

IT CookBook, SQL Server로 배우는 데이터베이스 개론과 실습

[강의교안 이용 안내]

- 본 강의교안의 저작권은 한빛아카데미(주)에 있습니다.
- 이 자료를 무단으로 전제하거나 배포할 경우 저작권법 136조에 의거하여 최고 5년 이하의 징역 또는 5천만원 이하의 벌금에 처할 수 있고 이를 병과(併科)할 수도 있습니다.



Chapter3. SQL 기초

SQL Server로 배우는 데이터베이스 개론과 실습

목차

1. SQL 학습을 위한 준비
2. SQL 개요
3. 데이터 조작용어 - 검색
4. 데이터 정의어
5. 데이터 조작용어 - 삽입, 수정, 삭제

학습목표

- SQL의 개념과 주요 명령어를 알아본다.
- SELECT 문을 이용하여 질의를 처리하는 방법을 알아본다.
- 집계 함수와 GROUP BY 문을 이용하여 질의를 처리하는 방법을 알아본다.
- 두 개 이상의 테이블을 조회하여 질의를 처리하는 방법을 알아본다.
- DDL을 이용하여 테이블의 구조를 정의하고 변경하는 방법을 알아본다.
- DML을 이용하여 데이터를 삽입, 수정, 삭제하는 방법을 알아본다.

01. SQL 학습을 위한 준비

- 마당서점의 데이터
- 누가 어떤 정보를 원하는가?
- SQL Server와 샘플 데이터 설치

01. SQL 학습을 위한 준비



그림 3-1 마당서점 현황

01. SQL 학습을 위한 준비

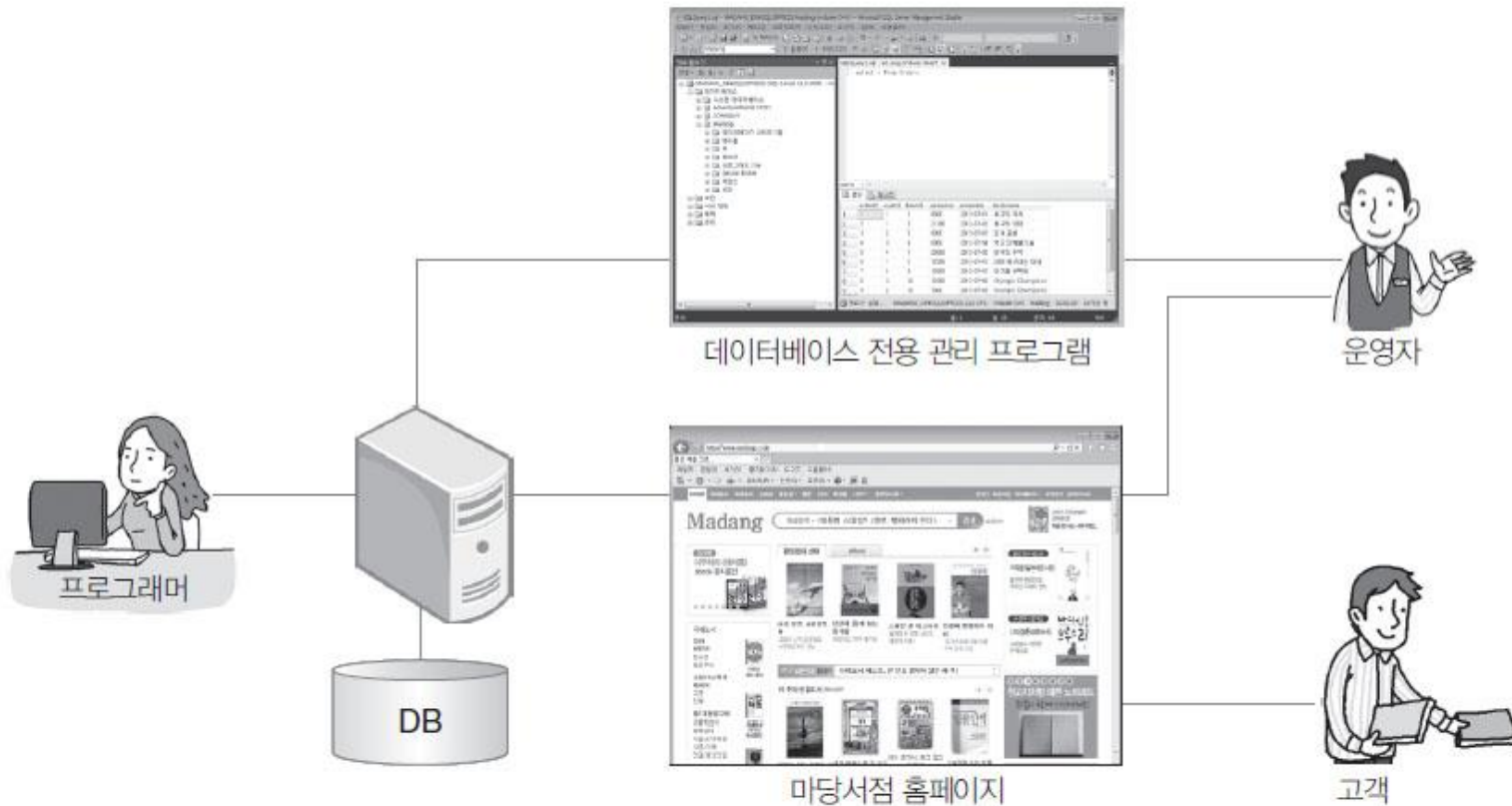


그림 3-2 마당서점 운영 시스템 환경

1.1 마당서점의 데이터

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	축구 아는 여자	나무수	13000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별기술	굿스포츠	6000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

그림 3-3 Book 테이블

custid	name	address	phone
1	박지성	영국 맨체스타	000-5000-0001
2	김연아	대한민국 서울	000-6000-0001
3	장미란	대한민국 강원도	000-7000-0001
4	추신수	미국 클리블랜드	000-8000-0001
5	박세리	대한민국 대전	NULL

그림 3-4 Customer 테이블

orderid	custid	bookid	saleprice	orderdate
1	1	1	6000	2013-07-01
2	1	3	21000	2013-07-03
3	2	5	8000	2013-07-03
4	3	6	6000	2013-07-04
5	4	7	20000	2013-07-05
6	1	2	12000	2013-07-07
7	4	8	13000	2013-07-07
8	3	10	12000	2013-07-08
9	2	10	7000	2013-07-09
10	3	8	13000	2013-07-10

그림 3-4 Order 테이블

1.1 마당서점의 데이터

Book(bookid, bookname, publisher, price)

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000

Orders(orderid, custid, bookid, saleprice, orderdate)

orderid	custid	bookid	saleprice	orderdate
1	1	1	6000	2013-07-01

Customer(custid, name, address, phone)

custid	name	address	phone
1	박지성	영국 맨체스터	000-5000-0001

그림 3-6 마당서점의 데이터 구성도

1.2 누가 어떤 정보를 원하는가?

