

마당서점의 데이터

Book 테이블

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

Orders 테이블

orderid	custid	bookid	saleprice	orderdate
1	1	1	6000	2020-07-01
2	1	3	21000	2020-07-03
3	2	5	8000	2020-07-03
4	3	6	6000	2020-07-04
5	4	7	20000	2020-07-05
6	1	2	12000	2020-07-07
7	4	8	13000	2020-07-07
8	3	10	12000	2020-07-08
9	2	10	7000	2020-07-09
10	3	8	13000	2020-07-10

Customer 테이블

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

릴레이션 용어	실무에서 많이 사용되는 용어	같은 의미의 파일 시스템 용어
릴레이션(relation)	테이블(table)	파일(file)
속성(attribute)	열(column)	필드(field)
튜플(tuple)	행(row)	레코드(record)

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	20/07/01

Customer(custid, name, address, phone)

CUSTID	NAME	ADDRESS	PHONE
1	박지성	영국 맨체스터	000-5000-0001

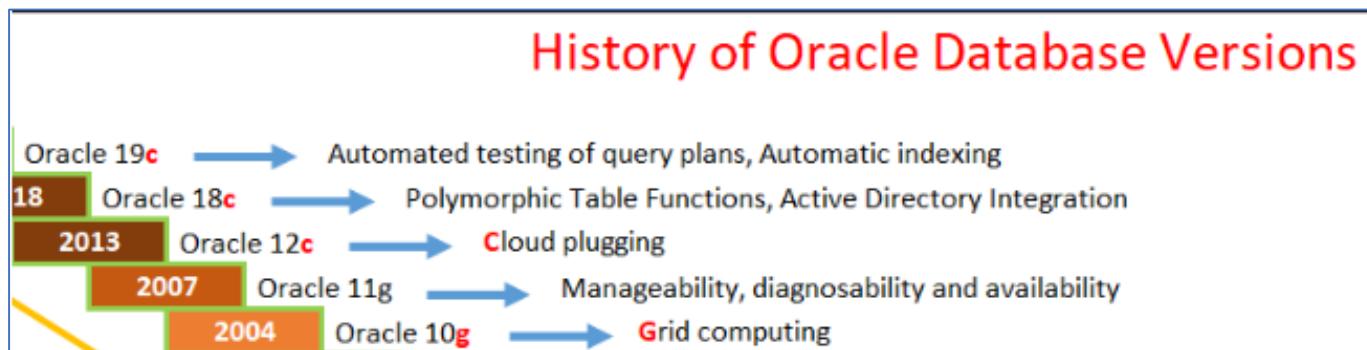
그림 3-3 마당서점의 데이터 구성도

오라클 설치와 기본 사용법

■ 오라클 18c(Express Edition) 설치(부록 A 참조)

- 오라클 DBMS를 내려받아 설치함 => <https://www.oracle.com/kr/database/technologies/xe-downloads.html> (C:\app\[사용자]\product\18.0.0\dbhomeXE 폴더)
- 시스템 관리자 계정 : system, 비밀번호 : Manager1 (비밀번호에 대문자와 숫자가 필요)
- [시작]-[Oracle-OraDB18_home1]-[응용 프로그램 개발] 메뉴에 포함된 SQL Plus 프로그램 사용 가능

오라클버전	
18c XE (Express Edition)	무료버전



- <여기서 잠깐> 오라클 18c 설치 소요시간
오라클 18c 버전은 설치 파일이 용량이 2G가 넘기 때문에 다운로드부터 설치까지 수분~수십분 이상이 소요
- <여기서 잠깐> 설치에 문제가 생기는 여러 원인들
컴퓨터 이름이 한글로 되어 있을 때
오라클을 한번 설치한 적이 있을 경우
- oracle home에서 ./deinstall/deinstall 실행
- regedit를 이용하여 기록을 삭제(방법 명령창에서 "sc delete OracleServiceXE"를 수행)

오라클 설치와 기본 사용법

■ 샘플 데이터베이스 설치(부록 B.3~B4 참조)

- madang 데이터베이스(본 교재에서 사용하는 데이터베이스)
- madang 사용자 계정 및 샘플 데이터베이스 설치 : B.3
- (스크립트를 실행한다 - [demo_madang.sql](#))

■ emp(employee) 데이터베이스(오라클에서 만들어 교육용으로 사용하는 데이터베이스)

- 사용자 계정은 scott : B.4
- 예제로 emp와 dept 테이블 포함
- (스크립트를 실행한다 - [demo_scott.sql](#))
- 이전 버전들에서는 scott 계정이 기본으로 설치되어있는 경우가 있다.
(계정 사용(해제) 명령 : ALTER USER c##scott ACCOUNT UNLOCK)

■ SQL Developer 설치(부록 B.1 참조)

- 오라클 홈페이지에서 본인의 환경에 맞는 버전 다운로드하여 설치

SQL Developer

■ SQL Developer 설치

- 오라클에서 내려받아 설치함 => <https://www.oracle.com/tools/downloads/sqldev-downloads.html>
- 임의의 폴더에 압축파일을 해제하고 실행은 폴더에 포함된 sqldeveloper.exe 파일을 실행시킨다.
(단축아이콘을 만들어두면 편리하다)

