동해본다.

SECTION 01 요구사항 분석과 시스템 설계 그리고 모델링

정보시스템 구축 절차 요약

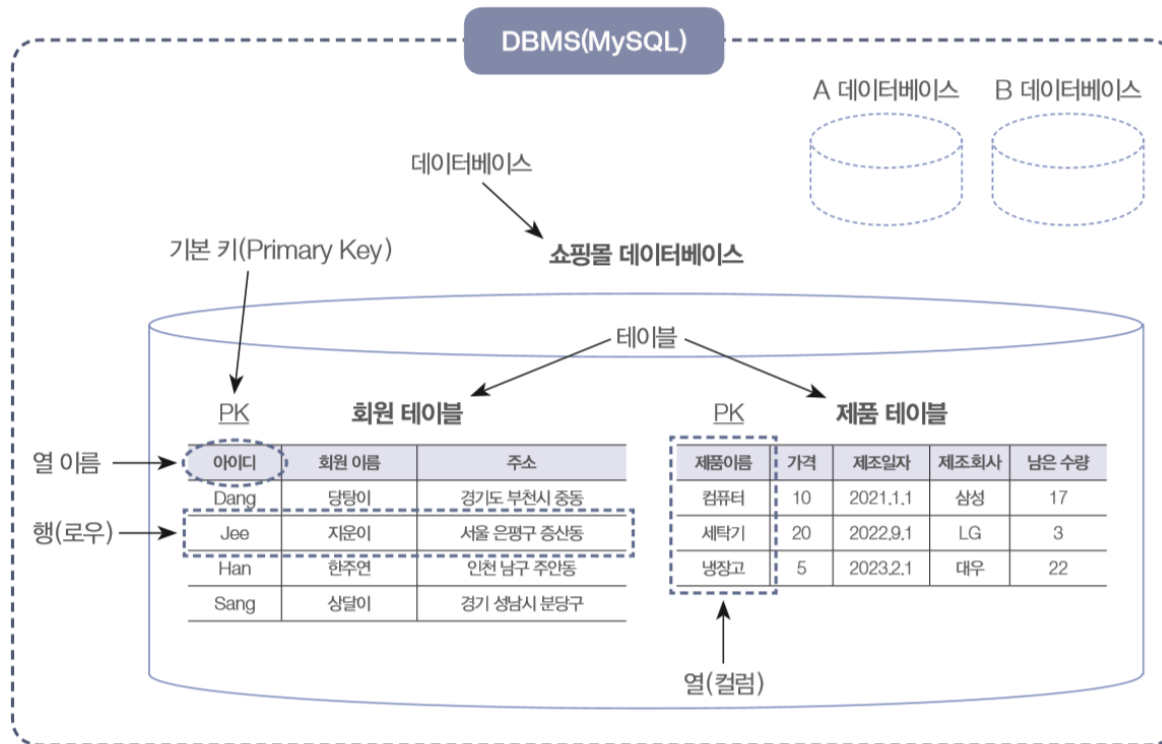
- 분석, 설계, 구현, 시험, 유지보수의 5가지 단계
- 분석
 - 구현하고자 하는 프로젝트의 가장 첫 번째 단계
 - 시스템 분석 또는 요구사항 분석이라고 불림
 - 요구사항 분석은 현재 우리가 '무엇을(What)' 할 것인지 결정
 - 사용자의 인터뷰와 업무 조사 등을 수행
 - 프로젝트의 첫 단추를 끼우는 중요한 단계
 - 분석의 결과로 많은 문서 작성
- 설계
 - 시스템 설계 또는 프로그램 설계
 - 구축하고자 하는 시스템을 '어떻게(How)' 할 것인지 결정
 - 대부분의 프로젝트에서 분석과 설계의 과정이 전체 공정의 50% 이상 차지

SECTION 01 요구사항 분석과 시스템 설계 그리고 모델링

데이터베이스 모델링과 필수 용어

데이터베이스 모델링

- 현실세계에서 사용되는 데이터를 MySQL에 어떻게 옮겨 놓을 것인지를 결정하는 과정
- 저장할 정보는 테이블(Table)이라는 형식에 맞춰 저장
- Ex) 쇼핑몰 데이터 베이스의 예



[그림 3-1] 테이블의 구조와 관련 용어

SECTION 01 요구사항 분석과 시스템 설계 그리고 모델링

데이터베이스 모델링과 필수 용어

- 데이터
 - 하나하나의 단편적인 정보
 - 정보는 있으나 아직 체계화 되지 못한 상태
- 테이블
 - 데이터를 입력하기 위해, 표 형태로 표현한 것
 - Ex) 회원 정보 테이블, 제품 정보 테이블
- 데이터베이스(DB)
 - 테이블이 저장되는 저장소
 - 각 데이터베이스는 서로 다른 고유한 이름을 가지고 있음
- DBMS (DataBase Management System)
 - 데이터베이스를 관리하는 시스템 또는 소프트웨어

SECTION 01 요구사항 분석과 시스템 설계 그리고 모델링

데이터베이스 모델링과 필수 용어

- 열(=컬럼=필드)
 - 각 테이블은 열로 구성
 - 회원 테이블의 경우에는 아이디, 회원 이름, 주소 등 3개의 열로 구성
- 열 이름
 - 각 열을 구분하기 위한 이름
 - 열 이름은 각 테이블 내에서는 중복되지 않고, 고유해야 함
- 데이터 형식
 - 열의 데이터 형식
 - 테이블을 생성할 때 열 이름과 함께 지정
- 행(=로우=레코드)
 - 실질적인 데이터
 - 회원 테이블의 경우 4건의 행 데이터, 즉 4명의 회원이 존재함

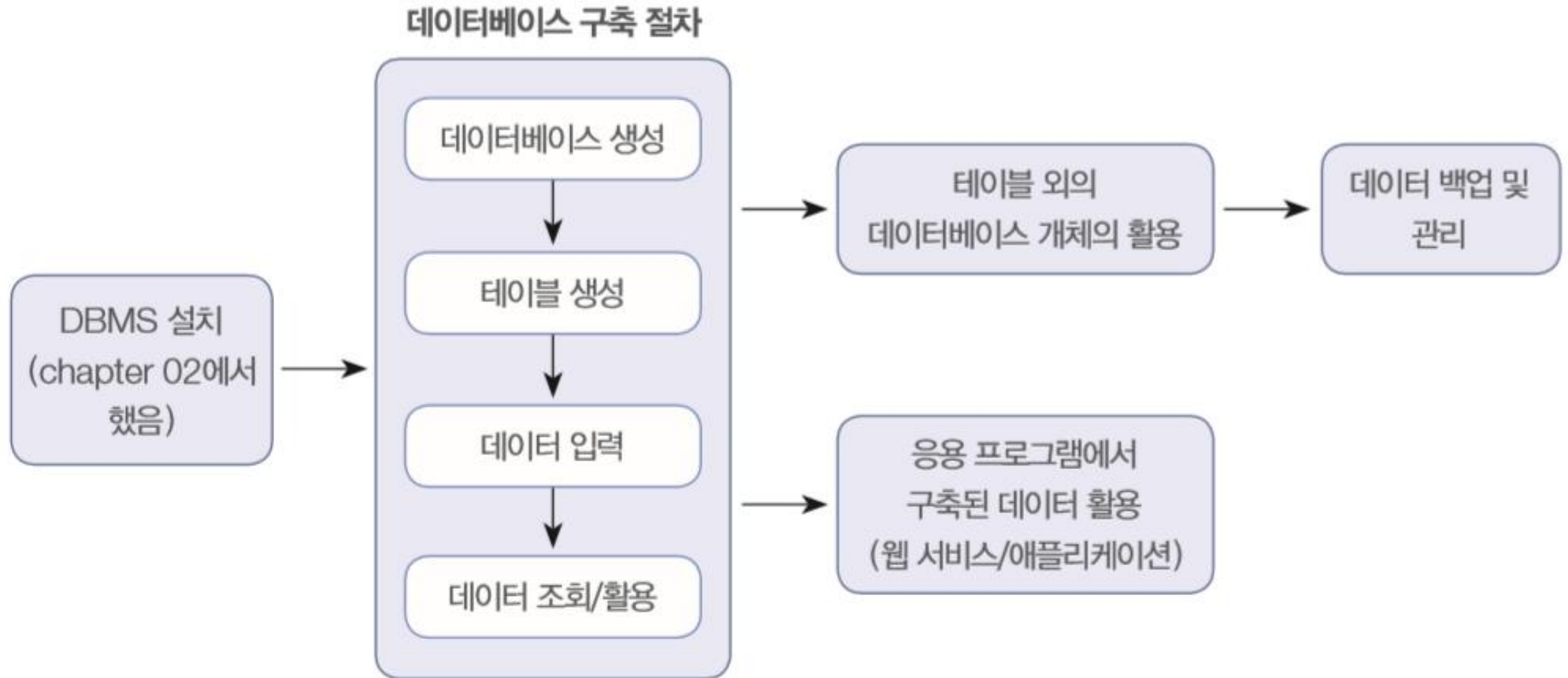
SECTION 01 요구사항 분석과 시스템 설계 그리고 모델링

데이터베이스 모델링과 필수 용어

- 기본 키 (Primary Key) 열
 - 기본 키(또는 주 키) 열은 각 행을 구분하는 유일한 열
 - 중복되어서는 안되며, 비어 있어서도 안 됨
 - 각 테이블에는 기본 키가 하나만 지정
- 외래 키(Foreign Key) 필드
 - 두 테이블의 관계를 맺어주는 키
 - 4장 이후 설명
- SQL (Structured Query Language)
 - 구조화된 질의 언어
 - 사람과 DBMS가 소통하기 위한 말(언어)
 - 6, 7장에서 자세히 다룸

SECTION 02 MySQL을 이용한 데이터베이스 구축 절차

데이터베이스 구축/관리 및 활용의 전반적인 절차



[그림 3-2] 데이터베이스 구축/관리 및 활용의 전반적인 절차

SECTION 02 MySQL을 이용한 데이터베이스 구축 절차

인터넷 쇼핑몰 구축 위한 '쇼핑몰' DB 생성

- MySQL 서버 연결 및 설정
 - Windows의 [시작] >> [M] >> [MySQL] >> [MySQL Workbench 8.0 CE] 클릭해 Workbench 실행
 - [MySQL Connections] 창에서 비밀번호 입력하여 접속
 - Workbench의 초기 창
 - 기본적으로는 [Schemas] 탭 클릭해놓고 사용
 - Workbench 종료 → 설정 저장