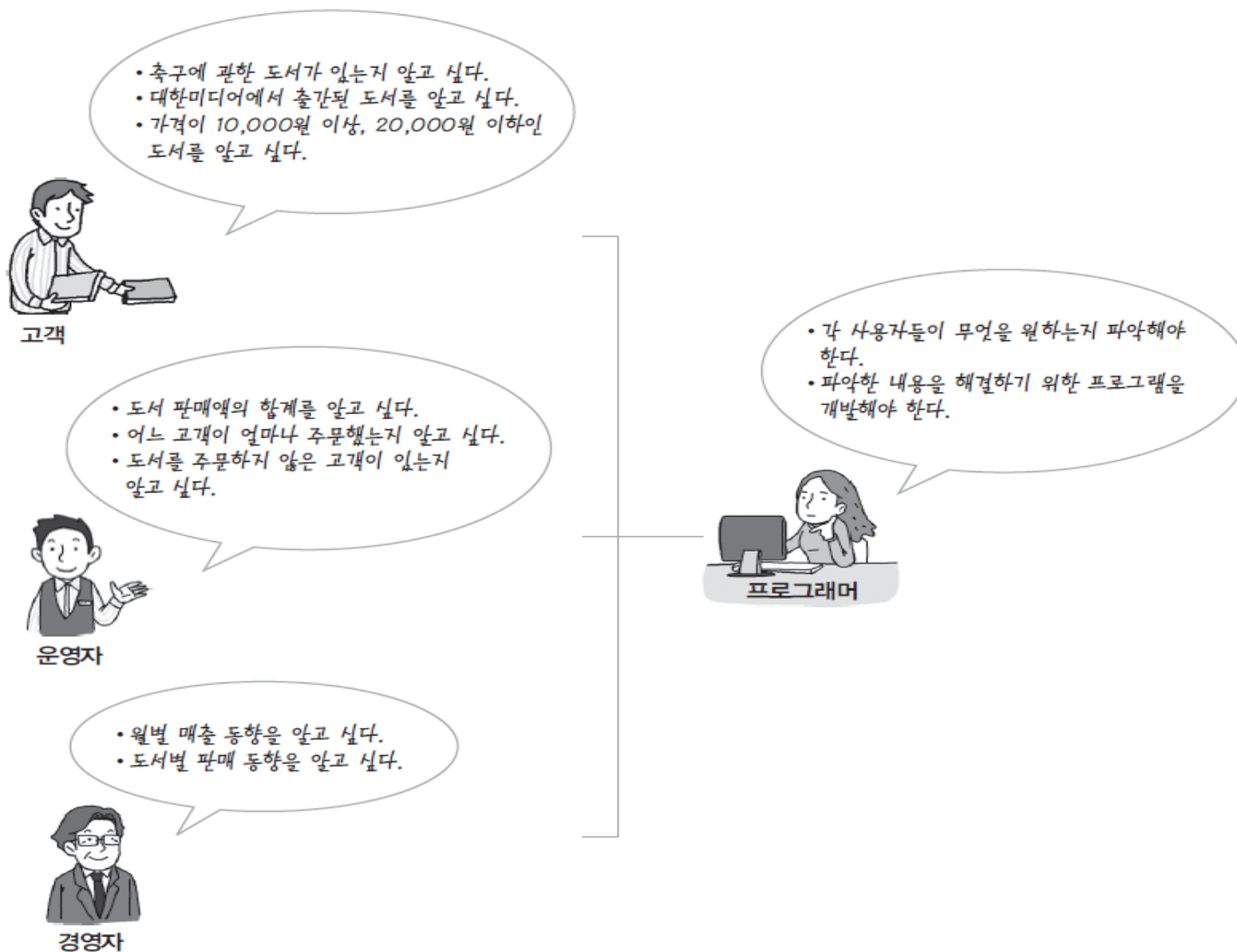
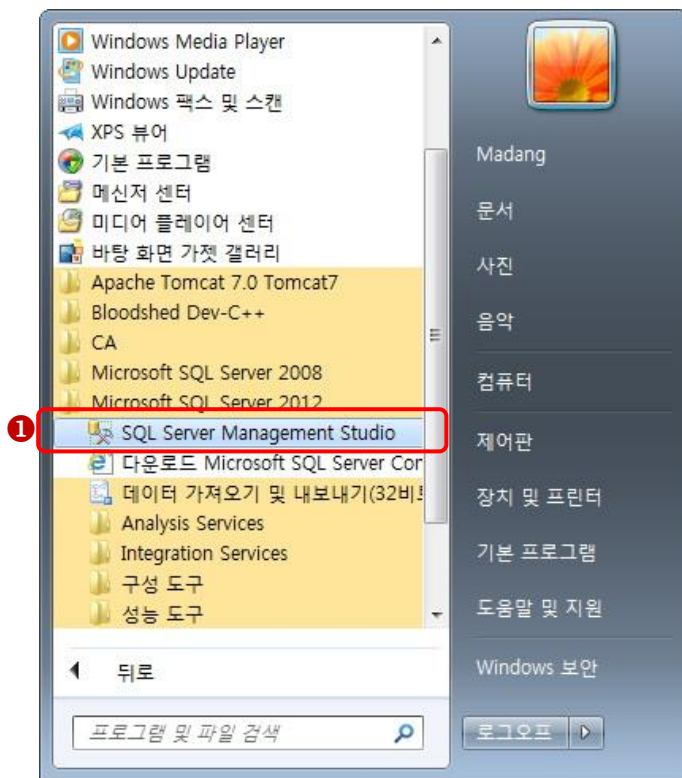


그림 3-7 사용자 그룹별로 원하는 정보

1.3 SQL Server와 샘플 데이터 설치

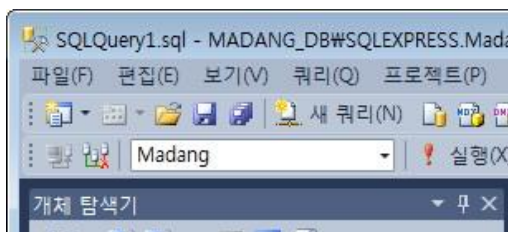
① SQL Server 시작



② 쿼리창 열기

1.3 SQL Server와 샘플 데이터 설치

③ 데이터베이스 선택



④ SQL 문 입력 및 결과 확인

A screenshot of the Microsoft SQL Server Management Studio (SSMS) interface. The 'SQLQuery1.sql' file is open, showing the following query:

```
SELECT *  
FROM Customer
```

The 'Results' tab is selected, displaying the query results in a table format:

	custid	name	address	phone
1	1	박지성	영국 맨체스터	000-5000-0001
2	2	김연아	대한민국 서울	000-6000-0001
3	3	장미란	대한민국 강원도	000-7000-0001
4	4	추신수	미국 클리블랜드	000-8000-0001
5	5	박세리	대한민국 대전	NULL

The 'Properties' tab is also visible on the right, showing the connection details for 'MADANG_D'.

Red annotations are present on the image:

- 쿼리창 (Query Window) pointing to the SQL query editor.
- 결과 (Results) pointing to the query results table.
- 쿼리결과창 (Query Results Window) pointing to the results table.
- 속성창 (Properties Window) pointing to the properties pane on the right.
- 개체탐색기 (Object Explorer) pointing to the left pane showing the database structure.

02. SQL 개요

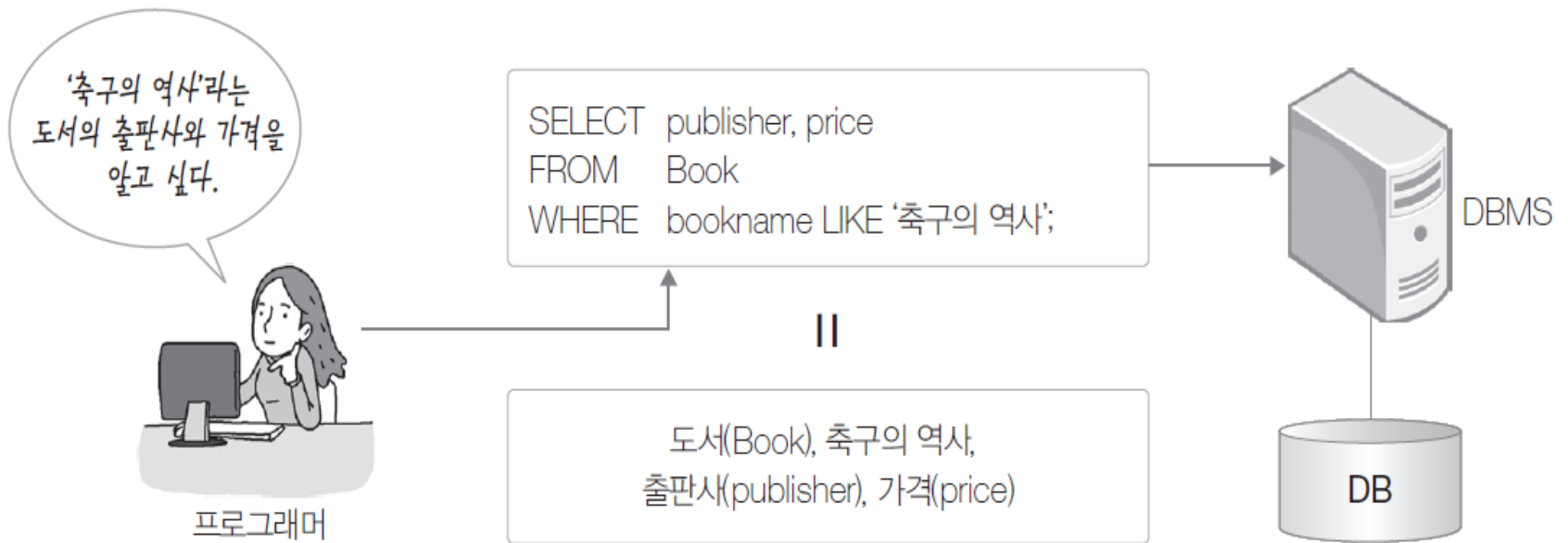


그림 3-11 SQL을 사용해 자료를 찾는 과정

02. SQL 개요

표 3-1 SQL과 일반 프로그래밍 언어의 차이점

	SQL	일반 프로그래밍 언어
용도	데이터베이스에서 데이터를 추출하여 문제 해결	모든 문제 해결
입출력	입력은 테이블, 출력도 테이블	모든 형태의 입출력 가능
번역	DBMS	컴파일러
사용 예	SELECT * FROM Book;	Int main() {...}

■ SQL 기능에 따른 분류

- 데이터 정의어(DDL) : 테이블이나 관계의 구조를 생성하는 데 사용하며 CREATE, ALTER, DROP 문 등이 있다.
- 데이터 조작어(DML) : 테이블에 데이터를 검색, 삽입, 수정, 삭제하는 데 사용하며 SELECT, INSERT, DELETE, UPDATE 문 등이 있다. 여기서 SELECT 문은 특별히 질의어(query)라고 부른다.
- 데이터 제어어(DCL) : 데이터의 사용 권한을 관리하는 데 사용하며 GRANT, REVOKE 문 등이 있다.