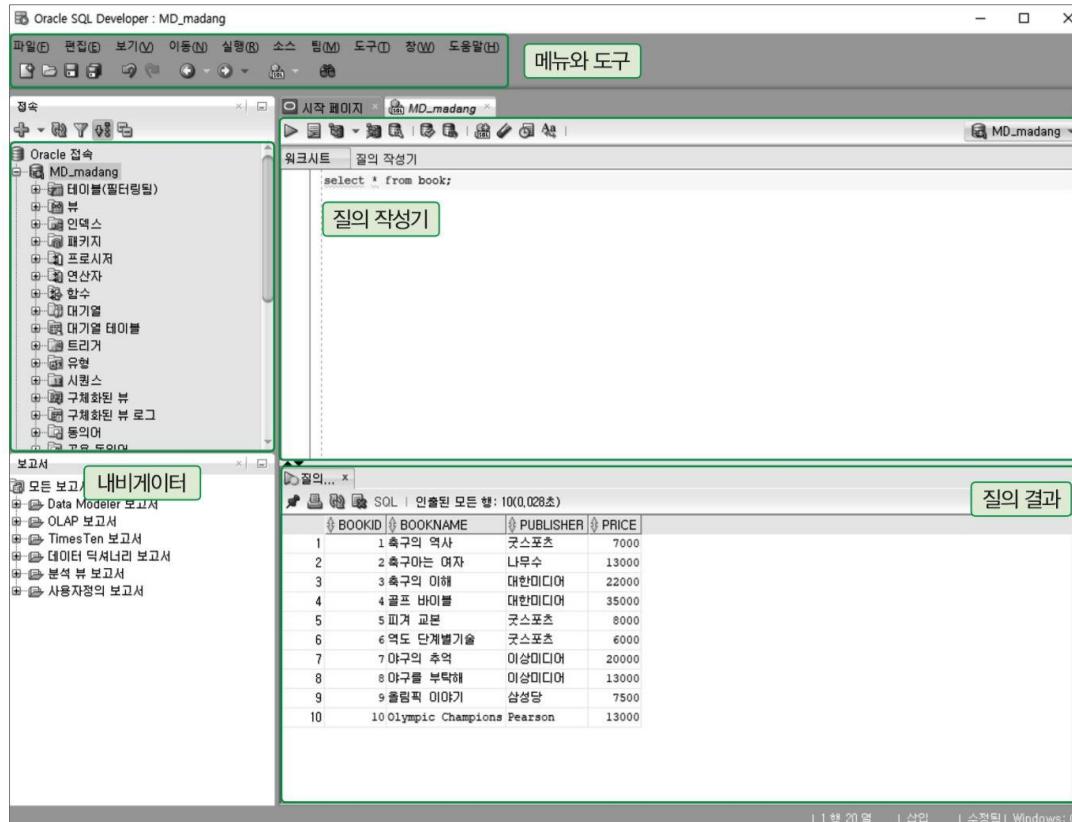


그림 3-6 SQL Developer에서 SQL 문을 실행한 화면

SQL Developer

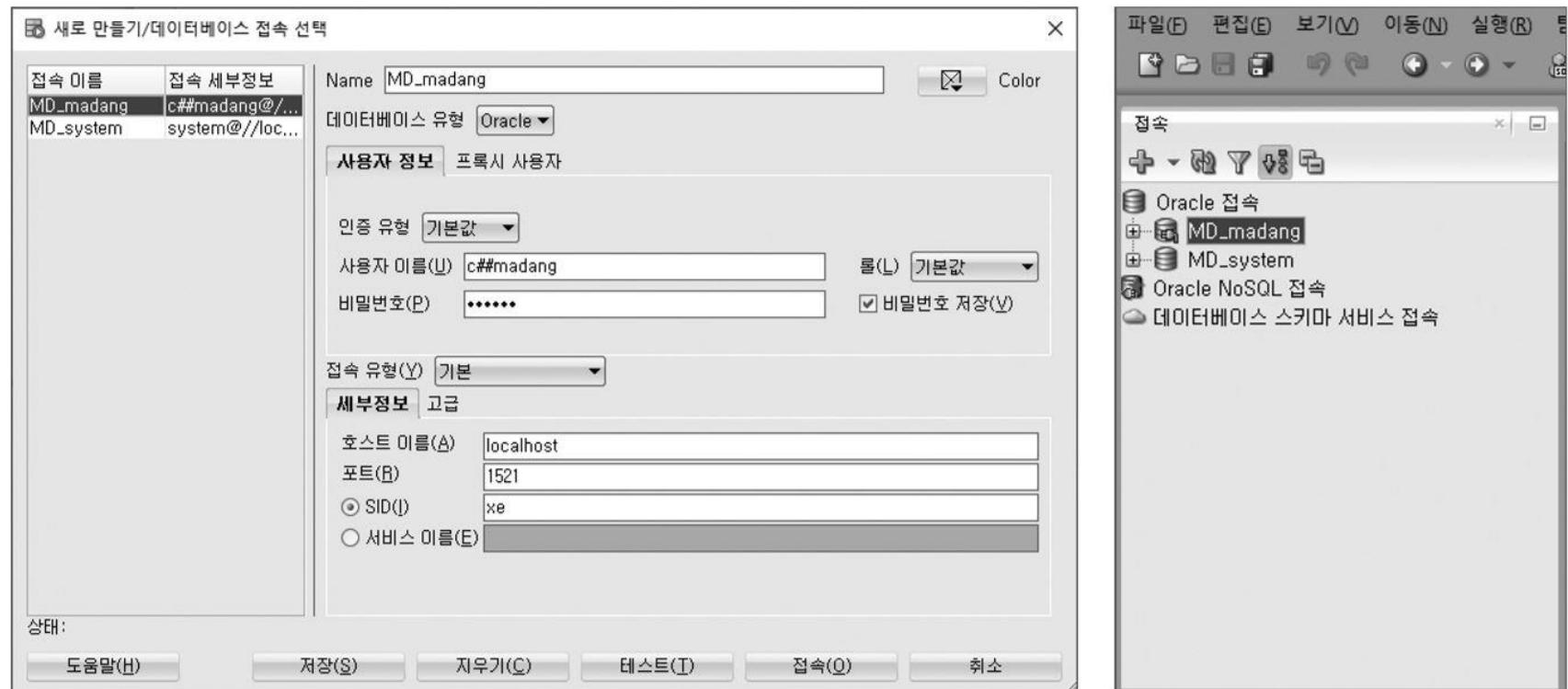


그림 3-7 접속 아이콘 생성 화면

SQL 소개

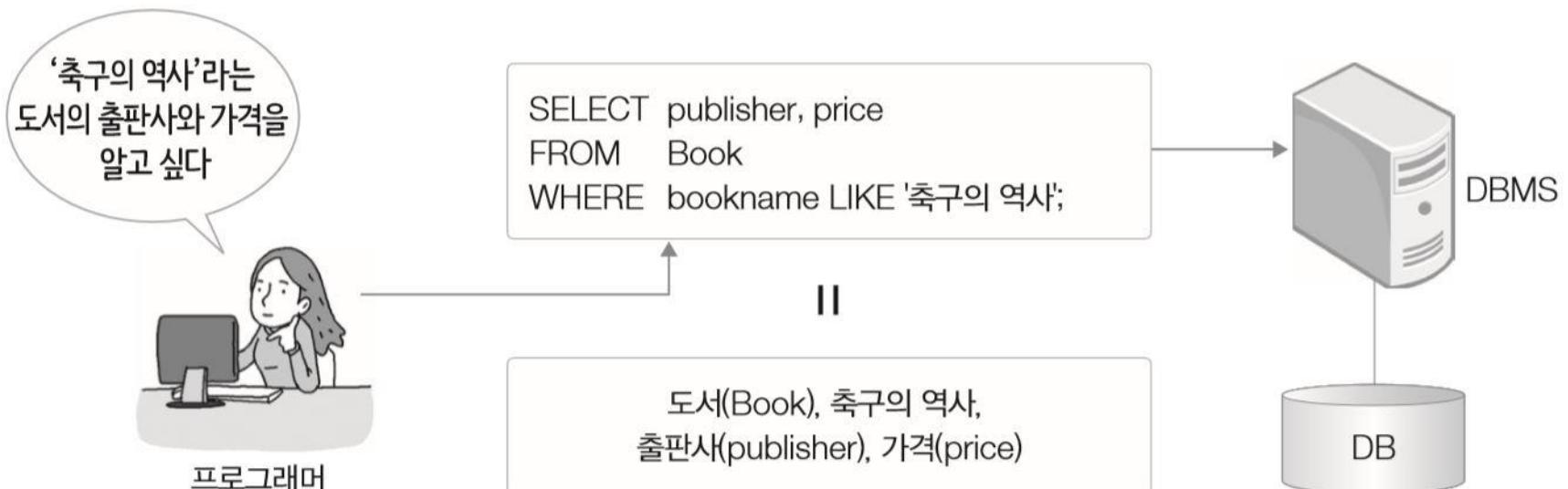


그림 3-8 SQL을 사용해 자료를 찾는 과정

SQL 소개

■ SQL 기능에 따른 분류

■ 데이터 정의어(**DDL**)

- 테이블이나 관계의 구조를 생성하는 데 사용
- CREATE, ALTER, DROP 문 등이 있음.

■ 데이터 조작어(**DML**)

- 테이블에 데이터를 검색, 삽입, 수정, 삭제하는 데 사용
- SELECT, INSERT, DELETE, UPDATE 문 등이 있음.
 - SELECT 문은 특별히 질의어(query)라고 함.

■ 데이터 제어어(**DCL**)

- 데이터의 사용 권한을 관리하는 데 사용
- GRANT, REVOKE 문 등이 있음.