SECTION 02 MySQL을 이용한 데이터베이스 구축 절차

인터넷 쇼핑몰 구축 위한 '쇼핑몰' DB 생성

◦ 스키마 (Schema) 생성

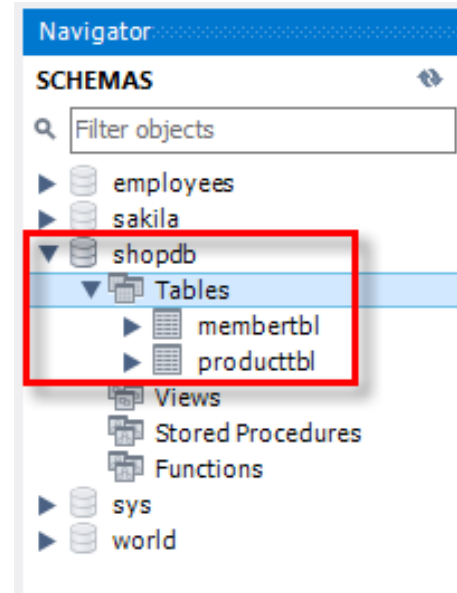
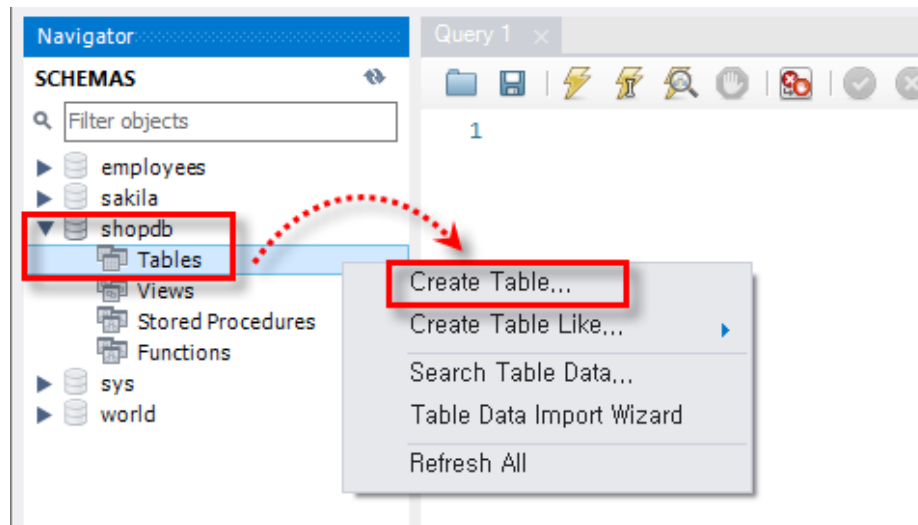
- MySQL에서는 스키마와 데이터베이스가 완전히 동일한 용어로 사용
- Workbench의 [SCHEMAS]의 빈 부분
 - 마우스 오른쪽 버튼 클릭 후 [Create Schema](=Create Database) 선택
 - CREATE SCHEMA 'shopdb'문을 쿼리 창에서 입력하는 것과 동일한 작동
 - 이름 입력하면 DB 생성
- 왼쪽 데이터베이스 목록에 shopdb 데이터베이스 확인
- 아무것도 들어있지 않은 데이터베이스 생성

SECTION 02 MySQL을 이용한 데이터베이스 구축 절차

인터넷 쇼핑몰 구축 위한 '쇼핑몰' DB 생성

테이블 생성

- 회원테이블, 제품 테이블 각 열의 영문 이름 및 데이터 형식 결정
 - 데이터베이스 모델링(특히 물리적 모델링) 시에 결정
 - 데이터 형식의 자세한 내용은 7장에서 학습
- 형식이 정해지면 Create Table 실행해 테이블이름, 열이름, 데이터형식등 테이블 내용 입력
- 생성된 SQL Query를 데이터베이스에 적용해 테이블 생성 완료
- ShopDB의 [테이블]에서 생성한 테이블 확인

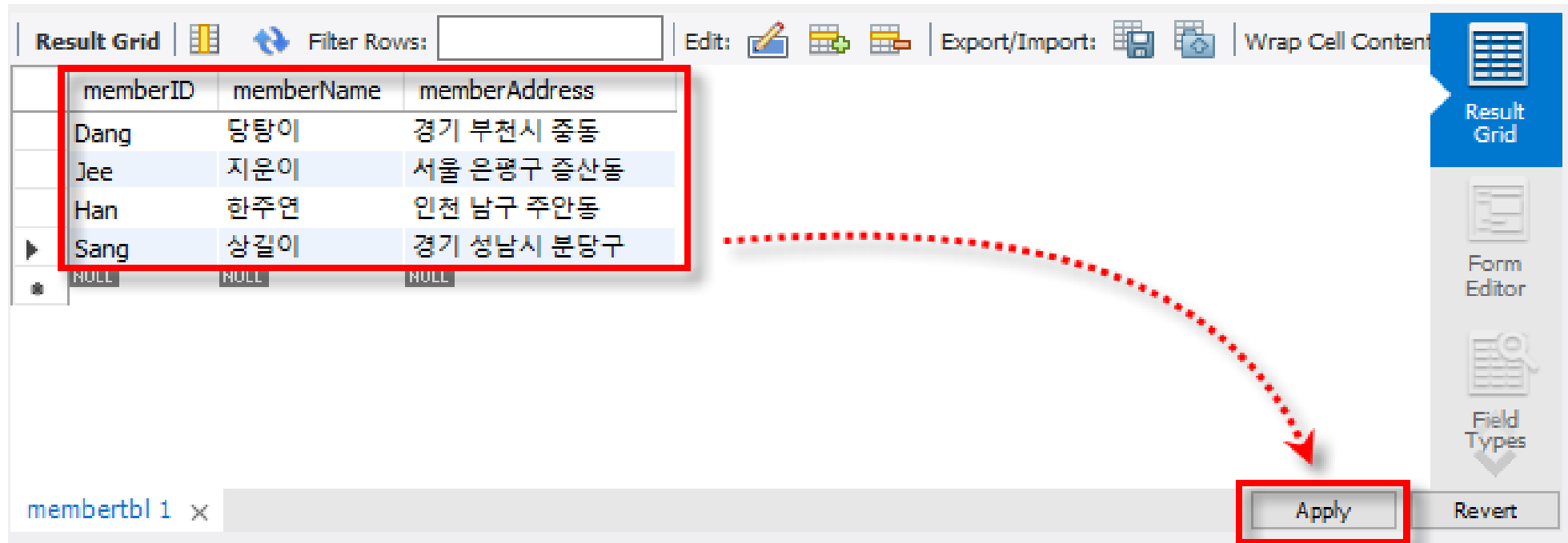


SECTION 02 MySQL을 이용한 데이터베이스 구축 절차

데이터 입력 - 행 데이터 입력

- 회원 테이블의 데이터 입력

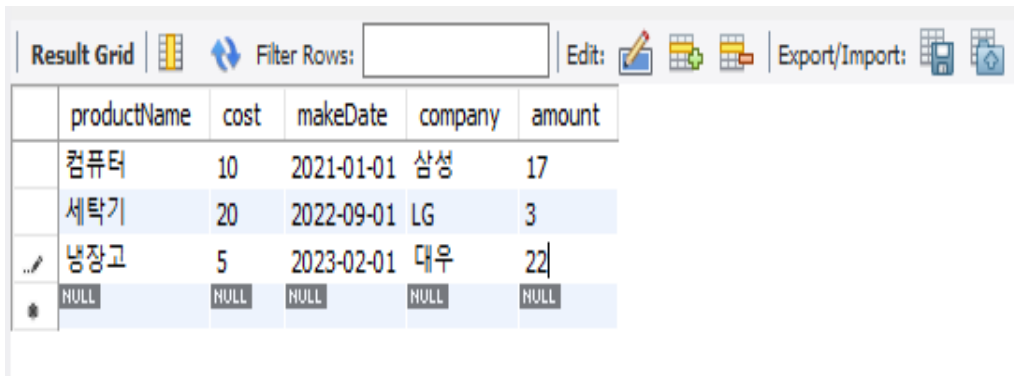
- Navigator의 [SCHEMAS]에서 [Shopdb] >> [Tables] >> [membertbl] 선택 후,
마우스 오른쪽 버튼 클릭하고 [Select Rows - Limits 1000] 선택
- 아래 그림의 회원 테이블 데이터 Grid에 입력 후 Apply 해 저장