02. SQL 개요

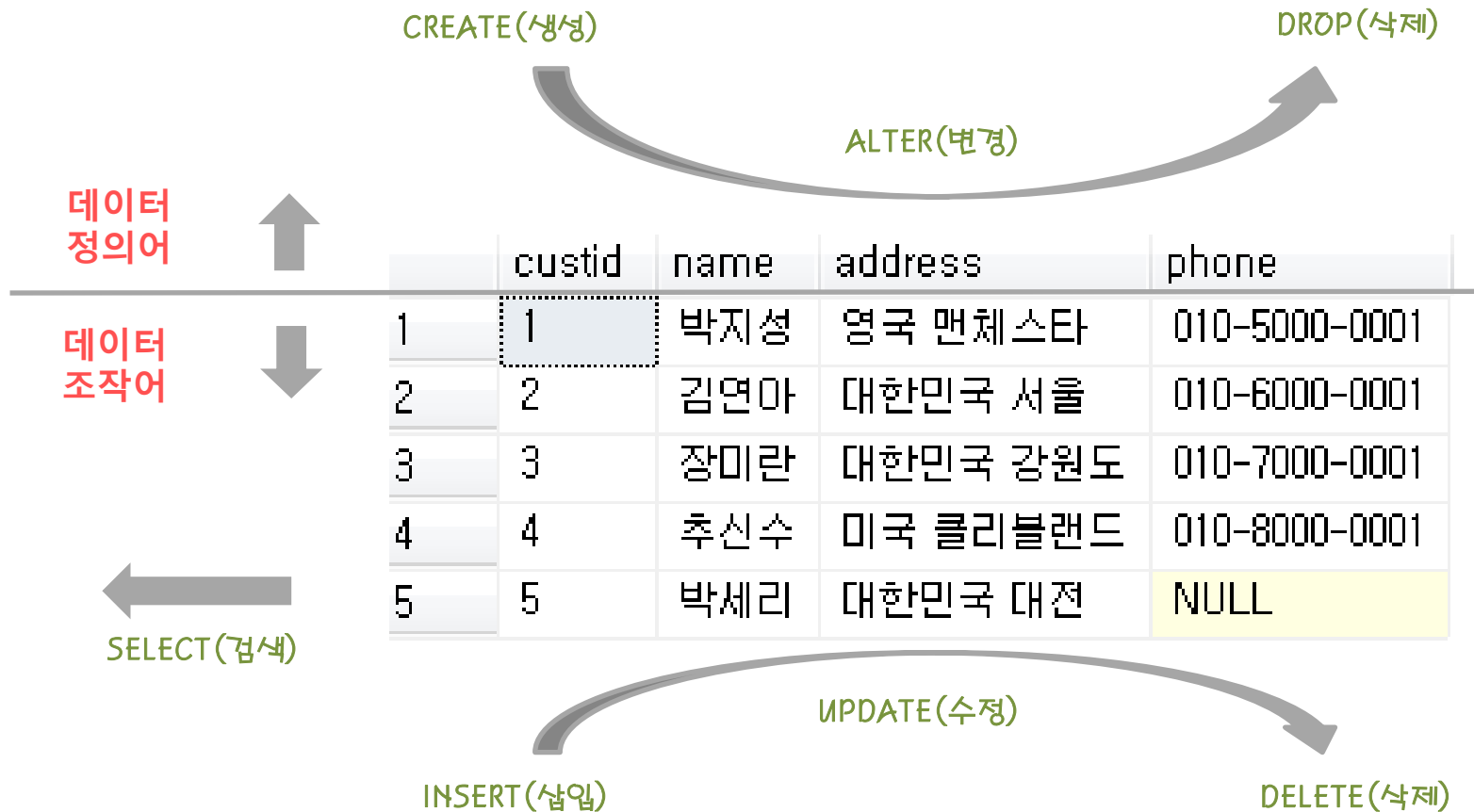


그림 3-12 데이터 정의어와 데이터 조작어의 주요 명령어

02. SQL 개요

EX) 김연아 고객의 전화번호를 찾으시오.

```
SELECT  phone
FROM    Customer
Where   name='김연아'
```

① FROM Customer

custid	name	address	phone
1	박지성	영국 맨체스터	000-5000-0001
2	김연아	대한민국 서울	000-6000-0001
3	장미란	대한민국 강원도	000-7000-0001
4	추신수	미국 클리블랜드	000-8000-0001
5	박세리	대한민국 대전	NULL



② WHERE name='김연아'

custid	name	address	phone
2	김연아	대한민국 서울	000-6000-0001



③ SELECT phone

phone
000-6000-0001

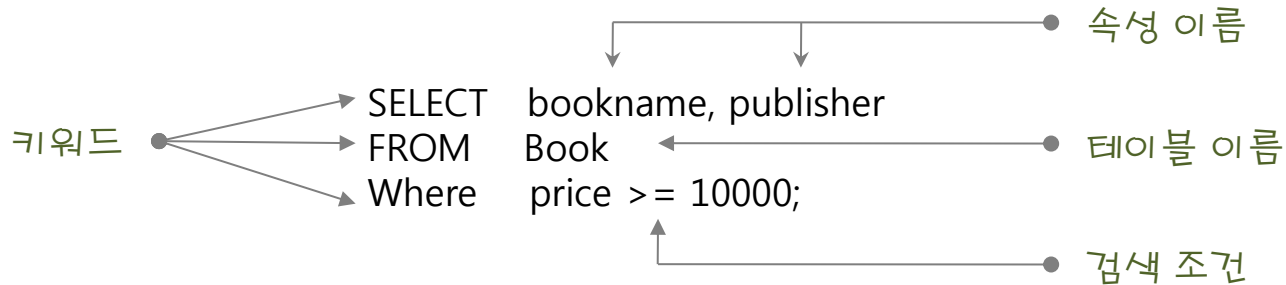
그림 3-13 SQL 문의 내부적 실행 순서

03. 데이터 조작용어 - 검색

- SELECT 문
- 집계 함수와 GROUP BY
- 두 개 이상 테이블에서 SQL 질의

03. 데이터 조작용어 - 검색

■ SELECT 문의 구성 요소



■ SELECT 문의 기본 문법

```
SELECT [ALL | DISTINCT] 속성이름(들)
FROM 테이블이름(들)
[WHERE 검색조건(들)]
[GROUP BY 속성이름]
[HAVING 검색조건(들)]
[ORDER BY 속성이름 [ASC | DESC]]
```

[] : 대괄호 안의 SQL 예약어들은 선택적으로 사용한다.
| : 선택 가능한 문법들 중 한 개를 사용할 수 있다.

3.1.1 SELECT/FROM _서점에 어떤 도서가 있는지 알고 싶다

질의 3-1 모든 도서의 이름과 가격을 검색하시오.

```
SELECT    bookname, price
FROM      Book;
```

bookname	price
축구의 역사	7000
축구 마는 여자	13000
축구의 이해	22000
골프 바이블	35000
피겨 교본	8000
역도 단계별 기술	6000
야구의 추억	20000
야구를 부탁해	13000
올림픽 이야기	7500
Olympic Champ...	13000

(질의 3-1 변형) 모든 도서의 가격과 이름을 검색하시오.

```
SELECT    price, bookname;
FROM      Book;
```

price	bookname
7000	축구의 역사
13000	축구 마는 여자
22000	축구의 이해
35000	골프 바이블
8000	피겨 교본
6000	역도 단계별 기술
20000	야구의 추억
13000	야구를 부탁해
7500	올림픽 이야기
13000	Olympic Champions

3.1.1 SELECT/FROM _서점에 어떤 도서가 있는지 알고 싶다

질의 3-2 모든 도서의 도서번호, 도서이름, 출판사, 가격을 검색하시오.

```
SELECT    bookid, bookname, publisher, price
FROM      Book;
```

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	축구 아는 여자	나무수	13000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별 기술	굿스포츠	6000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

```
SELECT    *
FROM      Book;
```

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	축구 아는 여자	나무수	13000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별 기술	굿스포츠	6000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

3.1.1 SELECT/FROM_서점에 어떤 도서가 있는지 알고 싶다

질의 3-3 도서 테이블에 있는 모든 출판사를 검색하시오.

```
SELECT publisher
FROM Book;
```

publisher
굿스포츠
나무수
대한미디어
대한미디어
굿스포츠
굿스포츠
이상미디어
이상미디어
삼성당
Pearson

※ 중복을 제거하고 싶으면 DISTINCT라는 키워드를 사용한다.

```
SELECT DISTINCT publisher
FROM Book;
```

publisher
굿스포츠
나무수
대한미디어
삼성당
이상미디어
Pearson

3.1.2 WHERE 조건_가격이 20,000원 미만인 도서가 무엇인지 알고 싶다

표 3-2 WHERE 절에 조건으로 사용할 수 있는 술어

술어	연산자	예
비교	=, <>, <, <=, >, >=	price < 20000
범위	BETWEEN	price BETWEEN 10000 AND 20000
집합	IN, NOT IN	price IN (10000, 20000, 30000)
패턴	LIKE	bookname LIKE '축구의 역사'
NULL	IS NULL, IS NOT NULL	price IS NULL
복합조건	AND, OR, NOT	(price < 20000) AND (bookname LIKE '축구의 역사')

■ 비교

질의 3-4 가격이 20,000원 미만인 도서를 검색하시오.

```
SELECT *  
FROM Book  
WHERE price < 20000;
```

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	2030 축구마는 여자	나무수	13000
5	피겨 교본	굿스포츠	8000
6	역도 단계별기술	굿스포츠	6000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

3.1.2 WHERE 조건_가격이 20,000원 미만인 도서가 무엇인지 알고 싶다

■ 범위

질의 3-5 가격이 10,000원 이상 20,000 이하인 도서를 검색하시오.

```
SELECT *  
FROM Book  
WHERE price BETWEEN 10000 AND 20000;
```

bookid	bookname	publisher	price
2	2030 축구마는 여자	나무수	13000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
10	Olympic Champions	Pearson	13000