SQL 소개

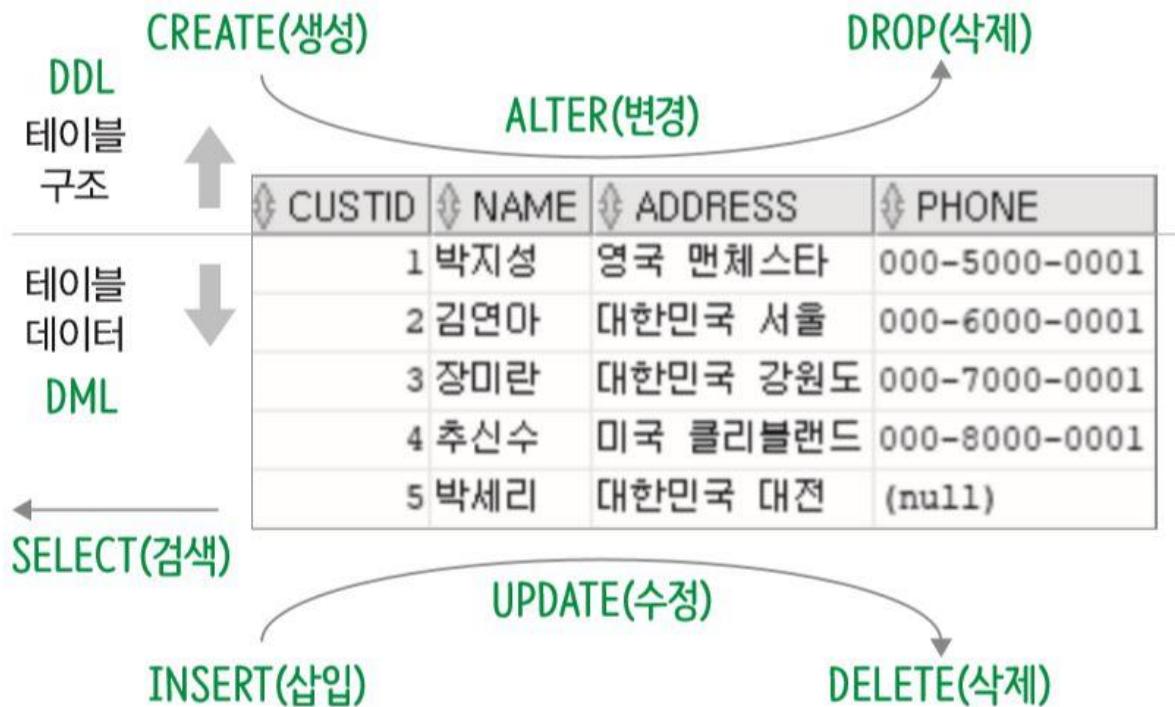


그림 3-9 데이터 정의어와 데이터 조작어의 주요 명령어

데이터 조작어 - 검색

■ SELECT 문의 구성 요소

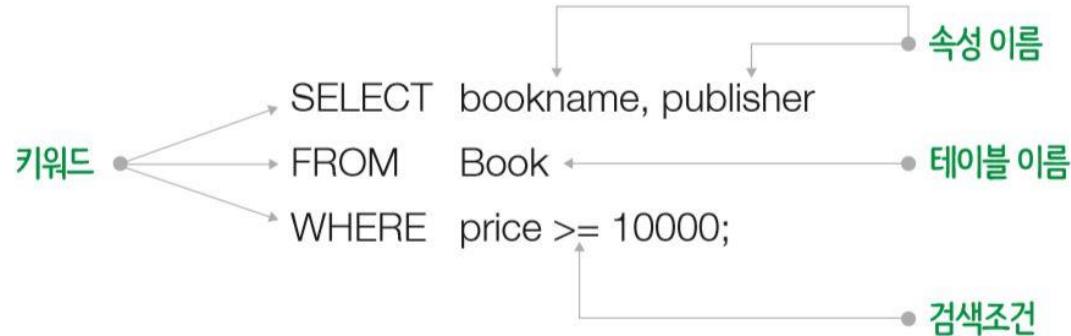


그림 3-11 SELECT 문의 예

■ 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.

04. 데이터 조작어 - 검색

■ SELECT 문의 기본 문법

```
SELECT
    [ALL | DISTINCT]
    [테이블이름.] { * | 속성이름 [[ AS ] 속성이름별칭] }
[FROM
    {테이블이름 [AS 테이블이름별칭]}]
    [INNER JOIN | LEFT [OUTER] JOIN | RIGHT [OUTER] JOIN
    {테이블이름 [ON 검색조건]}
    | FULL [OUTER] JOIN {테이블이름}]]
[WHERE 검색조건(들)]
[GROUP BY {속성이름, [..., n]}]
[HAVING 검색조건(들)]
[질의 UNION 질의 | 질의 UNION ALL 질의]
[ORDER BY {속성이름 [ASC | DESC], [..., n]}]
```

[]: 대괄호 안의 SQL 예약어들은 선택적으로 사용한다.

{ }: 중괄호 안의 SQL 예약어들은 필수적으로 사용한다.

|: 선택 가능한 문법 중 한 개를 사용할 수 있다.

SELECT 문 예제

질의 3-1 모든 도서의 이름과 가격을 검색하시오

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

BOOKNAME	PRICE
축구의 역사	7000
축구마는 여자	13000
축구의 이해	22000
골프 바이블	35000
피겨 교본	8000
역도 단계별기술	6000
야구의 추억	20000
야구를 부탁해	13000
올림픽 미야기	7500
Olympic Champions	13000

질의 3-1 변형 모든 도서의 이름과 가격을 검색하시오

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

PRICE	BOOKNAME
7000	축구의 역사
13000	축구마는 여자
22000	축구의 이해
35000	골프 바이블
8000	피겨 교본
6000	역도 단계별기술
20000	야구의 추억
13000	야구를 부탁해
7500	올림픽 미야기
13000	Olympic Champions

SELECT 문 예제

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

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

```
SELECT      *  
FROM        Book;
```

BOOKID	BOOKNAME	PUBLISHER	PRICE
1	축구의 역사	굿스포츠	7000
2	축구마는 여자	나무수	13000
3	축구의 미해	대한미디어	22000
4	골프 바이블	대한미디어	35000
5	피겨 교본	굿스포츠	8000
6	역도 단계별기술	굿스포츠	6000
7	야구의 추억	미상미디어	20000
8	야구를 부탁해	미상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

SELECT 문 예제

질의 3-3 도서 테이블에 있는 모든 출판사를 검색하시오.

```
SELECT publisher  
FROM Book;
```

PUBLISHER
굿스포츠
나무수
대한미디어
대한미디어
굿스포츠
굿스포츠
미상미디어
미상미디어
삼성당
Pearson

※ 중복을 제거하고 싶으면 DISTINCT 키워드를 사용

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

PUBLISHER
굿스포츠
삼성당
대한미디어
Pearson
나무수
미상미디어

조건 검색 WHERE

표 3-3 WHERE 절에 조건으로 사용할 수 있는 술어

술어	연산자	사용 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	축구마는 여자	나무수	13000
5	피겨 교본	굿스포츠	8000
6	역도 단계별기술	굿스포츠	6000
8	야구를 부탁해	이상미디어	13000
9	올림픽 미야기	삼성당	7500
10	Olympic Champions	Pearson	13000

조건 검색 WHERE

■ 비교

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	축구하는 여자	나무수	13000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
10	Olympic Champions	Pearson	13000

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

BOOKID	BOOKNAME	PUBLISHER	PRICE
2	축구하는 여자	나무수	13000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
10	Olympic Champions	Pearson	13000

조건 검색 WHERE

■ 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 도서를 검색

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

BOOKID	BOOKNAME	PUBLISHER	PRICE
2	축구마는 여자	나무수	13000
7	야구의 추억	이상미디어	20000
8	야구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

조건 검색 WHERE

■ 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
축구하는 여자	나무수
축구의 이해	대한미디어

조건 검색 WHERE

■ 패턴

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	축구마는 여자	나무수	13000
3	축구의 이해	대한미디어	22000
7	야구의 추억	미상미디어	20000
8	야구를 부탁해	미상미디어	13000

표 3-4 와일드 문자의 종류

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

조건 검색 WHERE

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

검색 결과의 정렬 ORDER BY

질의 3-12 도서를 이름순으로 검색하시오.

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

BOOKID	BOOKNAME	PUBLISHER	PRICE
10	Olympic Champions	Pearson	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
2	축구마는 여자	나무수	13000
1	축구의 역사	굿스포츠	7000
3	축구의 이해	대한미디어	22000
5	피겨 교본	굿스포츠	8000

검색 결과의 정렬 ORDER BY

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
10	Olympic Champions	Pearson	13000
8	야구를 부탁해	미상미디어	13000
2	축구마는 여자	나무수	13000
7	야구의 추억	미상미디어	20000
3	축구의 이해	대한미디어	22000
4	골프 바이블	대한미디어	35000