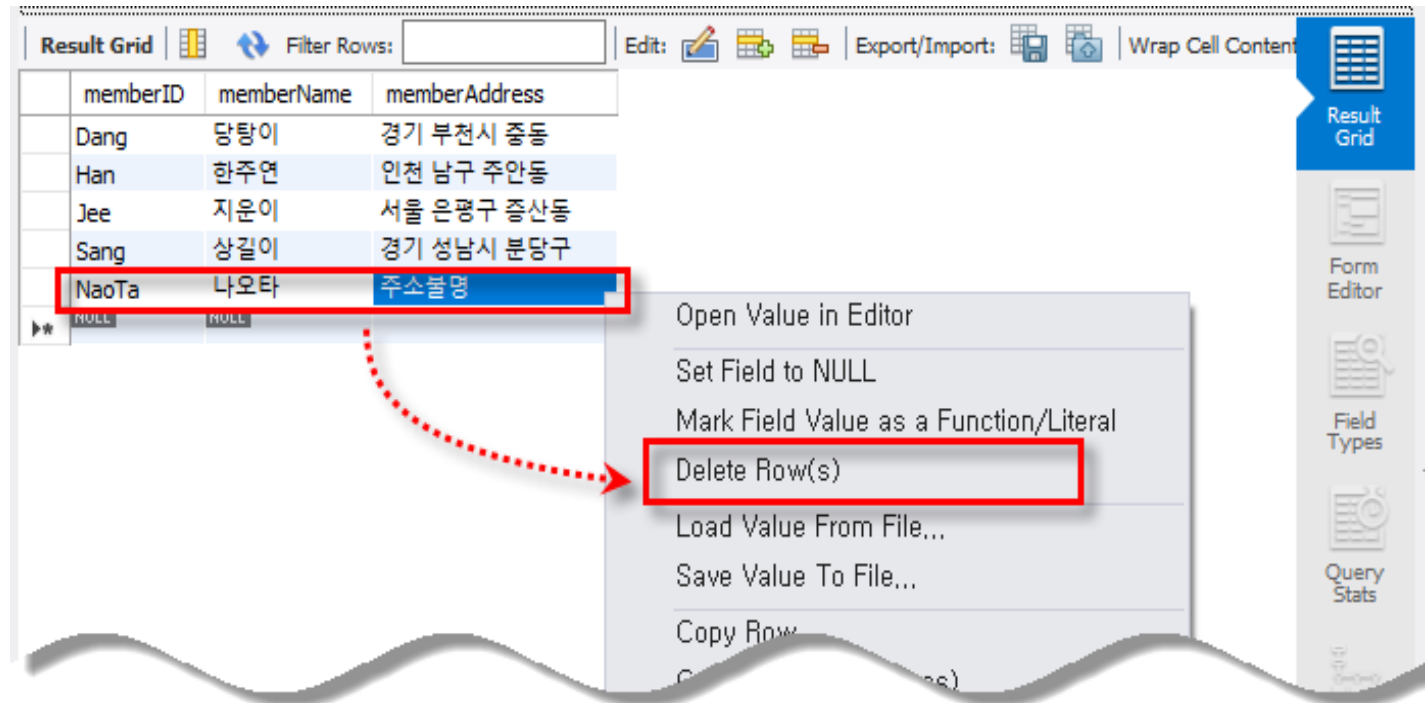
SECTION 02 MySQL을 이용한 데이터베이스 구축 절차

데이터 입력 - 행 데이터 입력

- 제품 테이블의 데이터 입력
 - 동일한 방식으로 제품 데이터 입력 후 저장
 - 데이터를 삭제하려면 삭제할 행의 앞 부분에 마우스 대고 오른쪽 메뉴 Delete 사용해 삭제



productName	cost	makeDate	company	amount
컴퓨터	10	2021-01-01	삼성	17
세탁기	20	2022-09-01	LG	3
냉장고	5	2023-02-01	대우	22
NULL	NULL	NULL	NULL	NULL



memberID	memberName	memberAddress
Dang	당탕이	경기 부천시 중동
Han	한주연	인천 남구 주안동
Jee	지운이	서울 은평구 증산동
Sang	상길이	경기 성남시 분당구
NaoTa	나오타	주소불명
NULL	NULL	NULL

- Open Value in Editor
- Set Field to NULL
- Mark Field Value as a Function/Literal
- Delete Row(s)
- Load Value From File...
- Save Value To File...
- Copy Row
- Copy Row(s)

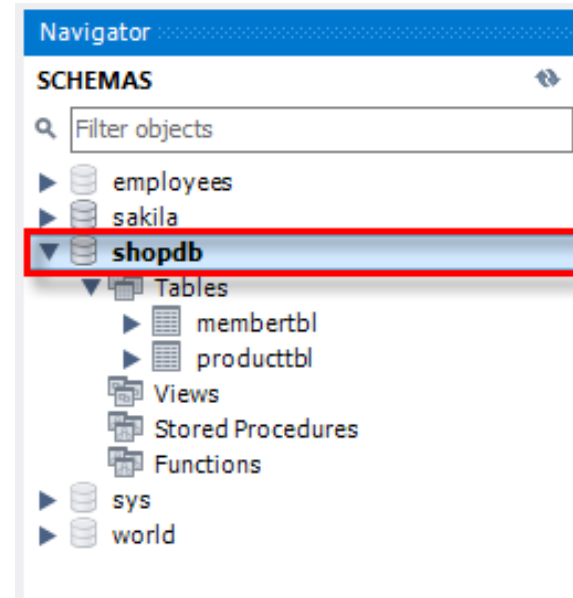
SECTION 02 MySQL을 이용한 데이터베이스 구축 절차

데이터 활용

- 주로 SELECT 문 사용해 데이터 활용

사용할 데이터 베이스 선택

- SCHEMA에서 사용할 DB를 더블 클릭
- 진하게 색상이 변하면서 선택 됨



SQL 실행법

- 툴바의 <Execute the selected portion~~> 아이콘 클릭
 - **Ctrl** + **Shift** + **Enter**
 - Workbench 메뉴의 [Query] >> [Execute(All or Selection)]
- SQL은 대소문자 구별 없음
 - 읽기 편하게 예약어는 대문자 (쿼리 창에서 파란색으로 표시)

SECTION 02 MySQL을 이용한 데이터베이스 구축 절차

데이터 활용

- SELECT 열 이름 FROM 테이블 이름 [WHERE 조건]
 - 모든 데이터 출력하기 (열 이름 대신 ' * ')
 - 열을 선택해 데이터 출력하기 (열 이름 나열)
 - 특정 데이터를 만족하는 데이터 출력하기 (WHERE절에 조건 입력)
- 새로운 테이블 생성
 - 테이블 이름에 space 가 들어간 경우의 처리(백틱[backtick]키 활용)
 - Navigator 창에서 "Refresh All" 의 중요성
 - 새로 테이블을 만든 뒤 개체가 보이지 않을 경우 필수로 실행할 것
- 테이블 삭제
 - DROP TABLE 테이블 이름

SECTION 03 테이블 외의 데이터베이스 개체의 활용

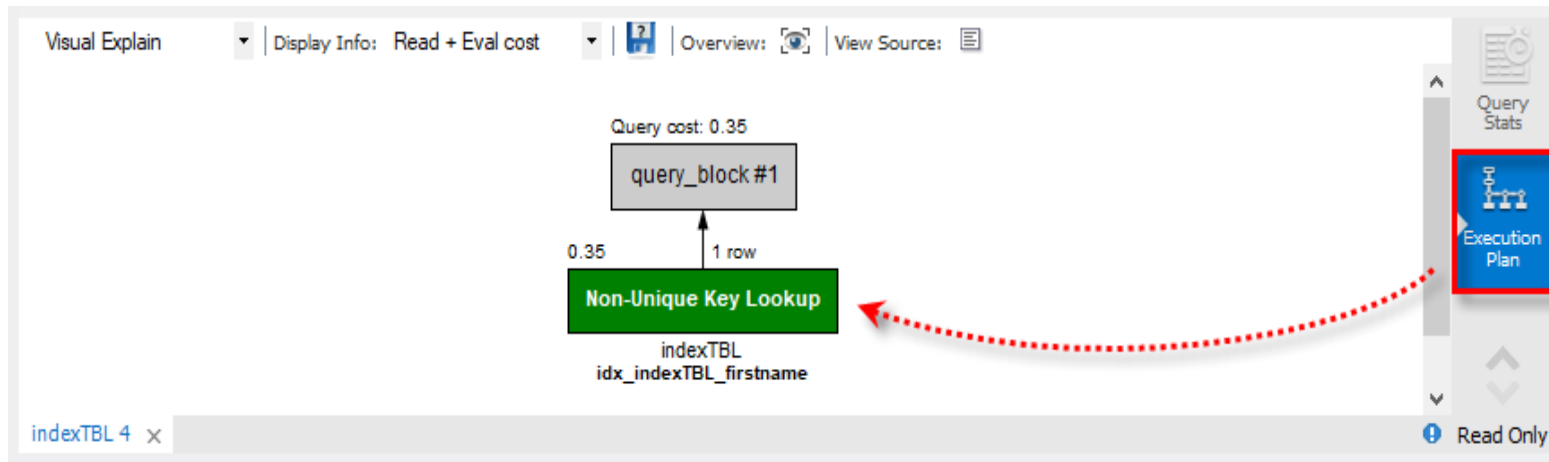
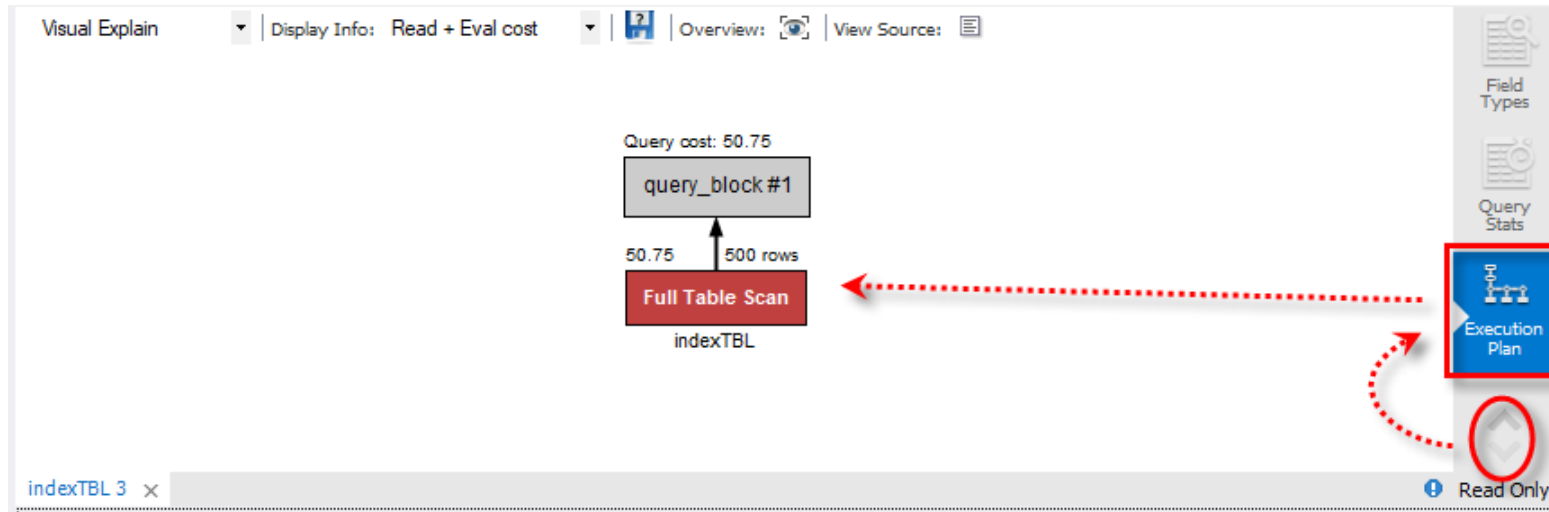
인덱스 (Index)