※ BETWEEN은 논리 연산자인 AND를 사용할 수 있다.

```
SELECT *  
FROM Book  
WHERE price >= 10000 AND price <= 20000;
```

bookid	bookname	publisher	price
2	2030 축구마는 여자	나무수	13000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
10	Olympic Champions	Pearson	13000

3.1.2 WHERE 조건_가격이 20,000원 미만인 도서가 무엇인지 알고 싶다

■ 집합

질의 3-6 출판사가 '굿스포츠' 혹은 '대한미디어'인 도서를 검색하시오.

```
SELECT *  
FROM Book  
WHERE publisher IN ('굿스포츠', '대한미디어');
```

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별기술	굿스포츠	6000

※ 출판사가 '굿스포츠' 혹은 '대한미디어'가 아닌 출판사를 검색하시오.

```
SELECT *  
FROM Book  
WHERE publisher NOT IN ('굿스포츠', '대한미디어');
```

bookid	bookname	publisher	price
2	2030 축구마는 여자	나무수	13000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

3.1.2 WHERE 조건_가격이 20,000원 미만인 도서가 무엇인지 알고 싶다

■ 패턴

질의 3-7 '축구의 역사'를 출간한 출판사를 검색하시오.

```
SELECT    bookname, publisher
FROM      Book
WHERE     bookname LIKE '축구의 역사';
```

bookname	publisher
축구의 역사	굿스포츠

질의 3-8 도서이름에 '축구'가 포함된 출판사를 검색하시오.

```
SELECT    bookname, publisher
FROM      Book
WHERE     bookname LIKE '%축구%';
```

bookname	publisher
축구의 역사	굿스포츠
2030 축구마는 여자	나무수
축구의 이해	대한미디어

3.1.2 WHERE 조건_가격이 20,000원 미만인 도서가 무엇인지 알고 싶다

질의 3-9 도서이름의 왼쪽 두 번째 위치에 '구'라는 문자열을 갖는 도서를 검색하시오.

```
SELECT *  
FROM Book  
WHERE bookname LIKE '_구%';
```

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	2030 축구마는 여자	나무수	13000
3	축구의 이해	대한미디어	22000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000

표 3-3 와일드 문자의 종류

와일드 문자	의미	사용 예
+	문자열을 연결	'골프' + '바이블' : '골프 바이블'
%	0개 이상의 문자열과 일치	'%축구%' : 축구를 포함하는 문자열
[]	1개의 문자와 일치	'[0-5]%' : 0-5 사이 숫자로 시작하는 문자열
[^]	1개의 문자와 불일치	'[^0-5]%' : 0-5 사이 숫자로 시작하지 않는 문자열
_	특정 위치의 1개의 문자와 일치	'_구%' : 두 번째 위치에 '구'가 들어가는 문자열

3.1.2 WHERE 조건_가격이 20,000원 미만인 도서가 무엇인지 알고 싶다

■ 복합조건

질의 3-10 축구에 관한 도서 중 가격이 20,000원 이상인 도서를 검색하시오.

```
SELECT *  
FROM Book  
WHERE bookname LIKE '%축구%' AND price >= 20000;
```

bookid	bookname	publisher	price
3	축구의 이해	대한미디어	22000

질의 3-11 출판사가 '굿스포츠' 혹은 '대한미디어'인 도서를 검색하시오.

```
SELECT *  
FROM Book  
WHERE publisher='굿스포츠' OR publisher='대한미디어';
```

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별기술	굿스포츠	6000

3.1.3 ORDER BY _도서를 이름순으로 보고 싶다

질의 3-12 도서를 이름순으로 검색하시오.

```
SELECT      *
FROM        Book
ORDER BY    bookname
```

bookid	bookname	publisher	price
2	2030 축구마는 여자	나무수	13000
4	골프 바이블	대한미디어	35000
8	야구를 부탁해	이상미디어	13000
7	야구의 추억	이상미디어	20000
6	역도 단계별기술	굿스포츠	6000
9	올림픽 이야기	삼성당	7500
1	축구의 역사	굿스포츠	7000
3	축구의 이해	대한미디어	22000
5	피겨 교본	굿스포츠	8000
10	Olympic Champions	Pearson	13000

질의 3-13 도서를 가격순으로 검색하고, 가격이 같으면 이름순으로 검색하시오.

```
SELECT      *
FROM        Book
ORDER BY    price, bookname;
```

bookid	bookname	publisher	price
6	역도 단계별기술	굿스포츠	6000
1	축구의 역사	굿스포츠	7000
9	올림픽 이야기	삼성당	7500
5	피겨 교본	굿스포츠	8000
2	2030 축구마는 여자	나무수	13000
8	야구를 부탁해	이상미디어	13000
10	Olympic Champions	Pearson	13000
7	야구의 추억	이상미디어	20000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000

3.1.3 ORDER BY _도서를 이름순으로 보고 싶다

질의 3-14 도서를 가격의 내림차순으로 검색하시오. 만약 가격이 같다면 출판사의 오름차순으로 검색한다.

```
SELECT      *  
FROM        Book  
ORDER BY    price DESC, publisher ASC;
```

bookid	bookname	publisher	price
4	골프 바이블	대한미디어	35000
3	축구의 이해	대한미디어	22000
7	야구의 추억	이상미디어	20000
2	축구 아는 여자	나무수	13000
8	야구를 부탁해	이상미디어	13000
10	Olympic Champions	Pearson	13000
5	피겨 교본	굿스포츠	8000
9	올림픽 이야기	삼성당	7500
1	축구의 역사	굿스포츠	7000
6	역도 단계별 기술	굿스포츠	6000

3.2.1 집계 함수 _ 도서 판매액의 합계를 알고 싶다

질의 3-15 고객이 주문한 도서의 총 판매액을 구하시오.

```
SELECT SUM(saleprice)
FROM Orders;
```

(열 이름 없음)
118000

※ 의미 있는 열 이름을 출력하고 싶으면 속성이름의 별칭을 지칭하는 AS 키워드를 사용하여 열 이름을 부여한다.

```
SELECT SUM(saleprice) AS 총매출
FROM Orders;
```

총매출
118000

3.2.1 집계 함수 _ 도서 판매액의 합계를 알고 싶다

질의 3-16 2번 김연아 고객이 주문한 도서의 총 판매액을 구하시오.

```
SELECT SUM(saleprice) AS 총매출
FROM Orders
WHERE custid=2;
```

총매출
15000

질의 3-17 고객이 주문한 도서의 총 판매액, 평균값, 최저가, 최고가를 구하시오.

```
SELECT SUM(saleprice) AS Total,
        AVG(saleprice) AS Average,
        MIN(saleprice) AS Minimum,
        MAX(saleprice) AS Maximum
FROM Orders;
```

Total	Average	Minimum	Maximum
118000	11800	6000	21000

3.2.1 집계 함수 _ 도서 판매액의 합계를 알고 싶다

질의 3-18 마당서점의 도서 판매 건수를 구하시오.

```
SELECT    COUNT(*)  
FROM      Orders;
```

(열 이름 없음)

10

표 3-4 집계 함수의 종류

집계 함수	문법	사용 예
SUM	SUM([ALL DISTINCT] 속성이름)	SUM(price)
AVG	AVG([ALL DISTINCT] 속성이름)	AVG(price)
COUNT	COUNT({[ALL DISTINCT] 속성이름} *)	COUNT(*)
MAX	MAX([ALL DISTINCT] 속성이름)	MAX(price)
MIN	MIN([ALL DISTINCT] 속성이름)	MIN(price)

3.2.2 GROUP BY _어느 고객이 얼마나 주문했는지 알고 싶다

질의 3-19 고객별로 주문한 도서의 총 수량과 총 판매액을 구하시오.

```
SELECT    custid, COUNT(*) AS 도서수량, SUM(saleprice) AS 총액
FROM      Orders
GROUP BY  custid;
```

custid	도서수량	총액
1	3	39000
2	2	15000
3	3	31000
4	2	33000

orderid	custid	bookid	saleprice	orderdate
1	1	1	6000	2013-07-01
2	1	3	21000	2013-07-03
6	1	2	12000	2013-07-07
9	2	10	7000	2013-07-09
3	2	5	8000	2013-07-03
4	3	6	6000	2013-07-04
10	3	8	13000	2013-07-10
8	3	10	12000	2013-07-08
7	4	8	13000	2013-07-07
5	4	7	20000	2013-07-05

custid	도서수량	총액
1	3	39000
2	2	15000
3	3	31000
4	2	33000

그림 3-15 GROUP BY 절의 수행

3.2.2 GROUP BY_어느 고객이 얼마나 주문했는지 알고 싶다

질의 3-20 가격이 8,000원 이상인 도서를 구매한 고객에 대하여 고객별 주문 도서의 총 수량을 구하시오. 단, 두 권 이상 구매한 고객만 구한다.

```
SELECT    custid, COUNT(*) AS 도서수량
FROM      Orders
WHERE     saleprice >= 8000
GROUP BY  custid
HAVING    count(*) >= 2;
```

custid	도서수량
1	2
3	2
4	2

3.2.2 GROUP BY_어느 고객이 얼마나 주문했는지 알고 싶다

표 3-5 GROUP BY와 HAVING 절의 문법과 주의사항

문법	주의사항
GROUP BY <속성>	<p>GROUP BY로 튜플을 그룹으로 묶은 후 SELECT 절에는 GROUP BY에서 사용한 <속성>과 집계함수만 나올 수 있다.</p> <ul style="list-style-type: none">▪ 맞는 예 SELECT custid, SUM(saleprice) FROM Orders GROUP BY custid;• 틀린 예 SELECT bookid, SUM(saleprice) /* SELECT 절에 bookid 속성이 올 수 없다 */ FROM Orders GROUP BY custid;
HAVING <검색조건>	<p>WHERE 절과 HAVING 절이 같이 포함된 SQL 문은 검색조건이 모호해질 수 있다. HAVING 절은 ① 반드시 GROUP BY 절과 같이 작성해야 하고 ② WHERE 절보다 뒤에 나와야 한다. 그리고 ③ <검색조건>에는 SUM, AVG, MAX, MIN, COUNT와 같은 집계함수가 와야 한다.</p> <ul style="list-style-type: none">• 맞는 예 SELECT custid, COUNT(*) AS 도서수량 FROM Orders WHERE saleprice >= 8000 GROUP BY custid HAVING count(*) >= 2;• 틀린 예 SELECT custid, COUNT(*) AS 도서수량 FROM Orders HAVING count(*) >= 2 /* 순서가 틀렸다 */ WHERE saleprice >= 8000 GROUP BY custid;