검색 결과의 정렬 ORDER BY

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

```
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	미상미디어	20000
10	Olympic Champions	Pearson	13000
2	축구마는 여자	나무수	13000
8	야구를 부탁해	미상미디어	13000
5	피겨 교본	굿스포츠	8000
9	올림픽 이야기	삼성당	7500
1	축구의 역사	굿스포츠	7000
6	역도 단계별기술	굿스포츠	6000

집계 함수

질의 3-15 고객이 주문한 도서의 총 판매액을 구하시오.

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

⌚ SUM(SALEPRICE)
118000

- 의미 있는 열 이름을 출력하고 싶으면 속성 이름의 별칭을 지정하는 AS 키워드를 사용하여 열 이름을 부여

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

⌚ 총매출
118000

집계 함수

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

집계 함수

질의 3-18 마당서점의 도서 판매 건수를 구하시오.

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

COUNT(*)
10

표 3-5 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)

GROUP BY 검색

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
4	2	33000
3	3	31000

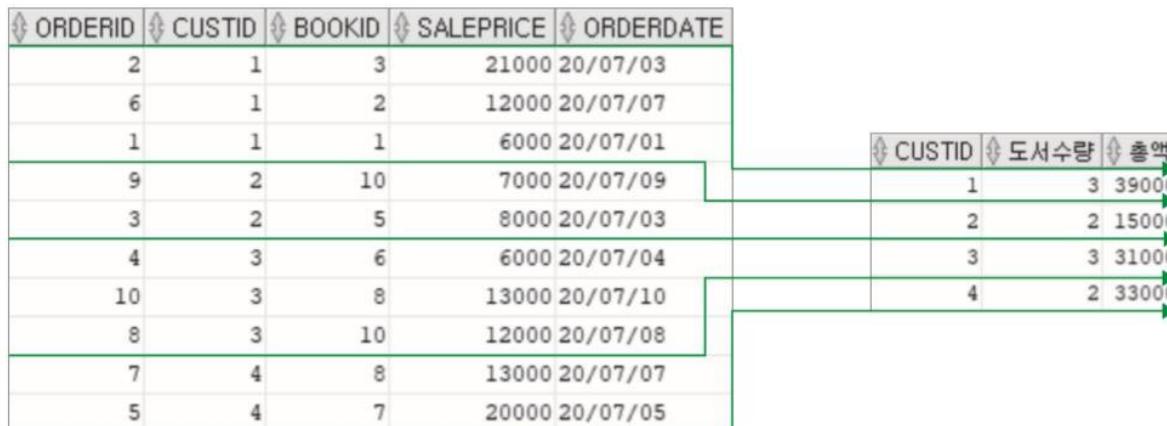


그림 3-12 GROUP BY 절의 수행

GROUP BY 검색

■ HAVING 절

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
4	2
3	2

GROUP BY 검색

■ GROUP BY와 HAVING 절의 문법과 주의사항

표 3-6 GROUP BY와 HAVING 절의 문법과 주의사항

GROUP BY <속성>

주의사항	GROUP BY로 투플을 그룹으로 묶은 후 SELECT 절에는 GROUP BY에서 사용한 <속성>과 같은 함수만 나올 수 있다.
맞는 예	<pre>SELECT custid, SUM(saleprice) FROM Orders GROUP BY custid;</pre>
틀린 예	<pre>SELECT bookid, SUM(saleprice) /* SELECT 절에 bookid 속성이 올 수 없다 */ FROM Orders GROUP BY custid;</pre>

HAVING <검색조건>

주의사항	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 집계 함수가 와야 한다.
맞는 예	<pre>SELECT custid, COUNT(*) AS 도서수량 FROM Orders WHERE saleprice >= 8000 GROUP BY custid HAVING count(*) >= 2;</pre>
틀린 예	<pre>SELECT custid, COUNT(*) AS 도서수량 FROM Orders HAVING count(*) >= 2 /* 순서가 틀렸다 */ WHERE saleprice >= 8000 GROUP BY custid;</pre>

연습문제

1. 마당서점의 고객이 요구하는 다음 질문에 대해 SQL 문을 작성하시오.

- (1) 도서번호가 1인 도서의 이름
- (2) 가격이 20,000원 이상인 도서의 이름
- (3) 박지성의 총 구매액(박지성의 고객번호는 1번으로 놓고 작성)
- (4) 박지성이 구매한 도서의 수(박지성의 고객번호는 1번으로 놓고 작성)

2. 마당서점의 운영자와 경영자가 요구하는 다음 질문에 대해 SQL 문을 작성하시오.

- (1) 마당서점 도서의 총 개수
- (2) 마당서점에 도서를 출고하는 출판사의 총 개수
- (3) 모든 고객의 이름, 주소
- (4) 2014년 7월 4일~7월 7일 사이에 주문 받은 도서의 주문번호
- (5) 2014년 7월 4일~7월 7일 사이에 주문 받은 도서를 제외한 도서의 주문번호
- (6) 성이 '김' 씨인 고객의 이름과 주소
- (7) 성이 '김' 씨이고 이름이 '아'로 끝나는 고객의 이름과 주소

두 개 이상 테이블에서 SQL 질의

■ Customer 테이블을 Orders 테이블과 조건 없이 연결

- Customer와 Orders 테이블의 합체 결과 튜플의 개수는 고객이 다섯 명이고 주문이 열 개이므로 5×10 해서 50이 됨

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

	CUSTID	NAME	ADDRESS	PHONE	ORDERID	CUSTID_1	BOOKID	SALEPRICE	ORDERDATE
1	1	박지성	영국 맨체스터	000-5000-0001	1	1	1	6000	20/07/01
2	1	박지성	영국 맨체스터	000-5000-0001	2	1	3	21000	20/07/03
3	1	박지성	영국 맨체스터	000-5000-0001	3	2	5	8000	20/07/03
4	1	박지성	영국 맨체스터	000-5000-0001	4	3	6	6000	20/07/04
5	1	박지성	영국 맨체스터	000-5000-0001	5	4	7	20000	20/07/05
6	1	박지성	영국 맨체스터	000-5000-0001	6	1	2	12000	20/07/07
7	1	박지성	영국 맨체스터	000-5000-0001	7	4	8	13000	20/07/07
8	1	박지성	영국 맨체스터	000-5000-0001	8	3	10	12000	20/07/08
9	1	박지성	영국 맨체스터	000-5000-0001	9	2	10	7000	20/07/09
10	1	박지성	영국 맨체스터	000-5000-0001	10	3	8	13000	20/07/10
11	2	김연아	대한민국 서울	000-6000-0001	1	1	1	6000	20/07/01
12	2	김연아	대한민국 서울	000-6000-0001	2	1	3	21000	20/07/03
13	2	김연아	대한민국 서울	000-6000-0001	3	2	5	8000	20/07/03
14	2	김연아	대한민국 서울	000-6000-0001	4	3	6	6000	20/07/04
15	2	김연아	대한민국 서울	000-6000-0001	5	4	7	20000	20/07/05
16	2	김연아	대한민국 서울	000-6000-0001	6	1	2	12000	20/07/07
17	2	김연아	대한민국 서울	000-6000-0001	7	4	8	13000	20/07/07
18	2	김연아	대한민국 서울	000-6000-0001	8	3	10	12000	20/07/08
19	2	김연아	대한민국 서울	000-6000-0001	9	2	10	7000	20/07/09
20	2	김연아	대한민국 서울	000-6000-0001	10	3	8	13000	20/07/10
... 연속됨 ...									
45	5	박세리	대한민국 대전	(null)	5	4	7	20000	20/07/05
46	5	박세리	대한민국 대전	(null)	6	1	2	12000	20/07/07
47	5	박세리	대한민국 대전	(null)	7	4	8	13000	20/07/07
48	5	박세리	대한민국 대전	(null)	8	3	10	12000	20/07/08
49	5	박세리	대한민국 대전	(null)	9	2	10	7000	20/07/09
50	5	박세리	대한민국 대전	(null)	10	3	8	13000	20/07/10

그림 3-13 Customer와 Orders 테이블의 합체

조인

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_1	BOOKID	SALEPRICE	ORDERDATE
1	박지성	영국 맨체스터	000-5000-0001	2	1	3	21000	20/07/03
1	박지성	영국 맨체스터	000-5000-0001	6	1	2	12000	20/07/07
1	박지성	영국 맨체스터	000-5000-0001	1	1	1	6000	20/07/01
2	김연아	대한민국 서울	000-6000-0001	9	2	10	7000	20/07/09
2	김연아	대한민국 서울	000-6000-0001	3	2	5	8000	20/07/03
3	장미란	대한민국 강원도	000-7000-0001	4	3	6	6000	20/07/04
3	장미란	대한민국 강원도	000-7000-0001	10	3	8	13000	20/07/10
3	장미란	대한민국 강원도	000-7000-0001	8	3	10	12000	20/07/08
4	추신수	미국 클리블랜드	000-8000-0001	7	4	8	13000	20/07/07
4	추신수	미국 클리블랜드	000-8000-0001	5	4	7	20000	20/07/05

조인

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

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

CUSTID	NAME	ADDRESS	PHONE	ORDERID	CUSTID_1	BOOKID	SALEPRICE	ORDERDATE
1	박지성	영국 맨체스터	000-5000-0001	2	1	3	21000	20/07/03
1	박지성	영국 맨체스터	000-5000-0001	6	1	2	12000	20/07/07
1	박지성	영국 맨체스터	000-5000-0001	1	1	1	6000	20/07/01
2	김연마	대한민국 서울	000-6000-0001	9	2	10	7000	20/07/09
2	김연마	대한민국 서울	000-6000-0001	3	2	5	8000	20/07/03
3	장미란	대한민국 강원도	000-7000-0001	4	3	6	6000	20/07/04
3	장미란	대한민국 강원도	000-7000-0001	10	3	8	13000	20/07/10
3	장미란	대한민국 강원도	000-7000-0001	8	3	10	12000	20/07/08
4	추신수	미국 클리블랜드	000-8000-0001	7	4	8	13000	20/07/07
4	추신수	미국 클리블랜드	000-8000-0001	5	4	7	20000	20/07/05

조인

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

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

NAME	SALEPRICE
박지성	21000
박지성	12000
박지성	6000
김연아	7000
김연아	8000
장미란	6000
장미란	13000
장미란	12000
추신수	13000
추신수	20000

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	SUM(SALEPRICE)
김연아	15000
박지성	39000
장미란	31000
추신수	33000

조인

다음과 같은 질의는 어떻게 구할 수 있는가?

- “고객의 이름과 고객이 주문한 도서의 이름을 구하시오”

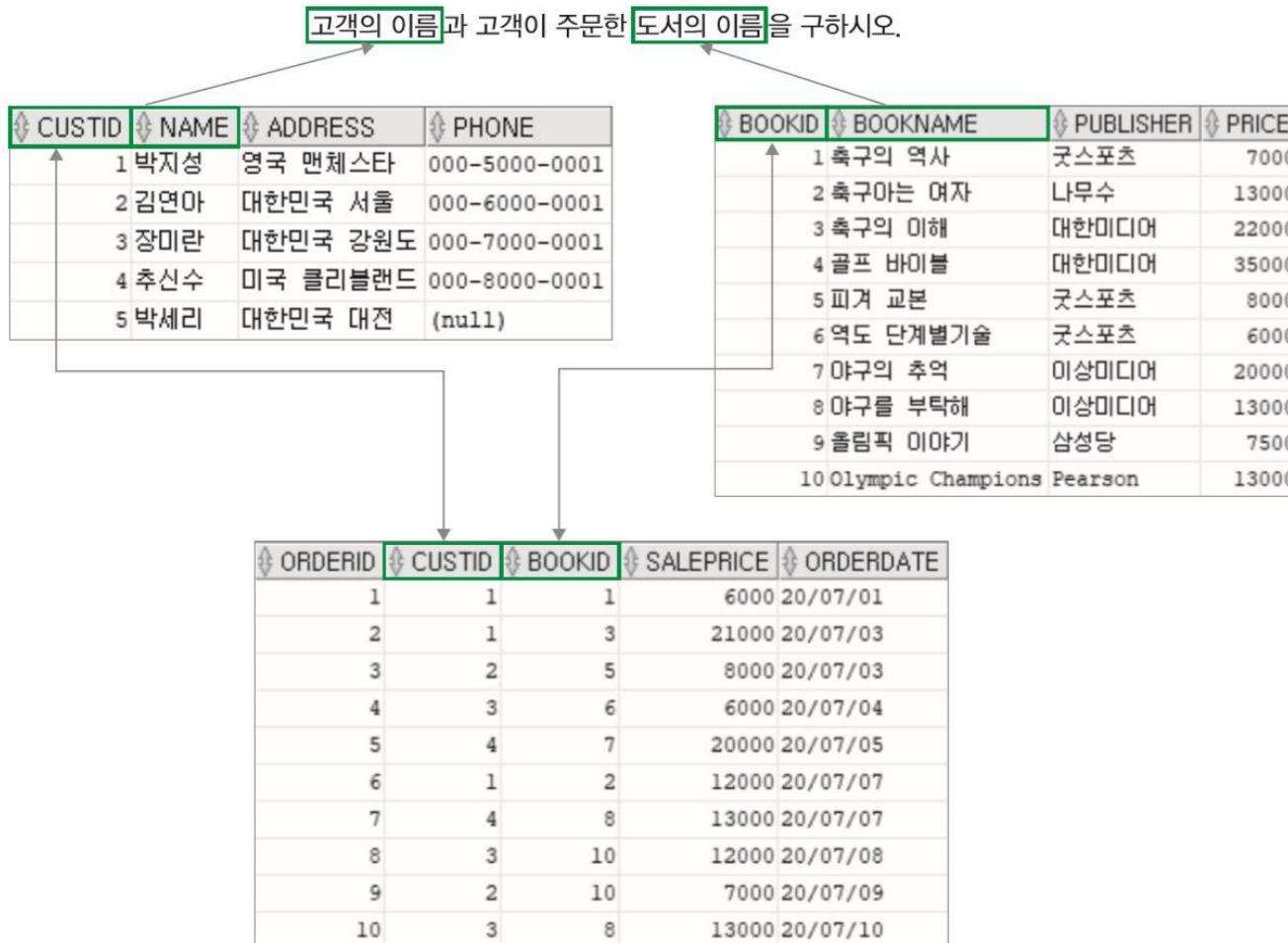


그림 3-14 마당서점 데이터 간의 연결

조인

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	축구마는 여자
박지성	축구의 미해
김연아	피겨 교본
장미란	역도 단계별기술
추신수	야구의 추억
장미란	야구를 부탁해
추신수	야구를 부탁해
김연아	Olympic Champions
장미란	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 Book. price=20000;
```