- 데이터베이스 '튜닝'의 개념
 - 데이터베이스 성능 향상
 - 쿼리에 응답하는 시간 단축시키는 것
- 책 뒤에 붙어 있는 '찾아보기'(또는 색인)와 같은 개념
- 데이터의 양이 많을수록 효과적으로 작용
 - 응답속도가 현저히 차이 나는 결과
- 테이블의 열 단위에 생성

SECTION 03 테이블 외의 데이터베이스 개체의 활용

인덱스 (Index)

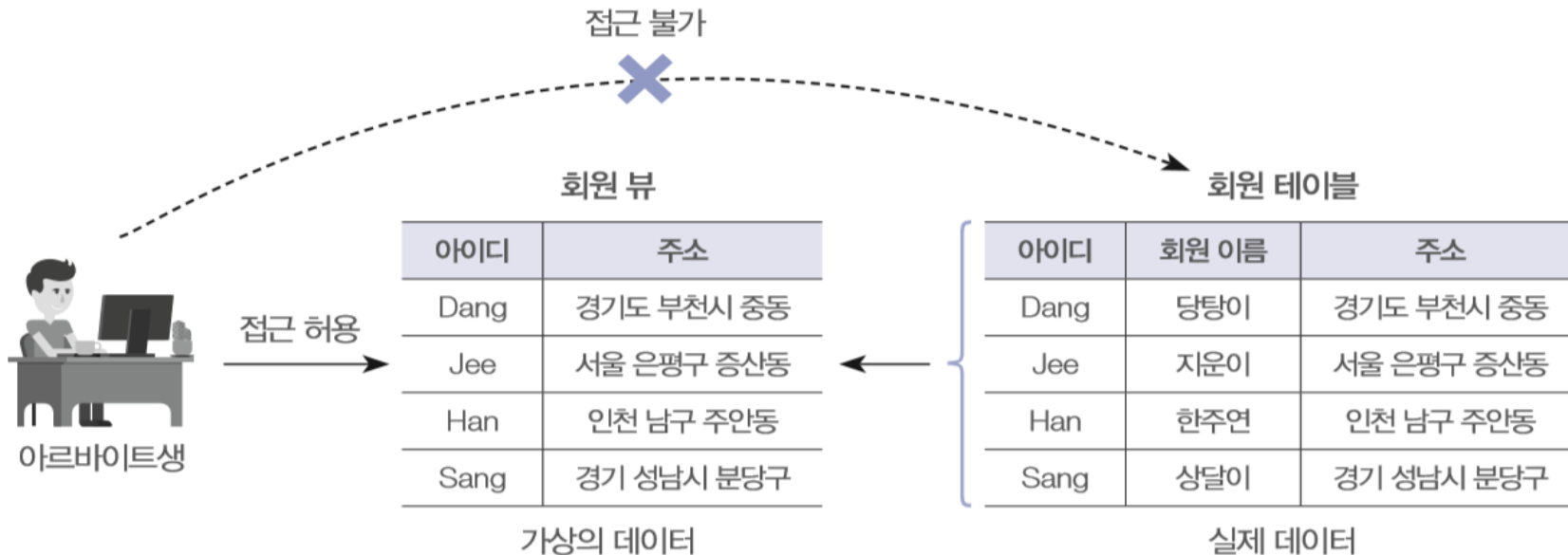
- 인덱스 사용 전/후의 실행 계획 (Execution Plan) 비교



SECTION 03 테이블 외의 데이터베이스 개체의 활용

뷰 (View)

- 가상의 테이블
- 실제 행 데이터를 가지고 있지 않음
 - 그 실체는 없는 것이며, 진짜 테이블에 링크Link된 개념
 - 뷰를 SELECT
 - 진짜 테이블의 데이터를 조회하는 것과 동일한 결과



[그림 3-33] 뷰의 사용 예

SECTION 03 테이블 외의 데이터베이스 개체의 활용

스토어드 프로시저 (Stored Procedure)

- MySQL에서 제공해주는 프로그래밍 기능
- SQL문을 하나로 묶어 편리하게 사용하는 기능
- 다른 프로그래밍 언어와 같은 기능을 담당할 수도 있음
 - 실무에서는 SQL문(주로 SELECT)을 매번 하나하나 수행 X
 - 스토어드 프로시저로 만들어 놓은 후 스토어드 프로시저 호출

SECTION 03 테이블 외의 데이터베이스 개체의 활용

트리거 (Trigger)

- 테이블에 부착되어 테이블에 INSERT나 UPDATE 또는 DELETE 작업이 발생되면 실행되는 코드
- 상세한 내용은 10장에서 학습 예정
- ex) 탈퇴회원 관리
 - 회원 테이블에서 빼서 탈퇴한 회원 관리 테이블로 옮김
 - 회원 정보 + 탈퇴한 날짜 를 관리하는 새 테이블의 필요성

SECTION 04 데이터베이스 백업 및 관리

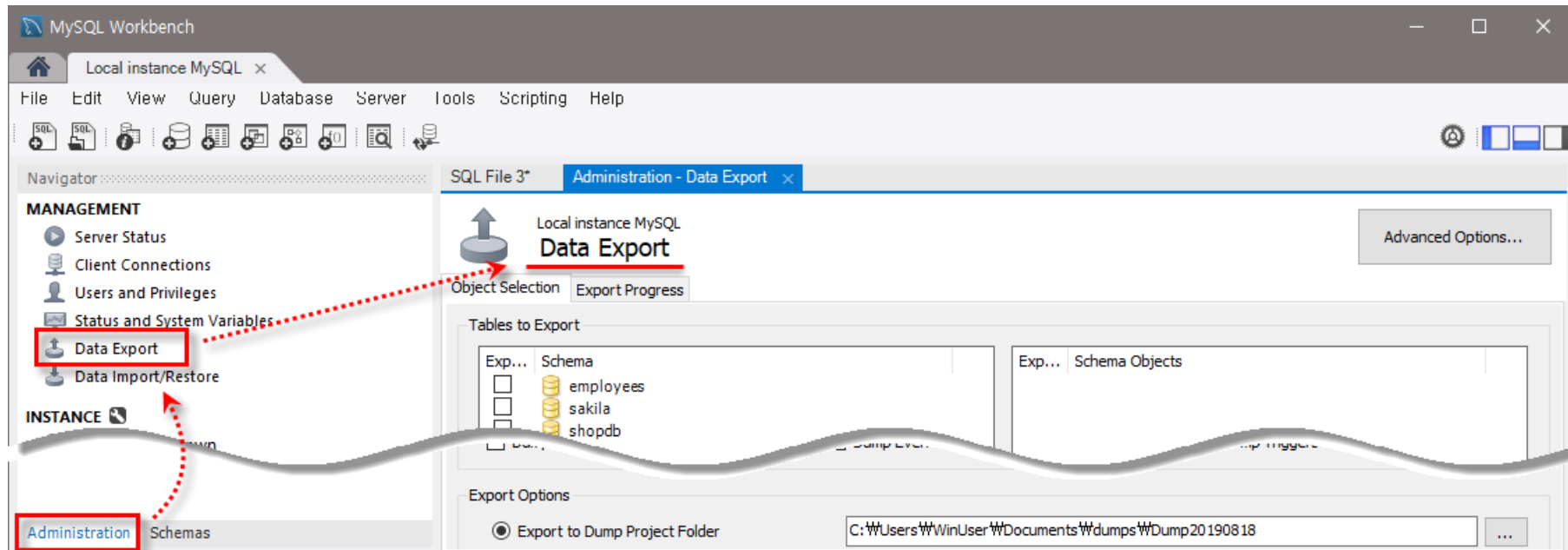
백업과 복원

- 백업
 - 현재의 데이터베이스를 다른 매체에 보관하는 작업
- 복원
 - 데이터베이스에 문제 발생 시 다른 매체에 백업된 데이터를 이용해 원상태로 돌려놓는 작업
- 백업과 복원은 DBA(DataBase Administrator: 데이터베이스 관리자)가 해야 할 가장 중요한 일