3.3.1 조인_2개의 테이블을 합체해보자

- Customer 테이블을 Orders 테이블과 조건 없이 연결해보자. Customer와 Orders 테이블의 합체 결과 튜플의 개수는 고객이 다섯 명이고 주문이 열 개이므로 5×10 해서 50이 된다.

```
SELECT *
FROM Customer, Orders;
```

custid	name	address	phone	orderid	custid	bookid	saleprice	orderdate
1	박지성	영국 맨체스타	000-5000-0001	1	1	1	6000	2013-07-01
1	박지성	영국 맨체스타	000-5000-0001	2	1	3	21000	2013-07-03
1	박지성	영국 맨체스타	000-5000-0001	3	2	5	8000	2013-07-03
1	박지성	영국 맨체스타	000-5000-0001	4	3	6	6000	2013-07-04
1	박지성	영국 맨체스타	000-5000-0001	5	4	7	20000	2013-07-05
1	박지성	영국 맨체스타	000-5000-0001	6	1	2	12000	2013-07-07
1	박지성	영국 맨체스타	000-5000-0001	7	4	8	13000	2013-07-07
1	박지성	영국 맨체스타	000-5000-0001	8	3	10	12000	2013-07-08
1	박지성	영국 맨체스타	000-5000-0001	9	2	10	7000	2013-07-09
1	박지성	영국 맨체스타	000-5000-0001	10	3	8	13000	2013-07-10
2	김연아	대한민국 서울	000-6000-0001	1	1	1	6000	2013-07-01
2	김연아	대한민국 서울	000-6000-0001	2	1	3	21000	2013-07-03
2	김연아	대한민국 서울	000-6000-0001	3	2	5	8000	2013-07-03
2	김연아	대한민국 서울	000-6000-0001	4	3	6	6000	2013-07-04
2	김연아	대한민국 서울	000-6000-0001	5	4	7	20000	2013-07-05
2	김연아	대한민국 서울	000-6000-0001	6	1	2	12000	2013-07-07
2	김연아	대한민국 서울	000-6000-0001	7	4	8	13000	2013-07-07

중략

5	박세리	대한민국 대전	NULL	5	4	7	20000	2013-07-05
5	박세리	대한민국 대전	NULL	6	1	2	12000	2013-07-07
5	박세리	대한민국 대전	NULL	7	4	8	13000	2013-07-07
5	박세리	대한민국 대전	NULL	8	3	10	12000	2013-07-08
5	박세리	대한민국 대전	NULL	9	2	10	7000	2013-07-09
5	박세리	대한민국 대전	NULL	10	3	8	13000	2013-07-10

그림 3-16 Customer와 Orders 테이블의 합체

3.3.1 조인_2개의 테이블을 합체해보자

질의 3-21 고객과 고객의 주문에 관한 데이터를 모두 보이시오.

```
SELECT      *  
FROM        Customer, Orders  
WHERE       Customer.custid =Orders.custid;
```

custid	name	address	phone	orderid	custid	bookid	saleprice	orderdate
1	박지성	영국 맨체스타	000-5000-0001	1	1	1	6000	2013-07-01
1	박지성	영국 맨체스타	000-5000-0001	2	1	3	21000	2013-07-03
2	김연아	대한민국 서울	000-6000-0001	3	2	5	8000	2013-07-03
3	장미란	대한민국 강원도	000-7000-0001	4	3	6	6000	2013-07-04
4	추신수	미국 클리블랜드	000-8000-0001	5	4	7	20000	2013-07-05
1	박지성	영국 맨체스타	000-5000-0001	6	1	2	12000	2013-07-07
4	추신수	미국 클리블랜드	000-8000-0001	7	4	8	13000	2013-07-07
3	장미란	대한민국 강원도	000-7000-0001	8	3	10	12000	2013-07-08
2	김연아	대한민국 서울	000-6000-0001	9	2	10	7000	2013-07-09
3	장미란	대한민국 강원도	000-7000-0001	10	3	8	13000	2013-07-10

3.3.1 조인_2개의 테이블을 합체해보자

질의 3-22 고객과 고객의 주문에 관한 데이터를 고객별로 정렬하여 보이시오.

```
SELECT      *
FROM        Customer, Orders
WHERE       Customer.custid =Orders.custid
ORDER BY    Customer.custid;
```

custid	name	address	phone	orderid	custid	bookid	saleprice	orderdate
1	박지성	영국 맨체스타	000-5000-0001	1	1	1	6000	2013-07-01
1	박지성	영국 맨체스타	000-5000-0001	2	1	3	21000	2013-07-03
1	박지성	영국 맨체스타	000-5000-0001	6	1	2	12000	2013-07-07
2	김연아	대한민국 서울	000-6000-0001	9	2	10	7000	2013-07-09
2	김연아	대한민국 서울	000-6000-0001	3	2	5	8000	2013-07-03
3	장미란	대한민국 강원도	000-7000-0001	4	3	6	6000	2013-07-04
3	장미란	대한민국 강원도	000-7000-0001	10	3	8	13000	2013-07-10
3	장미란	대한민국 강원도	000-7000-0001	8	3	10	12000	2013-07-08
4	추신수	미국 클리블랜드	000-8000-0001	7	4	8	13000	2013-07-07
4	추신수	미국 클리블랜드	000-8000-0001	5	4	7	20000	2013-07-05

3.3.1 조인_2개의 테이블을 합체해보자

질의 3-23 고객의 이름과 고객이 주문한 도서의 가격을 검색하시오.

```
SELECT    name, saleprice
FROM      Customer, Orders
WHERE     Customer.custid =Orders.custid;
```

name	saleprice
박지성	6000
박지성	21000
김연아	8000
장미란	6000
추신수	20000
박지성	12000
추신수	13000
장미란	12000
김연아	7000
장미란	13000

질의 3-24 고객별로 주문한 모든 도서의 총 판매액을 구하고, 고객별로 정렬하시오.

```
SELECT    name, SUM(saleprice)
FROM      Customer, Orders
WHERE     Customer.custid =Orders.custid
GROUP BY  Customer.name
ORDER BY  Customer.name;
```

name	(열 이름 없음)
김연아	15000
박지성	39000
장미란	31000
추신수	33000

3.3.1 조인_2개의 테이블을 합체해보자

고객의 이름과 구매한 고객이 주문한 도서의 이름을 구하여라.

custid	name	address	phone
1	박지성	영국 맨체스타	000-5000-0001
2	김연아	대한민국 서울	000-6000-0001
3	장미란	대한민국 강원도	000-7000-0001
4	추신수	미국 콜리블랜드	000-8000-0001
5	박세리	대한민국 대전	NULL

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	축구 아는 여자	나무수	13000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별 기술	굿스포츠	6000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

orderid	custid	bookid	saleprice	orderdate
1	1	1	6000	2013-07-01
2	1	3	21000	2013-07-03
3	2	5	8000	2013-07-03
4	3	6	6000	2013-07-04
5	4	7	20000	2013-07-05
6	1	2	12000	2013-07-07
7	4	8	13000	2013-07-07
8	3	10	12000	2013-07-08
9	2	10	7000	2013-07-09
10	3	8	13000	2013-07-10

그림 3-17 마당서점 데이터 간의 연결

3.3.1 조인_2개의 테이블을 합체해보자

질의 3-25 고객의 이름과 고객이 주문한 도서의 이름을 구하시오.

```
SELECT    Customer.name, book.bookname
FROM      Customer, Orders, Book
WHERE     Customer.custid =Orders.custid
          AND Orders.bookid =Book.bookid;
```

name	bookname
박지성	축구의 역사
박지성	2030 축구하는 여자
박지성	축구의 이해
김연아	피겨 교본
장미란	역도 단계별기술
추신수	야구의 추억
추신수	야구를 부탁해
장미란	야구를 부탁해
장미란	Olympic Champions
김연아	Olympic Champions

질의 3-26 가격이 20,000원인 도서를 주문한 고객의 이름과 도서의 이름을 구하시오.

```
SELECT    Customer.name, book.bookname
FROM      Customer, Orders, Book
WHERE     Customer.custid =Orders.custid AND Orders.bookid =Book.bookid
          AND Orders.saleprice =20000;
```

name	bookname
추신수	야구의 추억

3.3.1 조인_2개의 테이블을 합체해보자

■ 외부조인

질의 3-27 도서를 구매하지 않은 고객을 포함하여 고객의 이름과 고객이 주문한 도서의 가격을 구하시오.

```
SELECT Customer.name, saleprice
FROM Customer LEFT OUTER JOIN
Orders ON Customer.custid =Orders.custid;
```

name	saleprice
박지성	6000
박지성	21000
박지성	12000
김연아	8000
김연아	7000
장미란	6000
장미란	12000
장미란	13000
추신수	20000
추신수	13000
박세리	NULL

3.3.1 조인_2개의 테이블을 합체해보자

표 3-6 조인 문법

명령	문법	설명
일반적인 조인	SELECT <속성들> FROM 테이블1, 테이블2 WHERE <조인조건> AND <검색조건>	SQL 문에서는 주로 동등조인을 사용한다. 두 가지 문법 중 하나를 사용할 수 있다.
	SELECT <속성들> FROM 테이블1 INNER JOIN 테이블2 ON <조인조건> WHERE <검색조건>	
외부조인	SELECT <속성들> FROM 테이블1 {LEFT RIGHT FULL [OUTER]} JOIN 테이블2 ON <조인조건> WHERE <검색조건>	외부조인은 FROM 절에 조인 종류를 적고 ON을 이용하여 조인조건을 명시한다.