NAME	BOOKNAME
추신수	야구의 추억

조인

■ 외부조인

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

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

NAME	SALEPRICE
박지성	21000
박지성	12000
박지성	6000
김연아	7000
김연아	8000
장미란	6000
장미란	13000
장미란	12000
추신수	13000
추신수	20000
박세리	(null)

조인

표 3-7 조인 문법

명령	문법
내부조인	SELECT <속성들> FROM 테이블 1, 테이블 2 WHERE <조인조건> AND <검색조건>
	SELECT <속성들> FROM 테이블 1 INNER JOIN 테이블 2 ON <조인조건> WHERE <검색조건>
외부조인	SELECT <속성들> FROM 테이블 1 {LEFT RIGHT FULL [OUTER]} JOIN 테이블 2 ON <조인조건> WHERE <검색조건>

부속질의

질의 3-28 가장 비싼 도서의 이름을 보이시오.

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

BOOKNAME
골프 바이블

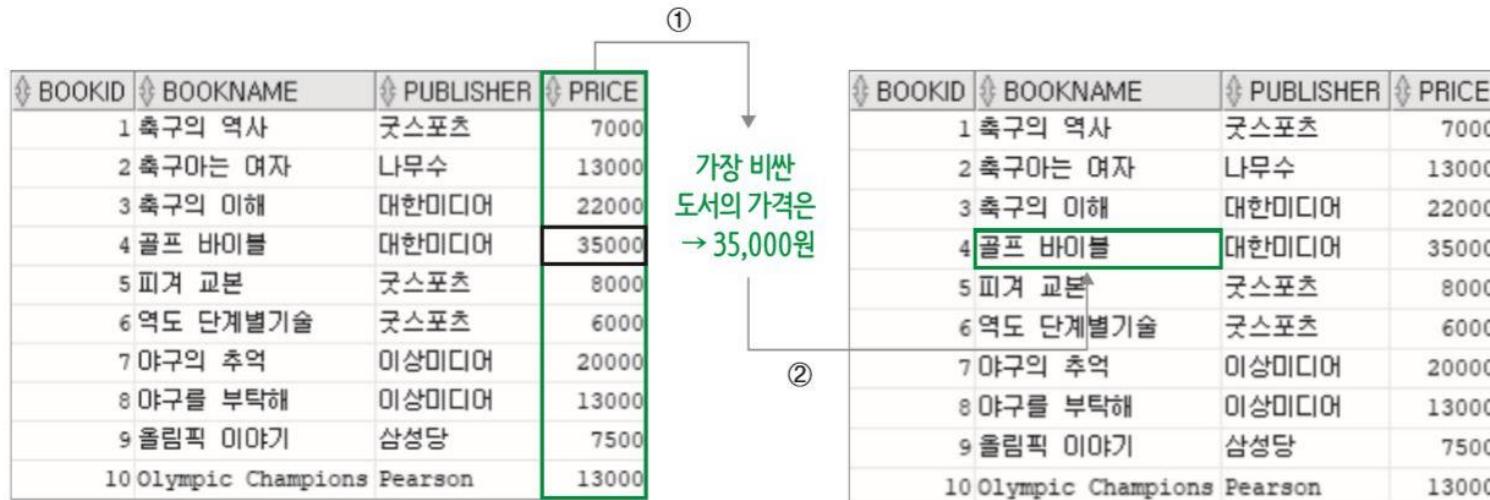


그림 3-15 부속질의의 실행 순서

부속질의

질의 3-29 도서를 구매한 적이 있는 고객의 이름을 검색하시오.