SECTION 04 데이터베이스 백업 및 관리

데이터베이스 백업

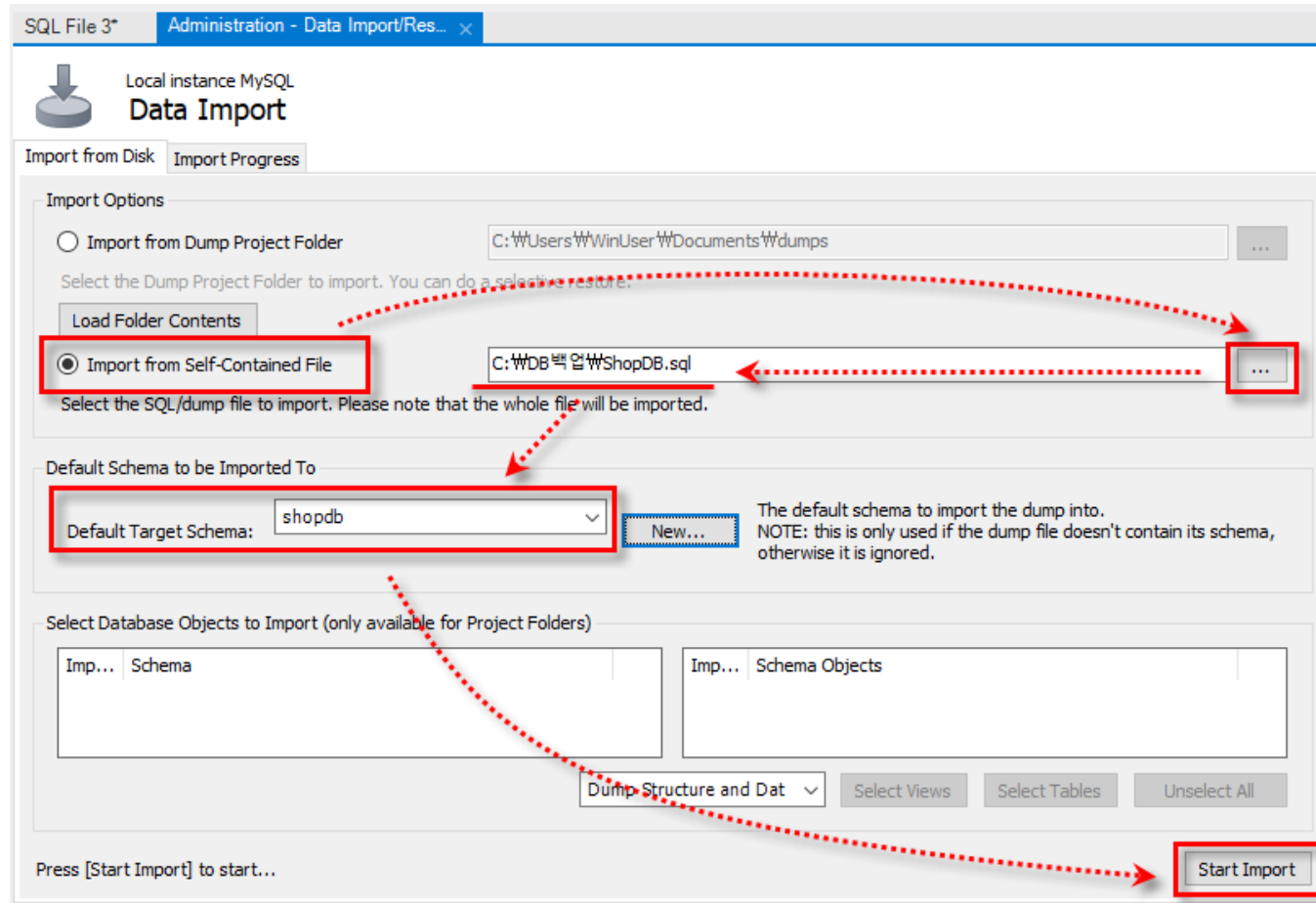
- 백업용 폴더 작성
 - 실제로는 다른 디스크에 이루어져야 의미 있음
- DB 백업
 - DB내의 모든 트리거, 스토어드 프로시저까지 백업
 - 백업 폴더에 백업파일 저장



SECTION 04 데이터베이스 백업 및 관리

데이터베이스 복구

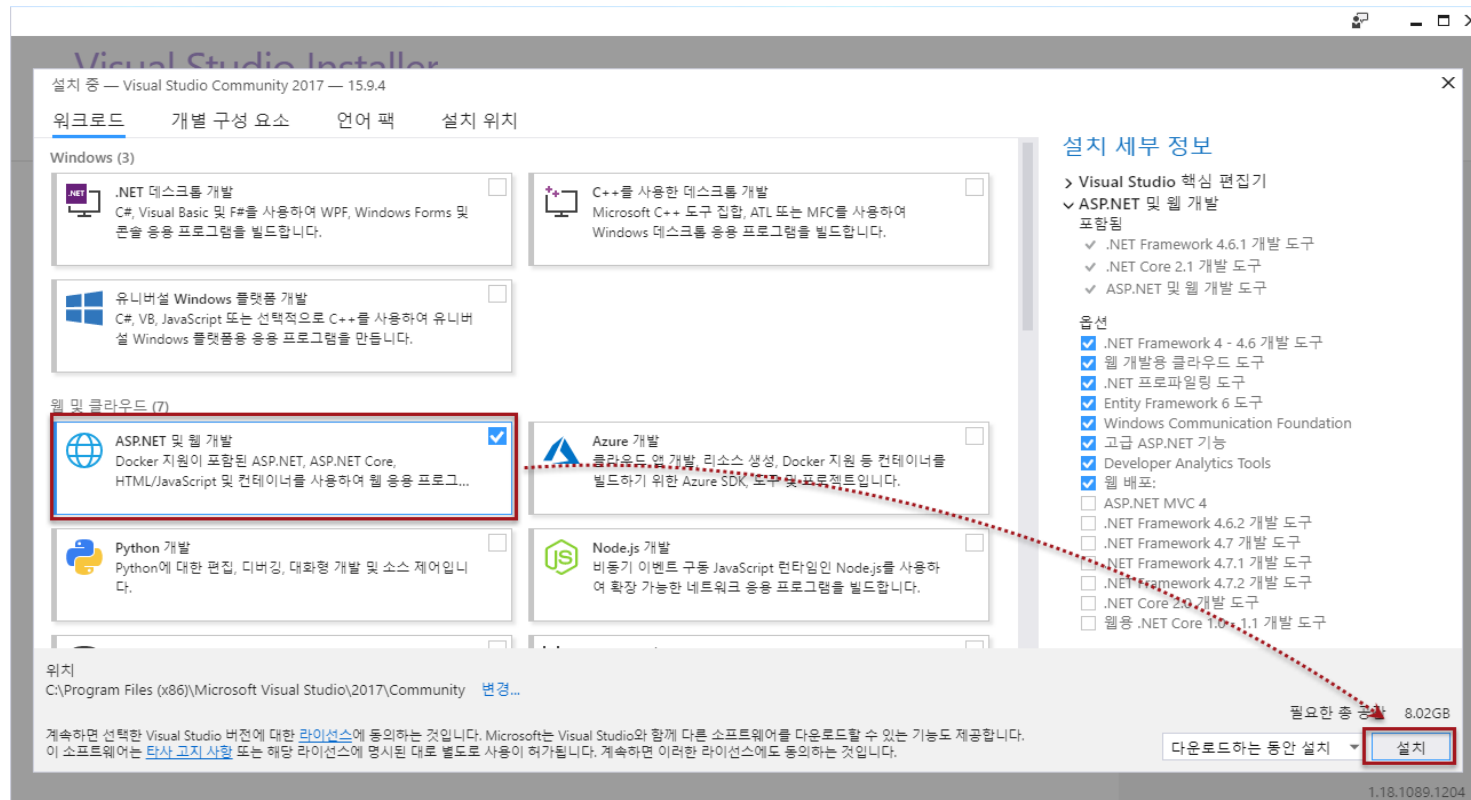
- DB 삭제 같은 큰 사고를 인위로 발생시켜 실습
- 복원 후 데이터가 온전한지 check 하는 것이 중요함



SECTION 05 MySQL과 응용프로그램의 연결

MySQL에서 구축한 쇼핑몰 데이터베이스를 웹에서 서비스

- 개발 툴로 사용할 Microsoft Visual Studio Community 2017 설치 파일 다운로드
 - <http://cafe.naver.com/thisisMySQL> 또는 <http://download.hanbit.co.kr/mysql/8.0/>
 - Visual Studio Community 2017(vs_community_2017.exe, 1.22MB) 다운로드 후 설치
 - <ASP.NET 및 웹 개발> 체크하고 <설치> 클릭



SECTION 05 MySQL과 응용프로그램의 연결

MySQL과 응용프로그램 연결

- Connector/ODBC 설치 2017 설치 파일 다운로드
 - <http://dev.mysql.com/downloads/connector/odbc/>
 - <http://cafe.naver.com/thisMySQL>
 - 반드시 32bit용 다운로드 (64bit용은 작동에 문제 있음)
 - mysql-connector-odbc-8.0.17-win32.msi 실행해 설치
- ODBC 연결 설정
 - Windows의 [시작] >> [W] >> [Windows 관리도구] >> [ODBC Data Sources(32-bit)] 실행
 - [ODBC 데이터 원본 관리자(32비트)] [시스템 DSN] 탭 클릭 <추가> 버튼 클릭
 - [MySQL ODBC 8.0 Unicode Driver] 선택하고 <마침> 클릭

SECTION 05 MySQL과 응용프로그램의 연결

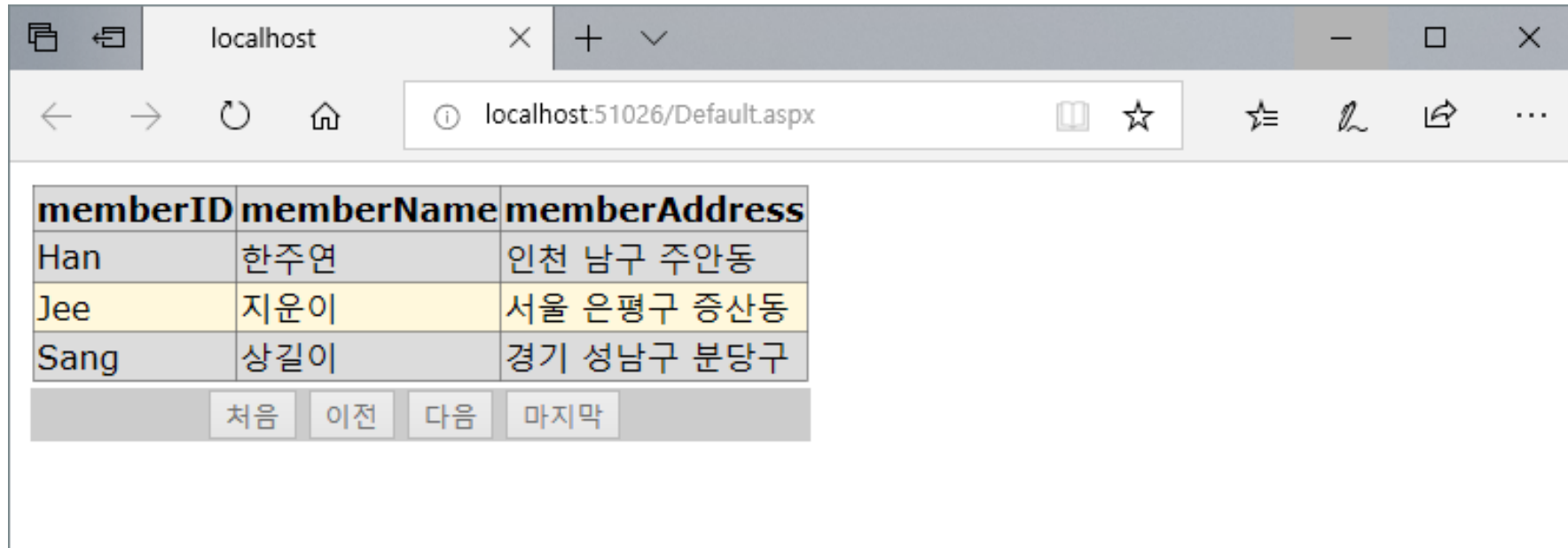
ASP.NET 웹 응용프로그램 작성

- Visual Studio 2017 실행
- GUI 모드에서 웹 사이트 작성
 - [파일] >> [새로 만들기] >> [프로젝트] 클릭, [Visual C#] >> [웹]>>[이전 버전]
<ASP.NET 빈 웹 사이트> 선택
 - 오른쪽 솔루션 탐색기에서 지구 모양 아이콘
 - 'WebSite1(1)'에서 마우스 오른쪽 버튼 클릭, [추가]>>[Web Form]
 - 이름은 'Default' 그대로 두고 <확인> 클릭
 - [디자인]을 클릭해 디자인 모드로 변경
 - [도구 상자] 클릭해 확장
 - [데이터]부분의 [SqlDataSource]를 더블클릭하거나 드래그해서 우측의 빈 디자인 창에 갖다놓기