3.3.2 부속질의 _SQL 문 내에 또 다른 SQL 문을 작성해보자

질의 3-28 가장 비싼 도서의 이름을 보이시오.

```
SELECT    bookname
FROM      Book
WHERE     price = ( SELECT MAX(price)
                   FROM Book;
```

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	축구 아는 여자	나무수	13000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별 기술	굿스포츠	6000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

가장 비싼
도서의 가격은
→ 35,000원

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	축구 아는 여자	나무수	13000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별 기술	굿스포츠	6000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

그림 3-18 부속질의의 실행 순서

3.3.2 부속질의 SQL 문 내에 또 다른 SQL 문을 작성해보자

질의 3-29 도서를 구매한 적이 있는 고객의 이름을 검색하시오.

```
SELECT    name
FROM      Customer
WHERE     custid IN (SELECT custid
                     FROM    Orders);
```

name
박지성
김연마
장미란
추신수

질의 3-30 대한미디어에서 출판한 도서를 구매한 고객의 이름을 보이시오.

```
SELECT    name
FROM      Customer
WHERE     custid IN (SELECT custid
                     FROM    Orders
                     WHERE   bookid IN (SELECT bookid
                                         FROM    Book
                                         WHERE   publisher='대한미디어'));
```

name
박지성

3.3.2 부속질의 _SQL 문 내에 또 다른 SQL 문을 작성해보자

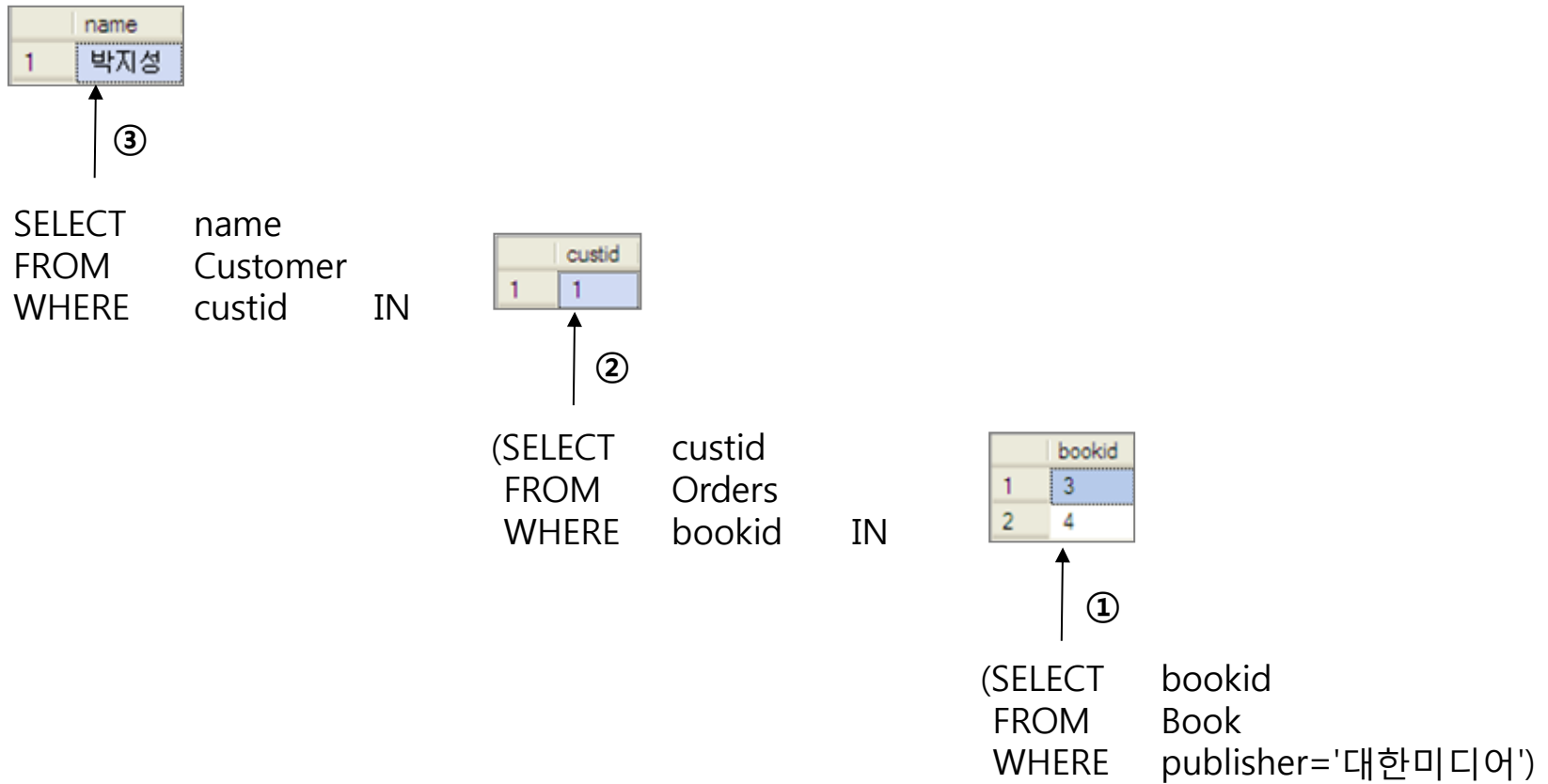


그림 3-19 3단계 부속질의 실행 순서

3.3.2 부속질의 SQL 문 내에 또 다른 SQL 문을 작성해보자

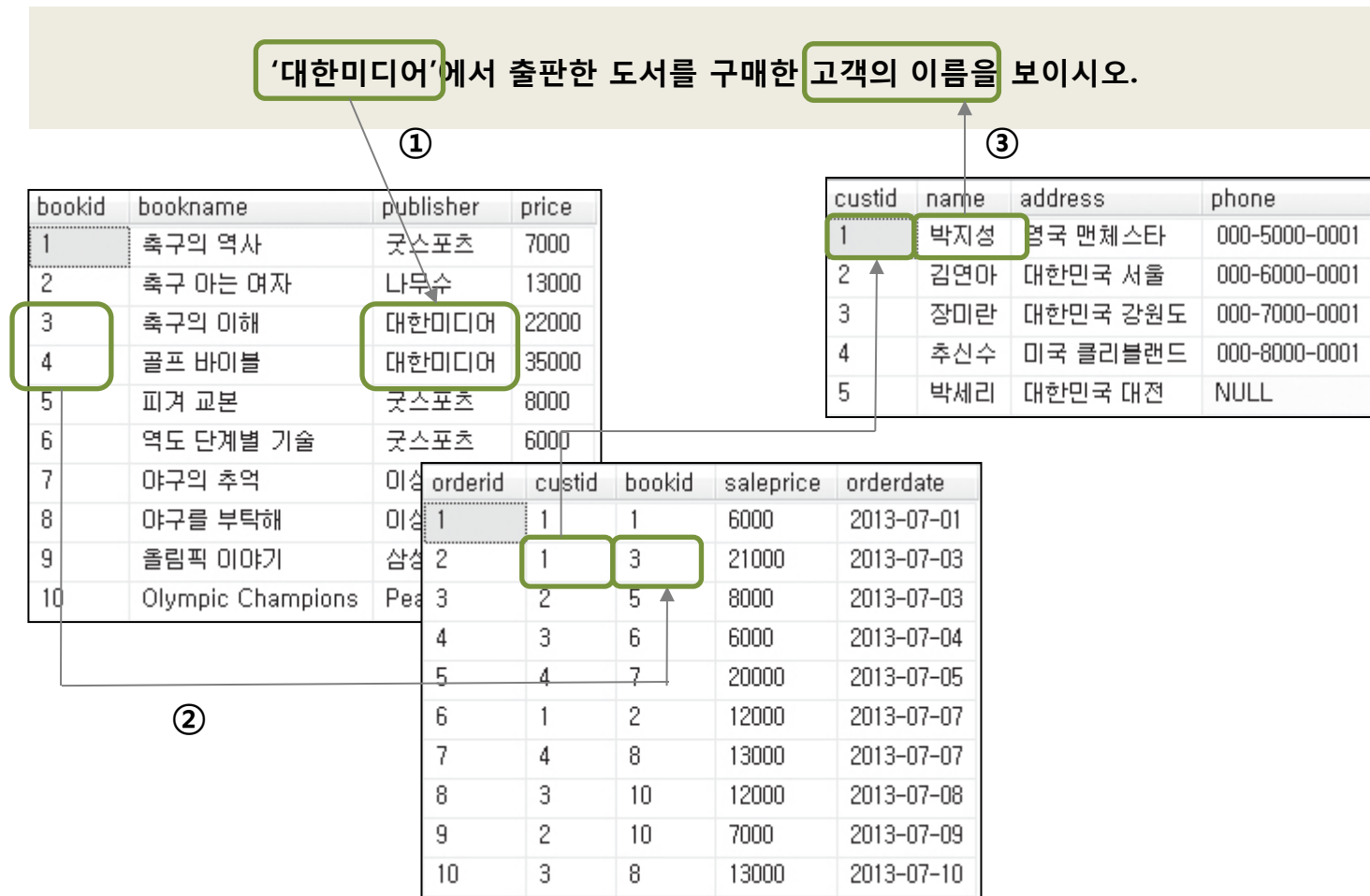


그림 3-20 3단계 부속질의 실행 순서와 데이터 예

3.3.2 부속질의 SQL 문 내에 또 다른 SQL 문을 작성해보자

- 상관 부속질의(correlated subquery)는 상위 부속질의의 튜플을 이용하여 하위 부속질의를 계산한다. 즉 상위 부속질의와 하위 부속질의가 독립적이지 않고 서로 관련을 맺고 있다.

질의 3-31 출판사별로 출판사의 평균 도서 가격보다 비싼 도서를 구하시오.

```
SELECT b1.bookname
FROM Book b1
WHERE b1.price > (SELECT avg(b2.price)
                  FROM Book b2
                  WHERE b2.publisher=b1.publisher);
```

bookname
피겨 교본
골프 바이블
야구의 추억

3.3.2 부속질의 SQL 문 내에 또 다른 SQL 문을 작성해보자

테이블 Book - b1으로 나타냄

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	축구 아는 여자	나무수	13000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별 기술	굿스포츠	6000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

B1 테이블의
튜플 1에 해당되
는 출판사를 b2
테이블로 가져
가서 해당되는
출판사 튜플들
의 price 값의
평균을 구한다.

테이블 Book - b2로 나타냄

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	축구 아는 여자	나무수	13000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별 기술	굿스포츠	6000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

그림 3-21 상관 부속질의 데이터 예

3.3.3 집합연산 _도서를 주문하지 않은 고객을 알고 싶다

■ 합집합 UNION, 차집합 EXCEPT, 교집합 INTERSECT

{도서를 주문하지 않은 고객} = {모든 고객} - {도서를 주문한 고객}

질의 3-32 도서를 주문하지 않은 고객의 이름을 보이시오.

```
SELECT  name
FROM    Customer
EXCEPT
SELECT  name