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

NAME
박지성
김연아
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

부속질의

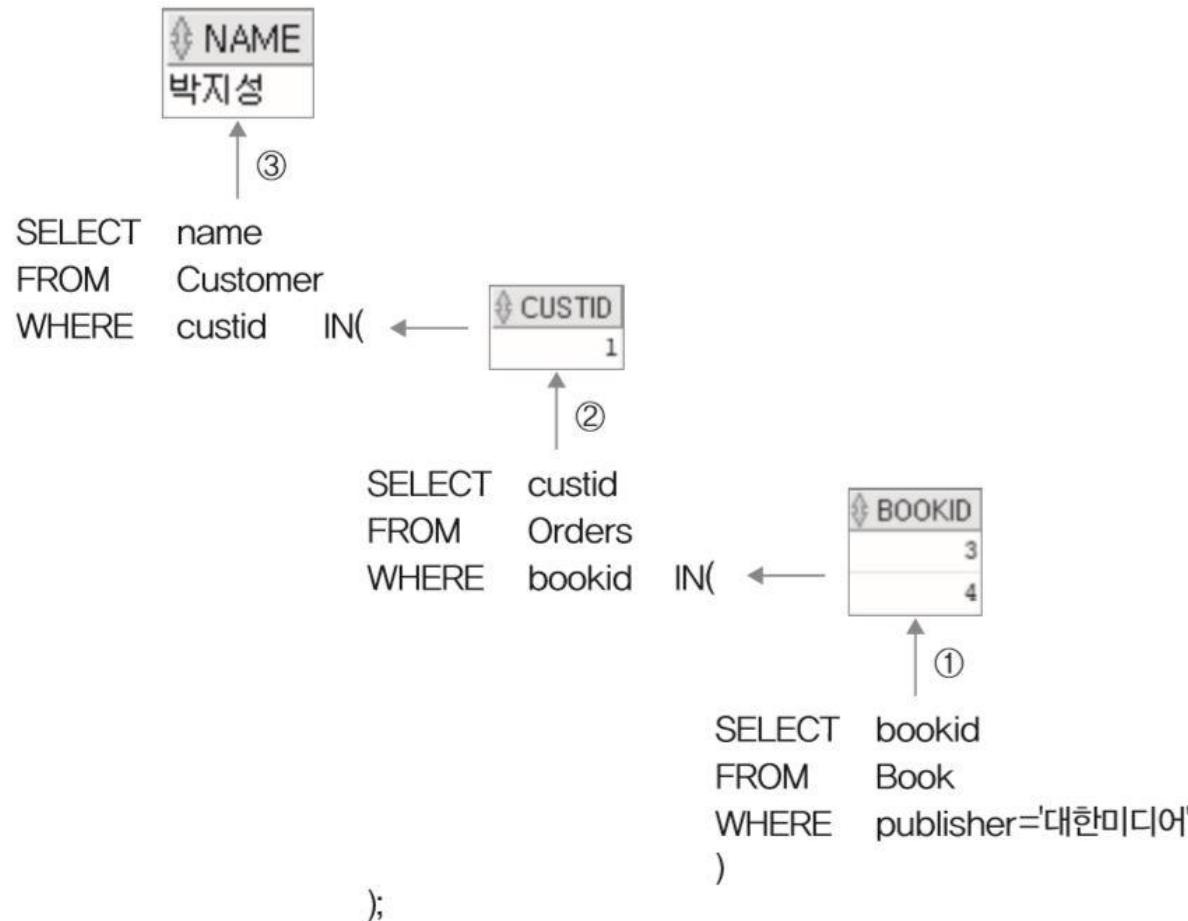


그림 3-16 3단계 부속질의의 실행 순서

부속질의

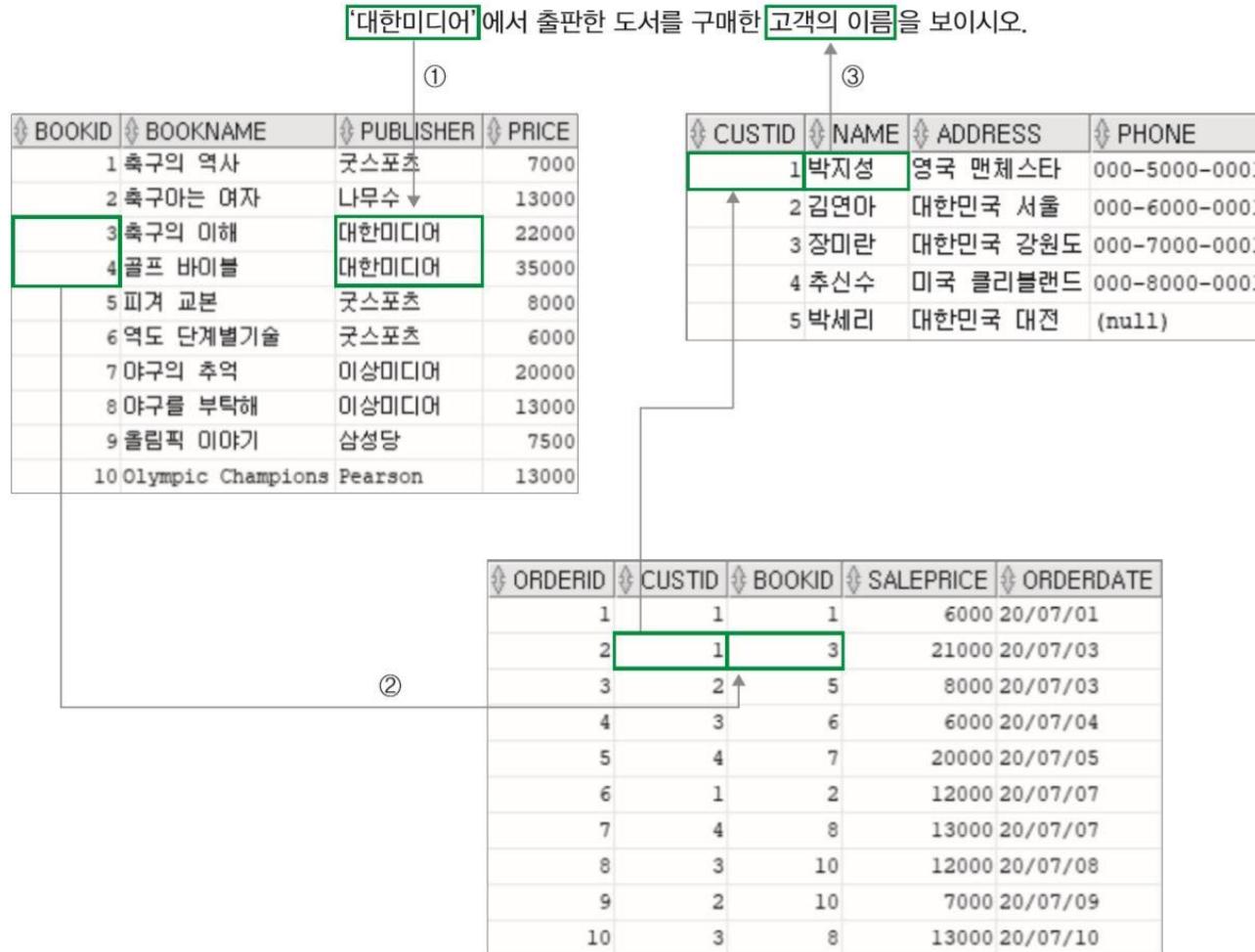


그림 3-17 3단계 부속질의의 실행 순서와 데이터 예

부속질의

■ 상관 부속질의(correlated subquery)

- 상위 부속질의의 튜플을 이용하여 하위 부속질의를 계산함.
- 상위 부속질의와 하위 부속질의가 독립적이지 않고 서로 관련을 맺고 있음.

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
골프 바이블
피겨 교본
마구의 추억

부속질의

Book 테이블: b1으로 나타냄

BOOKID	BOOKNAME	PUBLISHER	PRICE
1	축구의 역사	굿스포츠	7000
2	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

b1 테이블의
튜플 t에
해당하는
출판사를
b2 테이블로
가져가서,
같은 출판사를
가진 튜플들의
price 평균을
구한다.