SECTION 05 MySQL과 응용프로그램의 연결

ASP.NET 웹 응용프로그램 작성

- GUI 모드에서 웹 사이트 작성
 - 데이터 연결
 - 웹에서 보여줄 내용의 Query 입력
 - SQL문 작성
 - 적절한 레이아웃 설정
 - 웹사이트 서비스 테스트



A screenshot of a web browser window. The address bar shows 'localhost:51026/Default.aspx'. The main content area displays a table with three columns: 'memberID', 'memberName', and 'memberAddress'. The table contains three rows of data. The second row, with 'Jee' as the memberID, is highlighted in yellow. Below the table are four buttons: '처음' (First), '이전' (Previous), '다음' (Next), and '마지막' (Last).

memberID	memberName	memberAddress
Han	한주연	인천 남구 주안동
Jee	지운이	서울 은평구 증산동
Sang	상길이	경기 성남구 분당구

처음 이전 다음 마지막

● CHAPTER 04 데이터베이스 모델링

- SECTION 01 프로젝트의 진행단계

- SECTION 02 데이터베이스 모델링

- 2.1 데이터베이스 모델링 개념

- 2.2 데이터베이스 모델링 실습

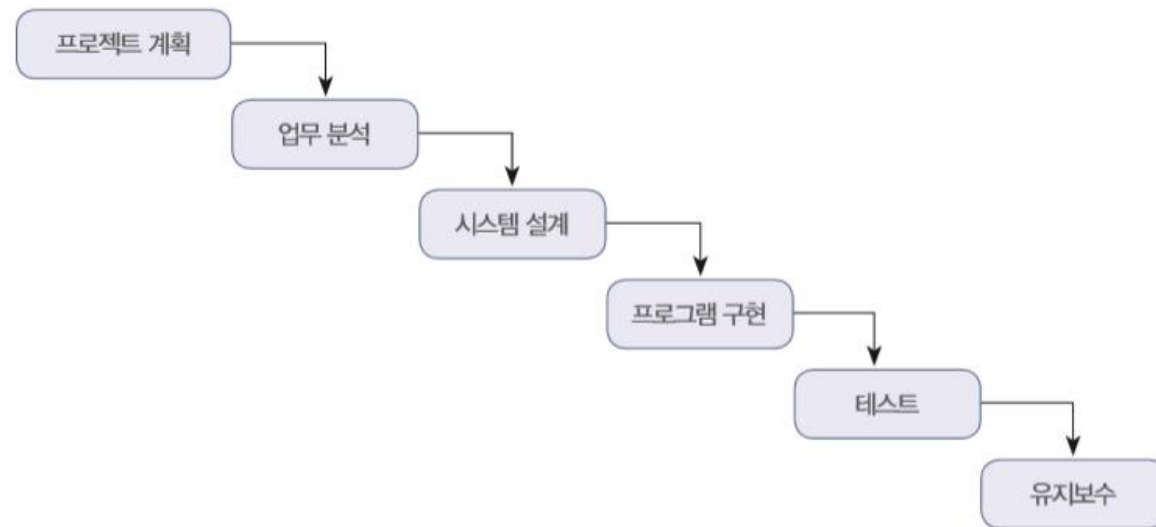
CHAPTER 04 데이터베이스 모델링

데이터베이스 모델링에 대한 개념을 파악하고 모델링 절차를 실습한다.

SECTION 01 프로젝트의 진행 단계

프로젝트 (Project)

- 현실세계의 업무를 컴퓨터 시스템으로 옮겨놓는 일련의 과정
- 대규모의 프로그램을 작성하기 위한 전체 과정
 - ex) 집 짓기의 경우 : 초가집 → 목조건물 → 수 십층 이상의 건물
- 분석과 설계 작업 등한시 → '소프트웨어 개발 방법론'의 대두
- 폭포수 모델 (Waterfall Model)



[그림 4-1] 폭포수 모델

SECTION 01 프로젝트의 진행 단계

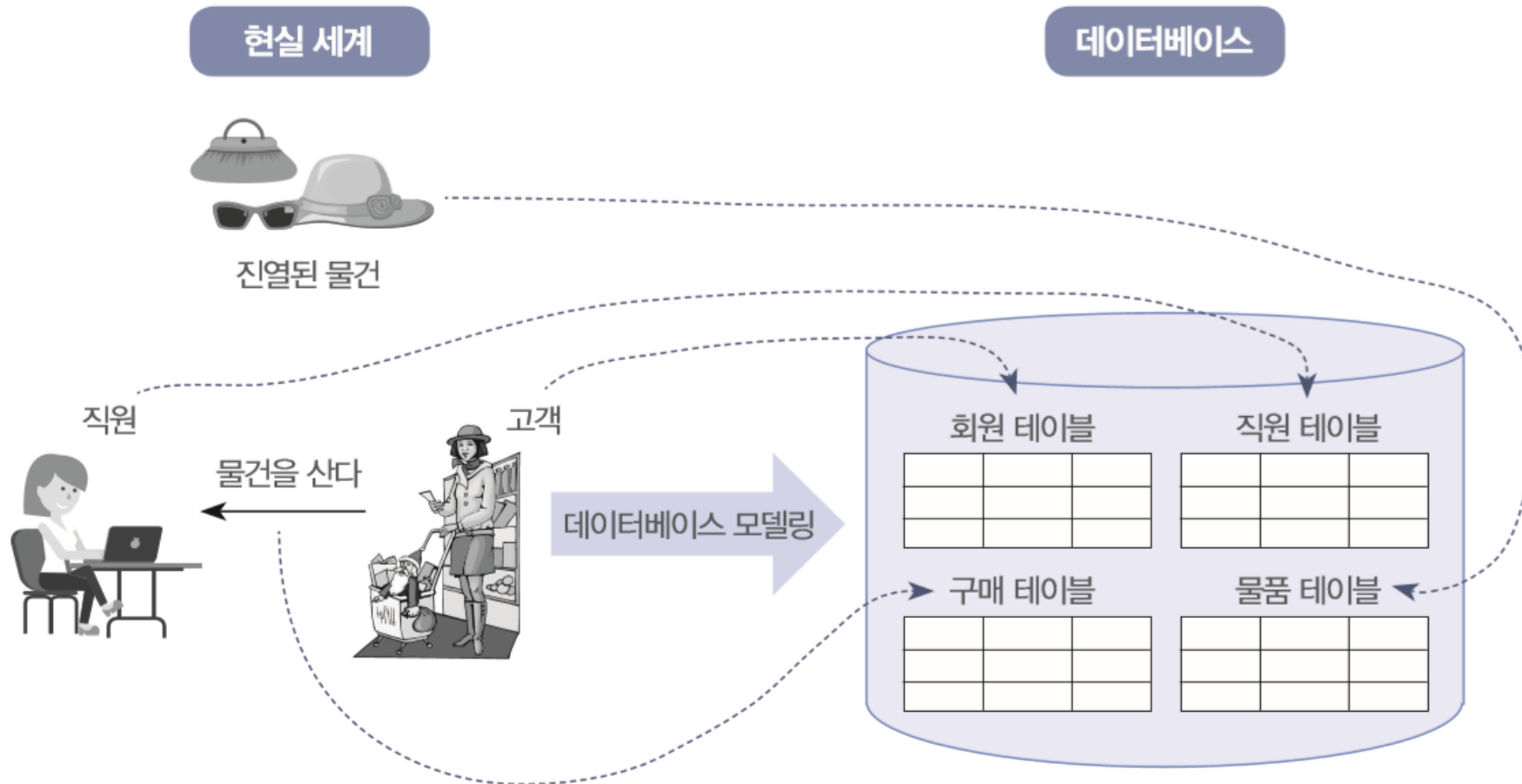
폭포수 모델 (Waterfall Model)

- 가장 오래되고 전통적으로 사용되는 소프트웨어 개발 모델
 - 폭포가 떨어지듯이 각 단계가 끝나면 다음 단계로 진행
- 장점
 - 각 단계가 명확히 구분되어 **프로젝트의 진행 단계가 명확해짐**
- 단점
 - 문제점이 발생할 경우 다시 **앞 단계로 거슬러 올라가기가 어려움**
 - 문제점이 대부분 **프로그램 구현 단계나 테스트 단계**에서 발생
 - 대부분의 문제점을 **업무 분석단계**에서 다시 시작하여 해결

SECTION 02 데이터베이스 모델링

데이터베이스 모델링(데이터 모델링) 개념

- 현 세계에서 사용되는 작업이나 사물들을 DBMS의 데이터베이스 개체로 옮기기 위한 과정



[그림 4-2] 데이터베이스 모델링의 개념

SECTION 02 데이터베이스 모델링

데이터베이스 모델링 실습

- 개념적 모델링
 - 업무 분석 단계에 포함
- 논리적 모델링
 - 업무 분석의 후반부와 시스템 설계의 전반부에 걸쳐 진행
- 물리적 모델링
 - 시스템 설계의 후반부에 주로 진행