FROM    Customer
WHERE   custid IN (SELECT custid
                  FROM  Orders);
```

name
박세리

3.3.4 EXISTS_주문이 있는 고객을 알고 싶다

- EXISTS는 원래 단어에서 의미하는 것과 같이 조건에 맞는 튜플이 존재하면 결과에 포함시킨다. 즉 부속질의문의 어떤 행이 조건에 만족하면 참이다. 반면 NOT EXISTS는 부속질의문의 모든 행이 조건에 만족하지 않을 때만 참이다.

질의 3-33 주문이 있는 고객의 이름과 주소를 보이시오.

```
SELECT    name, address
FROM      Customer cs
WHERE     EXISTS (SELECT *
                  FROM Orders od
                  WHERE cs.custid =od.custid);
```

name	address
박지성	영국 맨체스터
김연아	대한민국 서울
장미란	대한민국 강원도
추신수	미국 클리블랜드

3.3.4 EXISTS_주문이 있는 고객을 알고 싶다

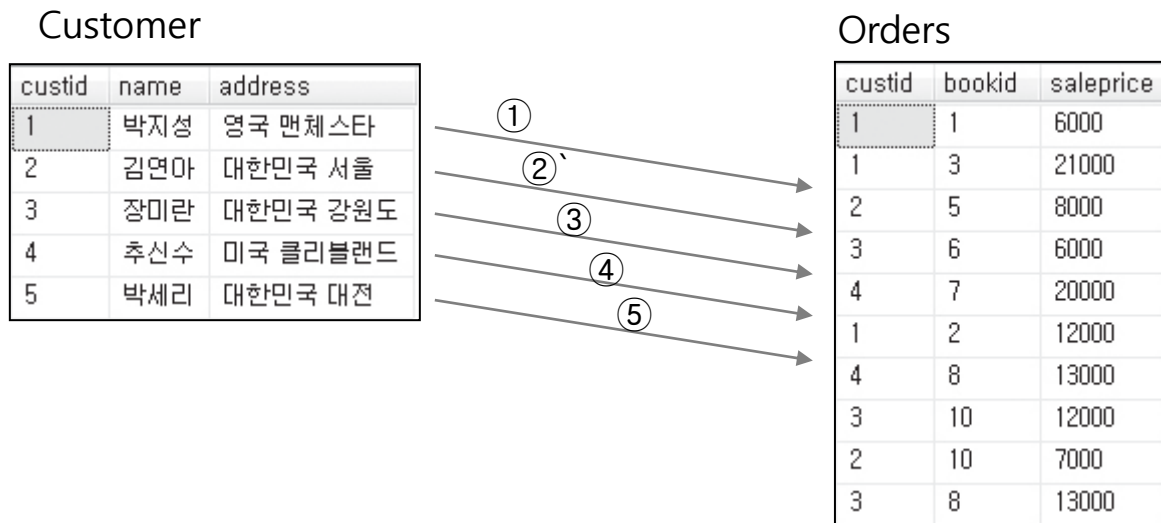


그림 3-24 EXIST 상관 부속질의문 데이터 예

04. 데이터 정의어

- CREATE 문
- ALTER 문
- DROP 문

4.1 CREATE 문

- CREATE 문은 테이블을 구성하고, 속성과 속성에 관한 제약을 정의하며, 기본키 및 외래키를 정의하는 명령이다. PRIMARY KEY는 기본키를 정할 때 사용하고 FOREIGN KEY는 외래키를 지정할 때 사용하며, ON UPDATE와 ON DELETE는 외래키 속성의 수정과 튜플 삭제시 동작을 나타낸다.

- CREATE 문의 기본 문법

```
CREATE TABLE 테이블이름
( { 속성이름 데이터타입
  [NOT NULL]
  [UNIQUE]
  [DEFAULT 기본값]
  [CHECK 체크조건]
}
  [PRIMARY KEY 속성이름(들)]
  {[FOREIGN KEY 속성이름 REFERENCES 테이블이름(속성이름)]
    [ON UPDATE [NO ACTION | CASCADE | SET NULL | SET DEFAULT]]
    [ON DELETE [NO ACTION | CASCADE | SET NULL | SET DEFAULT]]
  }
)
```

4.1 CREATE 문

질의 3-34 다음과 같은 속성을 가진 NewBook 테이블을 생성하시오, 정수형은 INT를 사용하며 문자형은 가변형 문자타입인 VARCHAR을 사용한다.

- bookid(도서번호) - INT
- bookname(도서이름) - VARCHAR(20)
- publisher(출판사) - VARCHAR(20)
- price(가격) - INT

```
CREATE TABLE NewBook (  
  bookid      INT,  
  bookname    VARCHAR(20),  
  publisher    VARCHAR(20),  
  price       INT);
```

※ 기본키를 지정하고 싶다면 다음과 같이 생성한다.

```
CREATE TABLE NewBook (  
  bookid      INT,  
  Bookname    VARCHAR(20),  
  publisher    VARCHAR(20),  
  price       INT PRIMARY KEY (bookid));
```

=

```
CREATE TABLE NewBook (  
  bookid      INT PRIMARY KEY,  
  bookname    VARCHAR(20),  
  publisher    VARCHAR(20),  
  price       INT);
```

4.1 CREATE 문

※ **bookid** 속성이 없어서 두 개의 속성 **bookname**, **publisher**가 기본키가 된다면 괄호를 사용하여 복합키를 지정한다.

```
CREATE TABLE NewBook (  
  bookname VARCHAR(20),  
  publisher VARCHAR(20),  
  price      INT  
  PRIMARY KEY (bookname, publisher));
```

※ **NewBook** 테이블의 **CREATE** 문에 좀 더 복잡한 제약사항을 추가한다.

bookname은 NULL 값을 가질 수 없고, publisher는 같은 값이 있으면 안 된다. price에 값이 입력되지 않을 경우 기본 값 10000을 저장한다. 또 가격은 최소 1,000원 이상으로 한다.

```
CREATE TABLE NewBook (  
  bookname  VARCHAR(20)      NOT NULL,  
  publisher  VARCHAR(20)      UNIQUE,  
  price      INT              DEFAULT 10000 CHECK(price > 1000),  
  PRIMARY KEY (bookname, publisher));
```


4.1 CREATE 문

질의 3-35 다음과 같은 속성을 가진 NewCustomer 테이블을 생성하시오.