Book 테이블: b2로 나타냄

BOOKID	BOOKNAME	PUBLISHER	PRICE
1	축구의 역사	굿스포츠	7000
2	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

$$avg(b2.price) \\ 28500$$

$$b1.price > avg(b2.price)$$

그림 3-18 상관 부속질의의 데이터 예

집합연산

■ 합집합 UNION, 차집합 MINUS, 교집합 INTERSECT

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

질의 3-32 도서를 주문하지 않은 고객의 이름을 보이시오.

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

NAME
박세리

- 주의할점: Oracle은 차집합을 **MINUS**로 하지만 SQL 표준에서는 **EXCEPT** 를 사용

EXISTS

■ EXISTS

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

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

EXISTS

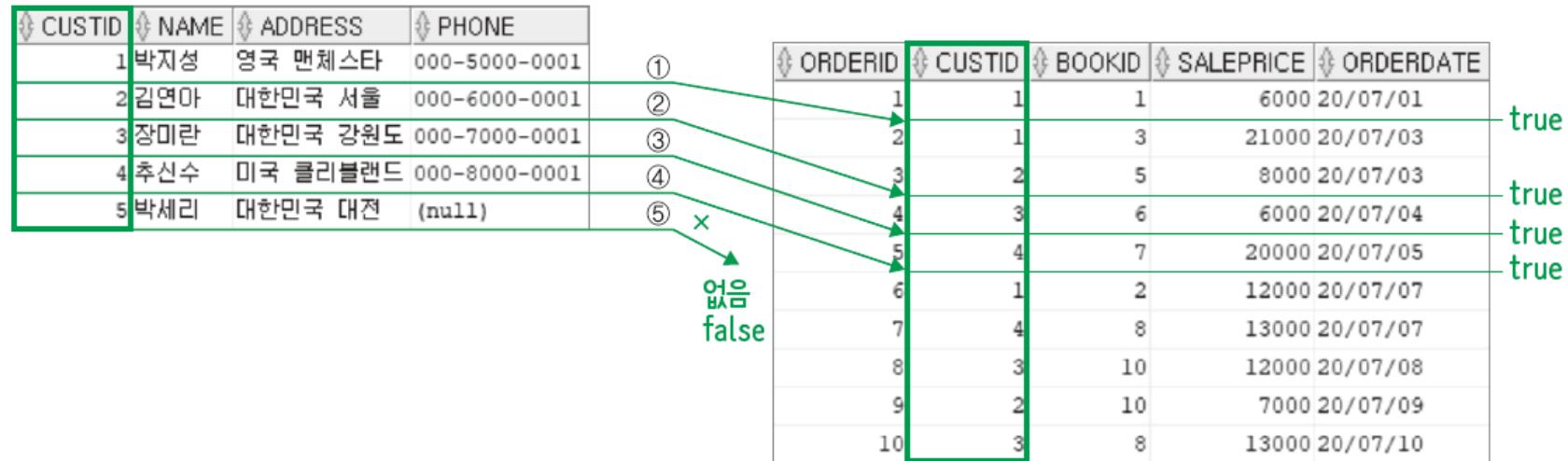


그림 3-21 EXIST 상관 부속질의문의 데이터 예

연습문제

1. 마당서점의 고객이 요구하는 다음 질문에 대해 SQL 문을 작성하시오.

- (5) 박지성이 구매한 도서의 출판사 수
- (6) 박지성이 구매한 도서의 이름, 가격, 정가와 판매가격의 차이
- (7) 박지성이 구매하지 않은 도서의 이름

2. 마당서점의 운영자와 경영자가 요구하는 다음 질문에 대해 SQL 문을 작성하시오.

- (8) 주문하지 않은 고객의 이름(부속질의 사용)
- (9) 주문 금액의 총액과 주문의 평균 금액
- (10) 고객의 이름과 고객별 구매액
- (11) 고객의 이름과 고객이 구매한 도서 목록
- (12) 도서의 가격(Book 테이블)과 판매가격(Orders 테이블)의 차이가 가장 많은 주문
- (13) 도서의 판매액 평균보다 자신의 구매액 평균이 더 높은 고객의 이름

CREATE TABLE 문

- 테이블을 구성하고, 속성과 속성에 관한 제약을 정의하며, 기본키 및 외래키를 정의하는 명령
- PRIMARY KEY는 기본키를 정할 때 사용하고 FOREIGN KEY는 외래키를 지정할 때 사용하며, ON UPDATE와 ON DELETE는 외래키 속성의 수정과 튜플 삭제 시 동작을 나타냄.
- CREATE TABLE 문의 기본 문법

```
CREATE TABLE 테이블이름
( {속성이름 데이터타입
    [NULL | NOT NULL | UNIQUE | DEFAULT 기본값 | CHECK 체크조건]
}
    [PRIMARY KEY 속성이름(들)]
    [FOREIGN KEY 속성이름 REFERENCES 테이블이름(속성이름)]
        [ON DELETE {CASCADE | SET NULL}]
)
```

CREATE TABLE 문

- 테이블을 구성하고, 속성과 속성에 관한 제약을 정의하며, 기본키 및 외래키를 정의하는 명령
- PRIMARY KEY는 기본키를 정할 때 사용하고 FOREIGN KEY는 외래키를 지정할 때 사용하며, ON UPDATE와 ON DELETE는 외래키 속성의 수정과 튜플 삭제 시 동작을 나타냄.
- CREATE TABLE 문의 기본 문법

```
CREATE TABLE 테이블이름
( {속성이름 데이터타입
    [NULL | NOT NULL | UNIQUE | DEFAULT 기본값 | CHECK 체크조건]
}
    [PRIMARY KEY 속성이름(들)]
    [FOREIGN KEY 속성이름 REFERENCES 테이블이름(속성이름)]
        [ON DELETE {CASCADE | SET NULL}]
)
```

CREATE TABLE 문

질의 3-34 다음과 같은 속성을 가진 NewBook 테이블을 생성하시오,
정수형은 NUMBER를, 문자형은 가변형 문자타입인 VARCHAR2를 사용

bookid(도서번호) – NUMBER
bookname(도서이름) – VARCHAR2(20)
publisher(출판사) – VARCHAR2(20)
price(가격) – NUMBER

```
CREATE TABLE      NewBook (
  bookid          NUMBER,
  bookname        VARCHAR2(20),
  publisher       VARCHAR2(20),
  price           NUMBER);
```

- 기본키를 지정하고 싶다면 다음과 같이 생성

```
CREATE TABLE      NewBook (
  bookid          NUMBER,
  bookname        VARCHAR2(20),
  publisher       VARCHAR2(20),
  price           NUMBER,
  PRIMARY KEY     (bookid));
```

CREATE TABLE 문

- bookid 속성이 없어서 두 개의 속성 bookname, publisher가 기본키가 된다면 괄호를 사용하여 복합키를 지정

```
CREATE TABLE      NewBook (
  bookname        VARCHAR2(20),
  publisher       VARCHAR2(20),
  price           NUMBER,
  PRIMARY KEY     (bookname, publisher));
```

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

- NewBook 테이블의 CREATE 문에 좀 더 복잡한 제약사항을 추가

```
CREATE TABLE      NewBook (
  bookname        VARCHAR2(20) NOT NULL,
  publisher       VARCHAR2(20) UNIQUE,
  price           NUMBER DEFAULT 10000 CHECK(price > 1000),
  PRIMARY KEY     (bookname, publisher));
```

CREATE TABLE 문

질의 3-35 다음과 같은 속성을 가진 NewCustomer 테이블을 생성하시오

custid(고객번호)	- NUMBER, 기본키
name(이름)	- VARCHAR2(40)
address(주소)	- VARCHAR2(40)
phone(전화번호)	- VARCHAR2(30)

```
CREATE TABLE      NewCustomer (
  custid        NUMBER          PRIMARY KEY,
  name          VARCHAR2(40),
  address       VARCHAR2(40),
  phone         VARCHAR2(30));
```

CREATE TABLE 문

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

orderid(주문번호) - NUMBER, 기본키

custid(고객번호) - NUMBER, NOT NULL 제약조건, 외래키(NewCustomer.custid, 연쇄삭제)

bookid(도서번호) - NUMBER, NOT NULL 제약조건

saleprice(판매가격) - NUMBER

orderdate(판매일자) - DATE

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

CREATE TABLE 문

- 외래키 제약조건을 명시할 때는 반드시 참조되는 테이블(부모 릴레이션)이 존재해야 함
- 참조되는 테이블의 기본키여야 함
- 외래키 지정 시 ON DELETE 또는 ON UPDATE 옵션은 참조되는 테이블의 튜플이 삭제되거나 수정될 때 취할 수 있는 동작을 지정
- NO ACTION은 어떠한 동작도 취하지 않음.