SECTION 02 데이터베이스 모델링

쇼핑몰 데이터 예제

- 방문 내역 & 구매내역 데이터
 - 메모장이나 엑셀로 작성되었다 가정
- 기록된 내용에서 물건 구매 내역이 없는 고객 위로 정렬
 - L자형 테이블이 되어 낭비되는 공간 생김
- L자형 테이블을 빈칸이 있는 곳과 없는 곳으로 분류
 - 고객테이블, 구매테이블로 분류하여 공간 절약
 - 고객 테이블 중복 제거
 - 기본 키 (PK, Primary Key) 필요
 - 고객 이름을 고객을 구분할 수 있는 구분자로 설정
 - 각 행을 구분하는 유일한 값
 - 기본 키의 조건은 중복되지 않고 비어있지 않아야 함
 - 구매 테이블에 '누가 구매했는지' 표기 위해 고객 이름 필요

SECTION 02 데이터베이스 모델링

쇼핑몰 데이터 예제

- 테이블 간의 업무적인 연관성(Relation) 정의
 - 주 (Master)가 되는 쪽이 부모 테이블
 - ex) 고객이 물건을 소유 (O) , 물건이 고객을 소유 (X)
 - 주가 되는 고객 테이블이 부모, 상세가 되는 구매 테이블이 자식이 됨 (1:N 모델)
 - 기본 키 (PK, Primary Key)
 - 중복되지 않고 비어있지 않아야 함
 - 외래 키 (FK, Foreign Key)
 - 외래 키로 부모 테이블에서 유일하게 하나의 정보를 얻을 수 있음
 - 제약조건
 - 새로운 데이터 들어갈 때는 부모 테이블에 먼저 넣어야 함
 - 데이터 삭제 시에는 자식 테이블에서도 지워야 함

SECTION 02 데이터베이스 모델링

쇼핑몰 데이터 예제

- 완성된 고객 테이블과 구매 테이블의 구조 정의

테이블 이름	열 이름	데이터 형식	Null 허용	기타
고객 테이블 (userTBL)	고객 이름(userName)	문자(최대 3글자)	X	PK
	출생년도(birthYear)	숫자(정수)	X	
	주소(addr)	문자(최대 2글자)	X	
	연락처(mobile)	문자(최대 12글자)	O	
구매 테이블 (buyTBL)	고객 이름(userName)	문자(최대 3글자)	X	FK
	구매한 물건(prodName)	문자(최대 3글자)	X	
	단가(price)	숫자(정수)	X	
	수량(amount)	숫자(정수)	X	

[표 4-1] 데이터베이스 설계로 완료된 2개의 테이블 설계

SQL 기본

- SECTION 01 SELECT문

- 1.1 원하는 데이터를 가져와 주는 기본적인 <SELECT... FROM>

- 1.2 특정한 조건의 데이터만 조회하는 <SELECT... FROM... WHERE>

- 1.3 GROUP BY 및 HAVING 그리고 집계 함수

- 1.4 SQL의 분류

Contents

● CHAPTER 06 SQL 기본

● SECTION 02 데이터의 변경을 위한 SQL문

2.1 데이터의 삽입 : INSERT

2.2 데이터의 수정 : UPDATE

2.3 데이터의 삭제 : DELETE FROM

2.4 조건부 데이터 입력, 변경

● SECTION 03 WITH절과 CTE

3.1 WITH절과 CTE 개요

3.2 비재귀적 CTE

CHAPTER 06 SQL 기본

데이터베이스를 운영하기 위한 기본적인 SQL문에 대하여 알아본다.

SECTION 01 SELECT문

<SELECT... FROM>

- 원하는 데이터를 가져와 주는 기본적인 구문
- 가장 많이 사용되는 구문
- 데이터베이스 내 테이블에서 원하는 정보 추출하는 명령

```
SELECT select_expr  
  [FROM table_references]  
  [WHERE where_condition]  
  [GROUP BY {col_name | expr | position}]  
  [HAVING where_condition]  
  [ORDER BY {col_name | expr | position}]
```



```
SELECT 열 이름  
FROM 테이블이름  
WHERE 조건
```

SECTION 01 SELECT문

USE 구문

- SELECT문 학습 위해 사용할 데이터베이스 지정
- 지정해 놓은 후 특별히 다시 USE문 사용하거나 다른 DB를 사용하겠다고 명시하지 않는 이상 모든 SQL문은 지정 DB에서 수행

```
USE 데이터베이스_이름;
```

- employees를 사용하기 위해서는 쿼리 창에 다음과 같이 입력한다.

```
USE employees;
```


SECTION 01 SELECT문

SELECT와 FROM

- SELECT *

- 선택된 DB가 employees 라면 다음 두 쿼리는 동일

```
SELECT * FROM employees.titles;  
SELECT * FROM titles;
```

- SELECT 열 이름

- 테이블에서 필요로 하는 열만 가져오기 가능

```
SELECT first_name FROM employees;
```

- 여러 개의 열을 가져오고 싶을 때는 콤마로 구분

```
SELECT first_name, last_name, gender FROM employees;
```

- 열 이름의 순서는 출력하고 싶은 순서대로 배열 가능

SECTION 01 SELECT문

SELECT와 FROM

- 주석(Remark)

여기서 잠깐

주석(Remark)

MySQL은 '--' 이후부터 주석으로 처리된다. 주로 코드에 설명을 달거나 잠시 해당 부분의 실행을 막고 싶을 때 사용한다. 주의할 점은 -- 뒤에 바로 붙여서 쓰면 안되며, 공백이 하나 이상 있어야 한다.

-- 한 줄 주석 연습

```
SELECT first_name, last_name, gender -- 이름과 성별 열을 가져옴  
FROM employees;
```

여러 줄 주석은 '/* */'로 묶는다.

/* 블록 주석 연습

```
SELECT first_name, last_name, gender  
FROM employees;  
*/
```

주석으로 묶이면 해당 글자들은 모두 회색으로 보인다.

SECTION 01 SELECT문

DB, TABLE, 열의 이름이 확실하지 않을 때 조회하는 방법

- 현재 서버에 어떤 DB가 있는지 보기
 - SHOW DATABASES;
- 현재 서버에 어떤 TABLE이 있는지 보기
 - 데이터베이스에 있는 테이블 정보 조회
 - SHOW TABLE STATUS;
 - 테이블 이름만 간단히 보기
 - SHOW TABLES;
- employees 테이블의 열이 무엇이 있는지 확인
 - DESCRIBE employees; 또는 DESC employees;
- Workbench의 [Navigator]로 확인 가능 하나 명령어를 알아두면 Linux 명령어 모드에서 사용 가능

SECTION 01 SELECT문

특정 조건의 데이터만 조회 - <SELECT ... FROM ... WHERE>

◦ 기본적인 WHERE절

- 조회하는 결과에 특정한 조건을 줘서 원하는 데이터만 보고 싶을 때 사용
- SELECT 필드이름 FROM 테이블이름 WHERE 조건식;

● ex)

```
SELECT * FROM usertbl WHERE name = '김경호';
```

◦ 관계 연산자의 사용

- OR 연산자 : '...했거나', '... 또는'
- AND 연산자 : '...하고', '...면서', '... 그리고'
- 조건 연산자(=, <, >, <=, >=, < >, != 등)와 관계 연산자(NOT, AND, OR 등)를 조합하여 데이터를 효율적으로 추출 가능

● ex)

```
SELECT userID, Name FROM usertbl WHERE birthYear >= 1970 AND height >= 182;
```

SECTION 01 SELECT문

특정 조건의 데이터만 조회 - <SELECT ... FROM ... WHERE>

- BETWEEN... AND와 IN() 그리고 LIKE

- 데이터가 숫자로 구성되어 있으며 연속적인 값 : **BETWEEN ... AND** 사용

- ex)

```
SELECT name, height FROM usertbl WHERE height BETWEEN 180 AND 183;
```

- 이산적인(Discrete) 값의 조건 : **IN()** 사용

- ex)

```
SELECT name, addr FROM usertbl WHERE addr IN ('경남','전남','경북');
```

- 문자열의 내용 검색 : **LIKE** 사용(문자뒤에 % - 무엇이든 허용, 한 글자와 매치 '_' 사용)

- ex)

```
SELECT name, height FROM usertbl WHERE name LIKE '김%';
```

SECTION 01 SELECT문

ANY/ALL/SOME ,서브쿼리(SubQuery, 하위쿼리)

○ 서브쿼리

- 쿼리문 안에 또 쿼리문이 들어 있는 것
- 서브쿼리 사용하는 쿼리로 변환 예제
 - ex) 김경호보다 키가 크거나 같은 사람의 이름과 키 출력
 - WHERE 조건에 김경호의 키를 직접 써주는 것을 쿼리로 해결

```
SELECT name, height FROM usertbl WHERE height > 177;
```



```
SELECT name, height FROM usertbl  
WHERE height > (SELECT height FROM usertbl WHERE Name = '김경호');
```

- 서브쿼리의 결과가 둘 이상이 되면 에러 발생

SECTION 01 SELECT문

ANY/ALL/SOME ,서브쿼리(SubQuery, 하위쿼리)

- ANY
 - 서브쿼리의 여러 개의 결과 중 한 가지만 만족해도 가능
 - SOME은 ANY와 동일한 의미로 사용
 - '= ANY(서브쿼리)'는 'IN(서브쿼리)'와 동일한 의미
- ALL
 - 서브쿼리의 결과 중 여러 개의 결과를 모두 만족해야 함

SECTION 01 SELECT문

원하는 순서대로 정렬하여 출력 : ORDER BY

◦ ORDER BY절

- 결과물에 대해 영향을 미치지 않는고 출력되는 순서를 조절하는 구문
- 기본적으로 오름차순 (ASCENDING) 정렬
- 내림차순(DESCENDING)으로 정렬하려면 **열 이름 뒤에 DESC**
- ORDER BY 구문을 혼합해 사용하는 구문도 가능
 - 키가 큰 순서로 정렬하되 만약 키가 같을 경우 이름 순으로 정렬

```
SELECT name, height FROM usertbl ORDER BY height DESC, name ASC;
```