- custid(고객번호) - INT, 기본키
- name(이름) - VARCHAR(40)
- address(주소) - VARCHAR(40)
- phone(전화번호) - VARCHAR(30)

```
CREATE TABLE NewCustomer (  
  custid      INT          PRIMARY KEY,  
  name        VARCHAR(40),  
  address     VARCHAR(40),  
  phone       VARCHAR(30));
```

4.1 CREATE 문

질의 3-36 다음과 같은 속성을 가진 **NewOrders** 테이블을 생성하시오.

- orderid(주문번호) - INT, 기본키
- custid(고객번호) - INT, NOT NULL 제약조건, 외래키(NewCustomer.custid, 연쇄삭제)
- bookid(도서번호) - INT, NOT NULL 제약조건
- saleprice(판매가격) - INT
- orderdate(판매일자) - DATE

```
CREATE TABLE NewOrders (  
orderid    INT,  
custid     INT          NOT NULL,  
bookid     INT          NOT NULL,  
saleprice  INT,  
orderdate  DATE,  
PRIMARY KEY (orderid),  
FOREIGN KEY (custid) REFERENCES NewCustomer(custid) ON DELETE CASCADE);
```

4.1 CREATE 문

- 외래키 제약조건을 명시할 때는 반드시 참조되는 테이블(부모 릴레이션)이 존재해야 하며 참조되는 테이블의 기본키여야 한다. 외래키 지정 시 ON DELETE 또는 ON UPDATE 옵션은 참조되는 테이블의 튜플이 삭제되거나 수정될 때 취할 수 있는 동작을 지정한다. NO ACTION은 어떠한 동작도 취하지 않고, SET NULL은 NULL 값으로 바꾸며, SET DEFAULT는 정해진 값으로 바꾼다.

표 3-7 속성의 데이터 타입 종류

데이터 타입	³¹ 설명	비슷한 타입
INT	정수형, 크기는 ± 2	BIGINT, SMALLINT
NUMERIC(p, s)	실수형 p자리 정수, s자리 소수	DECIMAL(p, s)
CHAR(n)	문자형 고정길이	
VARCHAR(n)	문자형 가변길이	
DATE	날짜형, 기본형은 YYYY-MM-DD	DATETIME, TIME

4.2 ALTER 문

- ALTER 문은 생성된 테이블의 속성과 속성에 관한 제약을 변경하며, 기본키 및 외래키를 변경한다. ADD, DROP은 속성을 추가하거나 제거할 때 사용한다. DEFAULT는 속성의 기본값을 설정하거나 삭제할 때 사용한다. 그리고 ADD <제약이름>, DROP <제약이름>은 제약사항을 추가하거나 삭제할 때 사용한다.
- ALTER 문의 기본 문법

ALTER TABLE 테이블이름

[ADD 속성이름 데이터타입]

[DROP COLUMN 속성이름]

[ALTER COLUMN 속성이름 데이터타입]

[ALTER COLUMN 속성이름 [NULL | NOT NULL]]

[ADD PRIMARY KEY(속성이름)]

[[ADD | DROP] 제약이름]

4.2 ALTER 문

질의 3-37 NewBook 테이블에 VARCHAR(13)의 자료형을 가진 isbn 속성을 추가하시오.

```
ALTER TABLE NewBook ADD isbn VARCHAR(13);
```

질의 3-38 NewBook 테이블의 isbn 속성의 데이터 타입을 INT형으로 변경하시오.

```
ALTER TABLE NewBook ALTER COLUMN isbn INT;
```

질의 3-39 NewBook 테이블의 isbn 속성을 삭제하시오.

```
ALTER TABLE NewBook DROP COLUMN isbn;
```

질의 3-40 NewBook 테이블의 bookid 속성에 NOT NULL 제약조건을 적용하시오.

```
ALTER TABLE NewBook ALTER COLUMN bookid INT NOT NULL;
```

질의 3-41 NewBook 테이블의 bookid 속성을 기본키로 변경하시오.

```
ALTER TABLE NewBook ADD PRIMARY KEY(bookid);
```

4.3 DROP 문

- DROP 문은 테이블을 삭제하는 명령이다. DROP 문은 테이블의 구조와 데이터를 모두 삭제하므로 사용에 주의해야 한다. (데이터만 삭제하려면 DELETE 문을 사용한다.)
- DROP문의 기본 문법

```
DROP TABLE 테이블이름
```

질의 3-42 NewBook 테이블의 bookid 속성을 기본키로 변경하시오.

```
DROP TABLE NewBook;
```

질의 3-43 NewCustomer 테이블을 삭제하시오. 만약 삭제가 거절된다면 원인을 파악하고 관련된 테이블을 같이 삭제하시오.

```
DROP TABLE NewCustomer;
```

05. 데이터 조작용어 - 삽입, 수정, 삭제

- INSERT 문
- UPDATE 문
- DELETE 문

5.1 INSERT 문

- INSERT 문은 테이블에 새로운 튜플을 삽입하는 명령이다.

- INSERT 문의 기본 문법

```
INSERT INTO 테이블이름[(속성리스트)]  
VALUES (값리스트);
```

질의 3-44 Book 테이블에 새로운 도서 '스포츠 의학'을 삽입하시오. 스포츠 의학은 한솔의학서적에서 출간했으며 가격은 90,000원이다.

```
INSERT INTO Book(bookid, bookname, publisher, price)  
VALUES (11, '스포츠 의학', '한솔의학서적', 90000);
```

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	2030 축구하는 여자	나무수	13000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별기술	굿스포츠	6000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000
11	스포츠 의학	한솔의학서적	90000

5.1 INSERT 문

질의 3-45 Book 테이블에 새로운 도서 '스포츠 의학'을 삽입하시오. 스포츠 의학은 한솔의학서적에서 출간했으며 가격은 미정이다.

```
INSERT INTO Book(bookid, bookname, publisher)
VALUES (12, '스포츠 의학', '한솔의학서적');
```

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	2030 축구마는 여자	나무수	13000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별기술	굿스포츠	6000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000
11	스포츠 의학	한솔의학서적	90000
12	스포츠 의학	한솔의학서적	NULL

5.1 INSERT 문

- 대량 삽입(bulk insert)이란 한꺼번에 여러 개의 튜플을 삽입하는 방법이다.

질의 3-46 수입도서 목록(Imported_book)을 Book 테이블에 모두 삽입하시오.

```
INSERT INTO Book(bookid, bookname, price, publisher)
SELECT bookid, bookname, price, publisher
FROM Imported_book;
```

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	2030 축구마는 여자	나무수	13000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별기술	굿스포츠	6000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000
11	스포츠 의학	한솔의학서적	90000
12	스포츠 의학	한솔의학서적	NULL
21	Zen Golf	Pearson	12000
22	Soccer Skills	Human Ki...	15000

5.2 UPDATE 문

- UPDATE 문은 특정 속성 값을 수정하는 명령이다.
- UPDATE 문의 기본 문법

```
UPDATE 테이블이름  
SET      속성이름1=값1[, 속성이름2=값2, ...]  
[WHERE <검색조건>];
```

5.2 UPDATE 문

질의 3-47 Customer 테이블에서 고객번호가 5인 고객의 주소를 '대한민국 부산'으로 변경하십시오.

```
UPDATE Customer
SET address='대한민국 부산'
WHERE custid=5;
```

custid	name	address	phone
1	박지성	영국 맨체스터	000-5000-0001
2	김연아	대한민국 서울	000-6000-0001
3	장미란	대한민국 강원도	000-7000-0001
4	추신수	미국 클리블랜드	000-8000-0001
5	박세리	대한민국 부산	NULL

질의 3-48 Customer 테이블에서 박세리 고객의 주소를 김연아 고객의 주소로 변경하십시오.

```
UPDATE Customer
SET address = (SELECT address
                FROM Customer
                WHERE name='김연아')
WHERE name='박세리';
```

custid	name	address	phone
1	박지성	영국 맨체스터	000-5000-0001
2	김연아	대한민국 서울	000-6000-0001
3	장미란	대한민국 강원도	000-7000-0001
4	추신수	미국 클리블랜드	000-8000-0001
5	박세리	대한민국 서울	NULL

5.3 DELETE 문

- DELETE 문은 테이블에 있는 기존 튜플을 삭제하는 명령이다.
- DELETE 문의 기본 문법

```
DELETE  
FROM   테이블이름  
[WHERE 검색조건];
```

질의 3-49 Customer 테이블에서 고객번호가 5인 고객을 삭제하시오.

```
DELETE  
FROM   Customer  
WHERE  custid=5;
```

custid	name	address	phone
1	박지성	영국 맨체스터	000-5000-0001
2	김연아	대한민국 서울	000-6000-0001
3	장미란	대한민국 강원도	000-7000-0001
4	추신수	미국 클리블랜드	000-8000-0001

질의 3-50 모든 고객을 삭제하시오.

```
DELETE  
FROM   Customer;
```

1. SQL Server 2012 익스프레스 에디션
2. SQL
3. 데이터 정의어(DDL)
4. 데이터 조작어(DML)
5. WHERE 조건
6. 집계 함수
7. GROUP BY
8. HAVING
9. 조인
10. 동등조인(내부조인)
11. 부속질의
12. 상관 부속질의
13. 튜플 변수
14. 집합연산
15. 집합연산
16. EXISTS
17. CREATE
18. ALTER
19. DROP
20. INSERT
21. UPDATE
22. DELETE

SQL Server로 배우는 데이터베이스 개론과 실습