표 3-8 속성의 데이터 타입 종류

데이터 타입	설명	ANSI SQL 표준 타입
NUMBER(p, s)	실수형 p자리 정수부분, s자리 소수부분, p와 s를 생략하여 NUMBER라고 쓰면 NUMBER(8, 2)로 저장된다.	DECIMAL(p, s) NUMERIC[(p,s)] INTEGER, INT SMALLINT
CHAR(n)	문자형 고정길이, 문자를 저장하고 남은 공간은 공백으로 채운다.	CHARACTER(n) CHAR(n)
VARCHAR2(n)	문자형 가변길이, 4,000바이트까지 저장된다.	CHARACTER VARYING(n) CHAR VARYING(n)
DATE	날짜형, 연도, 월, 일, 시간을 저장한다.	

ALTER TABLE 문

- ALTER 문은 생성된 테이블의 속성과 속성에 관한 제약을 변경하며, 기본키 및 외래키를 변경함
- ADD, DROP은 속성을 추가하거나 제거할 때 사용
- MODIFY는 속성의 기본값을 설정하거나 삭제할 때 사용
- ADD <제약이름>, DROP <제약이름>은 제약사항을 추가하거나 삭제할 때 사용
- 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]
```

실습을 위하여 NewBook 테이블을 지우고 질의3-34 NewBook 테이블을 새로 생성한다.

```
CREATE TABLE      NewBook (
  bookid          NUMBER,
  bookname        VARCHAR2(20),
  publisher       VARCHAR2(20),
  price           NUMBER);
```

ALTER TABLE 문

질의 3-37 NewBook 테이블에 VARCHAR2(13)의 자료형을 가진 isbn 속성을 추가하시오

```
ALTER TABLE NewBook ADD isbn VARCHAR2(13);
```

질의 3-38 NewBook 테이블의 isbn 속성의 데이터 타입을 NUMBER형으로 변경하시오

```
ALTER TABLE NewBook MODIFY isbn NUMBER;
```

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 MODIFY bookid NUMBER NOT NULL;
```

질의 3-41 NewBook 테이블의 bookid 속성을 기본키로 변경하시오

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

DROP TABLE 문

■ DROP 문

- DROP 문은 테이블을 삭제하는 명령
- DROP 문은 테이블의 구조와 데이터를 모두 삭제하므로 사용에 주의해야 함
 - (데이터만 삭제하려면 DELETE 문을 사용)

■ DROP문의 기본 문법

```
DROP TABLE 테이블이름
```

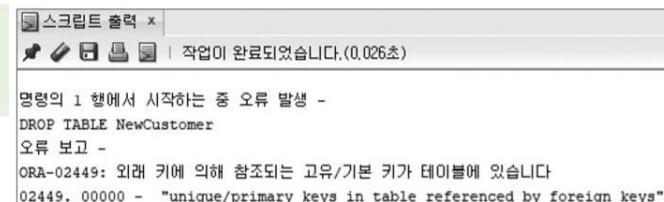
질의 3-42 NewBook 테이블을 삭제하시오

```
DROP TABLE NewBook;
```

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

(NewOrders 테이블이 NewCustomer를 참조하고 있음)

```
DROP TABLE NewCustomer;
```



INSERT 문

- INSERT 문은 테이블에 새로운 투플을 삽입하는 명령임.

- 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
11	스포츠 의학	한솔의학서적	90000
1	축구의 역사	굿스포츠	7000
2	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 주역	미상미디어	20000
8	야구를 부탁해	미상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

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 (14, '스포츠 의학', '한솔의학서적');
```

BOOKID	BOOKNAME	PUBLISHER	PRICE
11	스포츠 의학	한솔의학서적	90000
14	스포츠 의학	한솔의학서적	(null)
1	축구의 역사	굿스포츠	7000
2	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	미상미디어	20000
8	야구를 부탁해	미상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

INSERT 문

- 대량 삽입(bulk insert)이란 한꺼번에 여러 개의 투플을 삽입하는 방법임.

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
11	스포츠 의학	한솔의학서적	90000
14	스포츠 의학	한솔의학서적	(null)
21	Zen Golf	Pearson	12000
22	Soccer Skills	Human Kinetics	15000
1	축구의 역사	굿스포츠	7000
2	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	미상미디어	20000
8	야구를 부탁해	미상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

UPDATE 문

- UPDATE 문은 특정 속성 값을 수정하는 명령
- UPDATE 문의 기본 문법

```
UPDATE    테이블이름  
SET        속성이름 1=값 1[, 속성이름 2=값 2, ...]  
[WHERE    <검색조건>];
```

UPDATE 문

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

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

TIP 결과를 확인하기 위해서는 'SELECT * FROM Customer;' 명령을 실행해야 한다.

CUSTID	NAME	ADDRESS	PHONE
1	박지성	영국 맨체스터	000-5000-0001
2	김연마	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 변경하시오.

```
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	김연마	대한민국 서울	000-6000-0001
3	장미란	대한민국 강원도	000-7000-0001
4	추신수	미국 클리블랜드	000-8000-0001
5	박세리	대한민국 서울	(null)

DELETE 문

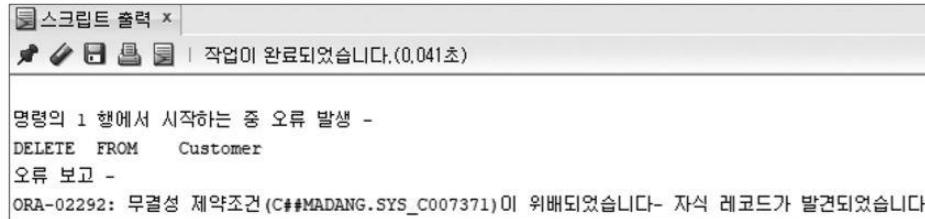
- DELETE 문은 테이블에 있는 기존 투플을 삭제하는 명령
- DELETE 문의 기본 문법

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

DELETE 문

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

```
DELETE FROM Customer;
```



The screenshot shows the 'Script Output' window of Oracle SQL Developer. It displays the command 'DELETE FROM Customer;' followed by an error message: 'ORA-02292: 무결성 제약조건 (C##MADANG.SYS_C007371)이 위배되었습니다- 자식 레코드가 발견되었습니다'. This indicates that a foreign key constraint is being violated because there are dependent rows in another table.

```
스크립트 출력 x
작업이 완료되었습니다.(0.041초)

명령의 1 행에서 시작하는 중 오류 발생 -
DELETE FROM Customer
오류 보고 -
ORA-02292: 무결성 제약조건 (C##MADANG.SYS_C007371)이 위배되었습니다- 자식 레코드가 발견되었습니다
```

질의 3-50 모든 고객을 삭제하시오.

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

```
SELECT * FROM Customer;
```

CUSTID	NAME	ADDRESS	PHONE
1	박지성	영국 맨체스터	000-5000-0001
2	김연마	대한민국 서울	000-6000-0001
3	장미란	대한민국 강원도	000-7000-0001
4	추신수	미국 클리블랜드	000-8000-0001

연습문제

1. 마당서점에서 다음의 심화된 질문에 대해 SQL 문을 작성하시오.

- (1) 박지성이 구매한 도서의 출판사와 같은 출판사에서 도서를 구매한 고객의 이름
- (2) 두 개 이상의 서로 다른 출판사에서 도서를 구매한 고객의 이름
- (3) (생략) 전체 고객의 30% 이상이 구매한 도서

2. 다음 질의에 대해 DML 문을 작성하시오.

- (1) 새로운 도서 ('스포츠 세계', '대한미디어', 10000원)이 마당서점에 입고되었다.
삽입이 안 될 경우 필요한 데이터가 더 있는지 찾아보자.
- (2) '삼성당'에서 출판한 도서를 삭제해야 한다.
- (3) '이상미디어'에서 출판한 도서를 삭제해야 한다. 삭제가 안 될 경우 원인을 생각해보자.
- (4) 출판사 '대한미디어'가 '대한출판사'로 이름을 바꾸었다.