- ASC(오름차순)는 디폴트 값이므로 생략 가능

SECTION 01 SELECT문

- 중복된 것은 하나만 남기는 DISTINCT
 - 중복된 것을 골라서 세기 어려울 때 사용하는 구문
 - 테이블의 크기가 클수록 효율적
 - 중복된 것은 1개씩만 보여주면서 출력
- 출력하는 개수를 제한하는 LIMIT
 - 일부를 보기 위해 여러 건의 데이터를 출력하는 부담 줄임
 - 상위의 N개만 출력하는 'LIMIT N' 구문 사용
 - 개수의 문제보다는 MySQL의 부담을 많이 줄여주는 방법
- 테이블을 복사하는 CREATE TABLE ... SELECT
 - 테이블을 복사해서 사용할 경우 주로 사용
 - CREATE TABLE 새로운 테이블 (SELECT 복사할 열 FROM 기존테이블)
 - 지정한 일부 열만 복사하는 것도 가능
 - PK나 FK 같은 제약 조건은 복사되지 않음

SECTION 01 SELECT문

GROUP BY 및 HAVING 그리고 집계 함수

GROUP BY절

- 그룹으로 묶어주는 역할
- 집계 함수(Aggregate Function)와 함께 사용
 - 효율적인 데이터 그룹화 (Grouping)
 - ex) 각 사용자 별로 구매한 개수를 합쳐 출력

```
SELECT userID, SUM(amount) FROM buytbl GROUP BY userID;
```



	userID	SUM(amount)
▶	BBK	19
	EJW	4
	JYP	1
	KBS	6
	SSK	5

- 읽기 좋게 하기 위해 별칭(Alias) AS 사용

```
SELECT userID AS '사용자 아이디', SUM(amount) AS '총 구매 개수'  
FROM buytbl GROUP BY userID;
```



	사용자 아이디	총 구매 개수
▶	BBK	19
	EJW	4
	JYP	1
	KBS	6
	SSK	5

SECTION 01 SELECT문

GROUP BY 및 HAVING 그리고 집계 함수

- GROUP BY와 함께 자주 사용되는 집계 함수

함수명	설명
AVG()	평균을 구한다.
MIN()	최소값을 구한다.
MAX()	최대값을 구한다.
COUNT()	행의 개수를 센다.
COUNT(DISTINCT)	행의 개수를 센다(중복은 1개만 인정).
STDEV()	표준편차를 구한다.
VAR_SAMP()	분산을 구한다.

[표 6-1] GROUP BY와 함께 사용되는 집계 함수

- ex) 전체 구매자가 구매한 물품의 개수 평균

```
USE sqlldb;  
SELECT AVG(amount) AS '평균 구매 개수' FROM buytbl ;
```



	평균 구매 개수
▶	2.9167

SECTION 01 SELECT문

GROUP BY 및 HAVING 그리고 집계 함수

◦ Having절

- WHERE와 비슷한 개념으로 조건 제한하는 것이지만, 집계 함수에 대해서 조건을 제한하는 것
- HAVING절은 꼭 GROUP BY절 다음에 나와야 함(순서 바뀌면 안됨)

◦ ROLLUP

- 총합 또는 중간 합계가 필요할 경우 사용
- GROUP BY절과 함께 WITH ROLLUP문 사용
 - ex) 분류(groupName) 별로 합계 및 그 총합 구하기

```
SELECT num, groupName, SUM(price * amount) AS '비용'
FROM buytbl
GROUP BY groupName, num
WITH ROLLUP;
```



	num	groupName	비용	
▶	1	NULL	60	
	10	NULL	60	
	12	NULL	60	
	NULL	NULL	180	소합계
	7	서적	75	
	8	서적	30	
	11	서적	15	
	NULL	서적	120	소합계
	5	의류	150	
	9	의류	50	
	NULL	의류	200	소합계
	2	전자	1000	
	3	전자	200	
	4	전자	1000	
	6	전자	800	
	NULL	전자	3000	소합계
	NULL	NULL	3500	총합계

SECTION 01 SELECT문

SQL의 분류

- DML (Data Manipulation Language, 데이터 조작 언어)
 - 데이터를 조작(선택, 삽입, 수정, 삭제)하는 데 사용되는 언어
 - DML 구문이 사용되는 대상은 **테이블의 행**
 - DML 사용하기 위해서는 **테이블이 정의되어 있어야 함**
 - SQL문 중 **SELECT, INSERT, UPDATE, DELETE**가 이 구문에 해당
 - 트랜잭션(Transaction)이 발생하는 SQL도 DML에 속함
 - 테이블의 데이터를 변경(입력/수정/삭제)할 때 실제 테이블에 완전히 적용하지 않고, **임시로 적용**시키는 것
 - 취소 가능

SECTION 01 SELECT문

SQL의 분류

- DDL (Data Definition Language, 데이터 정의 언어)
 - 데이터베이스, 테이블, 뷰, 인덱스 등의 데이터베이스 개체를 생성/삭제/변경하는 역할
 - **CREATE, DROP, ALTER** 자주 사용
 - **DDL은 트랜잭션 발생시키지 않음**
 - 되돌림(ROLLBACK)이나 완전적용(COMMIT) 사용 불가
 - 실행 즉시 MySQL에 적용
- DCL (Data Control Language, 데이터 제어 언어)
 - 사용자에게 어떤 권한을 부여하거나 빼앗을 때 주로 사용하는 구문
 - GRANT/REVOKE/DENY 구문

SECTION 02 데이터의 변경을 위한 SQL문

데이터의 삽입 : INSERT

◦ INSERT문의 기본

```
INSERT [INTO] 테이블[(열1, 열2, ...)] VALUES (값1, 값2 ...)
```

- 테이블 이름 다음에 나오는 열 생략 가능
 - 생략할 경우에 VALUES 다음에 나오는 값들의 순서 및 개수가 테이블이 정의된 열 순서 및 개수와 동일해야 함

◦ 자동으로 증가하는 AUTO_INCREMENT

- INSERT에서는 해당 열이 없다고 생각하고 입력
 - INSERT문에서 NULL 값 지정하면 자동으로 값 입력
- 1부터 증가하는 값 자동 입력
- 적용할 열이 PRIMARY KEY 또는 UNIQUE일 때만 사용가능
- 데이터 형은 숫자 형식만 사용 가능

SECTION 02 데이터의 변경을 위한 SQL문

데이터의 삽입 : INSERT

- 대량의 샘플 데이터 생성

- INSERT INTO ... SELECT 구문 사용

형식:

```
INSERT INTO 테이블이름 (열 이름1, 열 이름2, ...)  
SELECT문 ;
```

- 다른 테이블의 데이터를 가져와 대량으로 입력하는 효과
- SELECT문의 열의 개수 = INSERT 할 테이블의 열의 개수
- 테이블 정의 까지 생략 하려면 CREATE TABLE ... SELECT 구문을 사용

SECTION 02 데이터의 변경을 위한 SQL문

데이터의 수정 : UPDATE

- 기존에 입력되어 있는 값 변경하는 구문

```
UPDATE 테이블이름  
  SET 열1=값1, 열2=값2 ...  
  WHERE 조건 ;
```

- **WHERE절 생략 가능하나 WHERE절 생략하면 테이블의 전체 행의 내용 변경됨**
 - 실무에서 실수가 종종 일어남, 주의 필요
 - 원상태로 복구하기 복잡하며, 다시 되돌릴 수 없는 경우도 있음

SECTION 02 데이터의 변경을 위한 SQL문

데이터의 삭제 : DELETE FROM

- 행 단위로 데이터 삭제하는 구문

```
DELETE FROM 테이블이름 WHERE 조건;
```

- WHERE절 생략되면 전체 데이터를 삭제함
- 테이블을 삭제하는 경우의 속도 비교
 - DML문인 DELETE는 트랜잭션 로그 기록 작업 때문에 삭제 느림
 - DDL문인 DROP과 TRUNCATE문은 트랜잭션 없어 빠름
 - 테이블 자체가 필요 없을 경우에는 DROP 으로 삭제
 - 테이블의 구조는 남겨놓고 싶다면 TRUNCATE로 삭제하는 것이 효율적

SECTION 02 데이터의 변경을 위한 SQL문

조건부 데이터 입력, 변경

- 기본 키가 중복된 데이터를 입력한 경우
 - 오류로 입력 불가
- 대용량 데이터 처리의 경우 에러 발생하지 않은 구문 실행
 - INSERT IGNORE문
 - 에러 발생해도 다음 구문으로 넘어가게 처리
 - 에러 메시지 보면 적용되지 않은 구문이 어느 것인지 구분 가능
 - ON DUPLICATE KEY UPDATE 구문
 - 기본 키가 중복되면 데이터를 수정되도록 하는 구문도 활용 가능

SECTION 03 WITH절과 CTE

WITH절과 CTE 개요

- WITH절은 CTE(Common Table Expression)를 표현하기 위한 구문
- MySQL 8.0 이후부터 사용 가능하게 됨
- CTE는 기존의 뷰, 파생 테이블, 임시 테이블 등을 대신할 수 있으며 간결한 식으로 보여짐
- CTE는 ANSI-SQL99 표준(기존 SQL은 ANSI-SQL92 기준)
- CTE는 비재귀적 CTE와 재귀적 CTE가 있지만 주로 사용되는 것은 비재귀적 CTE

SECTION 03 WITH절과 CTE

비재귀적 CTE

- 단순한 형태, 복잡한 쿼리문장을 단순화하는데 적합

```
WITH CTE_테이블이름(열 이름)
AS
(
    <쿼리문>
)
SELECT 열 이름 FROM CTE_테이블이름 ;
```

- CTE는 뷰와 용도가 비슷하지만 개선된 점이 많음
- 뷰는 계속 존재해서 다른 구문에서도 사용 가능하지만, CTE와 파생 테이블은 구문이 끝나면 소멸됨
- 중복 CTE 허용됨