숭실대학교 Soongsil University

Ch3. SQL 기초

숭실대학교 스마트시스템소프트웨어 학과
Department of Smart Systems Software

목차

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

1.1 마당서점의 데이터

bookid	bookname	publisher	price
1	축구의 역사	굿스포츠	7000
2	축구아는 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

Book 테이블

orderid	custid	bookid	saleprice	orderdate
1	1	1	6000	2014-07-01
2	1	3	21000	2014-07-03
3	2	5	8000	2014-07-03
4	3	6	6000	2014-07-04
5	4	7	20000	2014-07-05
6	1	2	12000	2014-07-07
7	4	8	13000	2014-07-07
8	3	10	12000	2014-07-08
9	2	10	7000	2014-07-09
10	3	8	13000	2014-07-10

Orders 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

Customer 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	14/07/01

Customer(custid, name, address, phone)

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

그림 3-6 마당서점의 데이터 구성도

1.2 누가 어떤 정보를 원하는가?

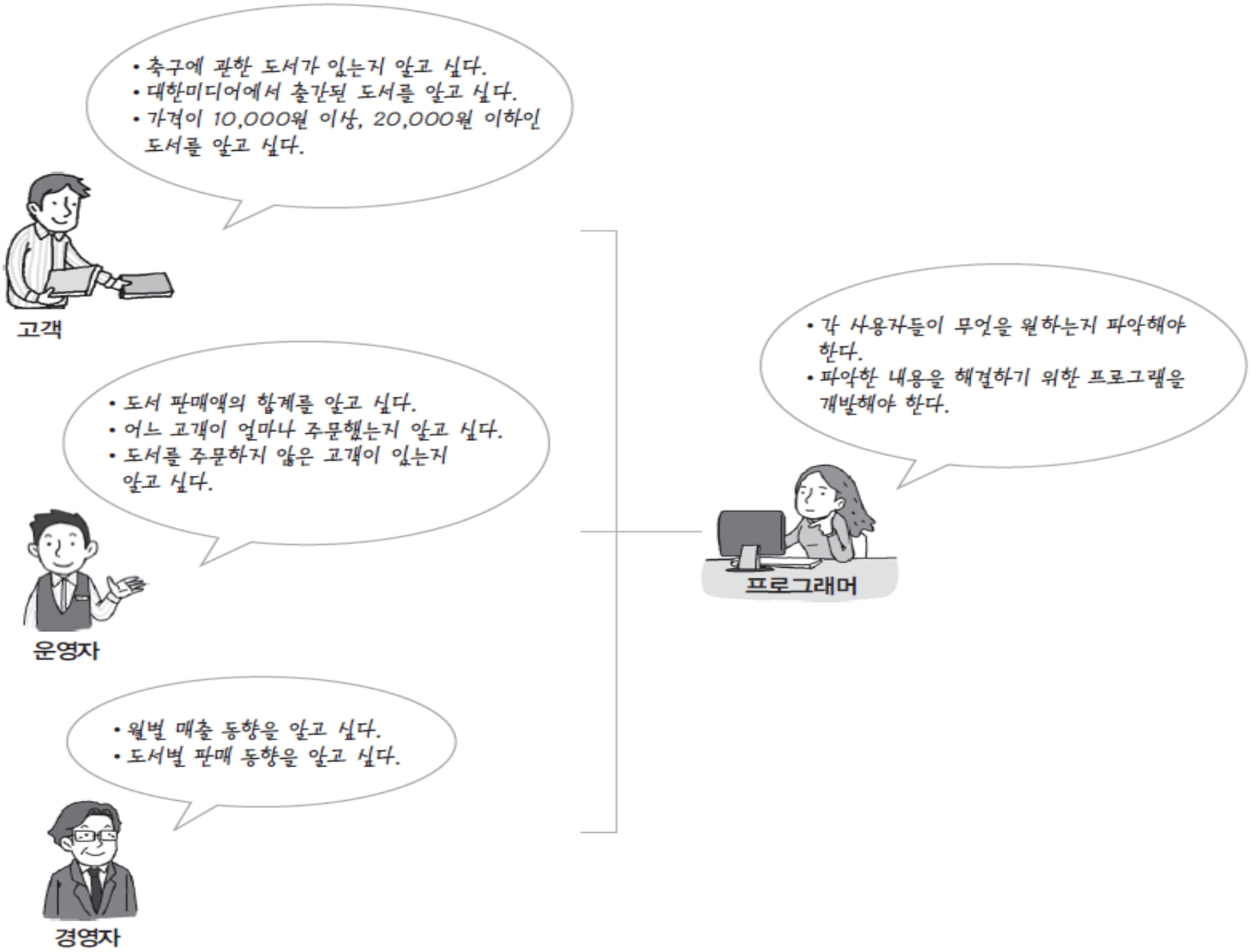


그림 3-7 사용자 그룹별로 원하는 정보

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() {...}

02. SQL 개요

SQL 기능에 따른 분류

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

02. SQL 개요

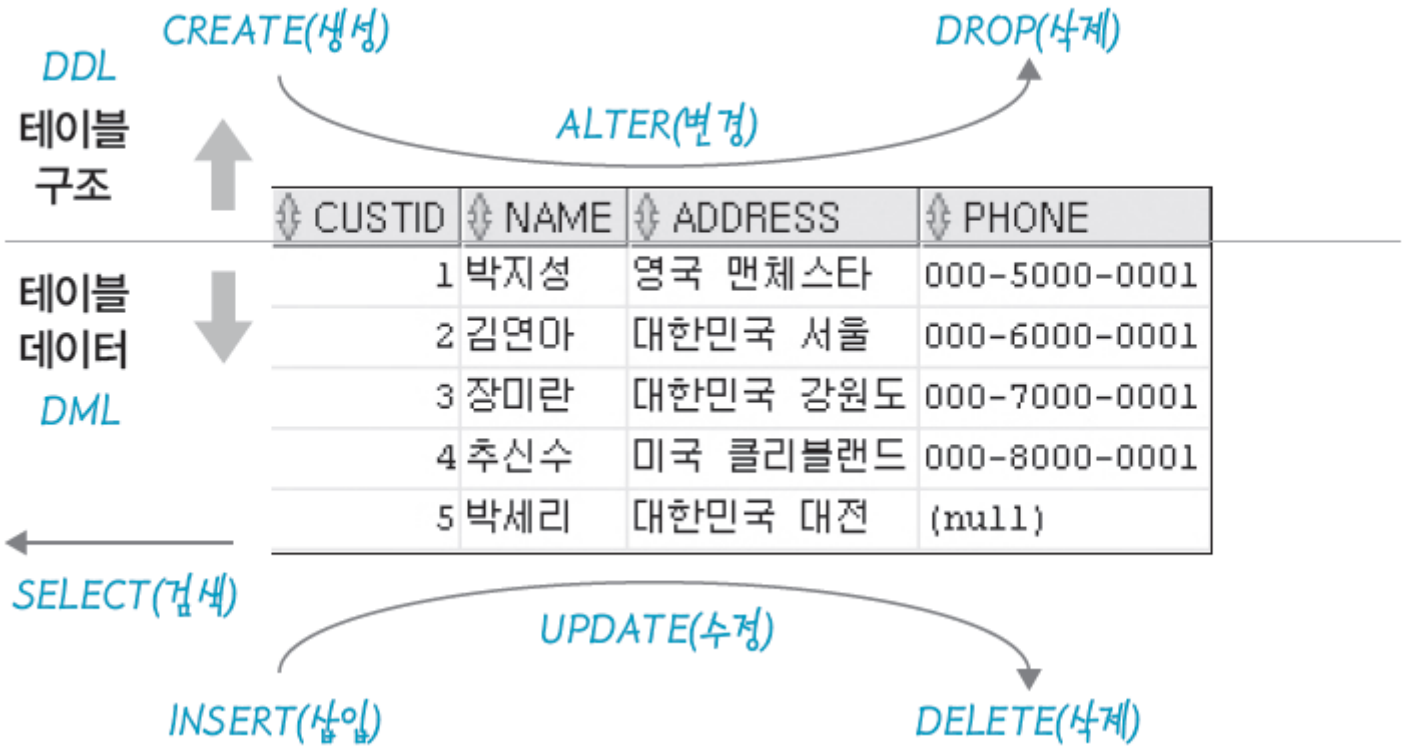


그림 3-12 데이터 정의어와 데이터 조작어의 주요 명령어

02. SQL 개요

예) 김연아 고객의 전화번호를 찾으시오.

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

① FROM Customer

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)

② WHERE name='김연아'

CUSTID	NAME	ADDRESS	PHONE
2	김연아	대한민국 서울	000-6000-0001

③ SELECT phone

PHONE
000-6000-0001

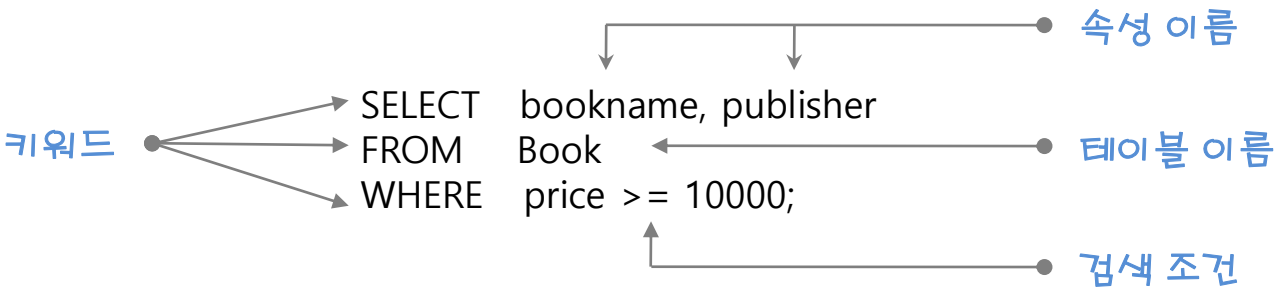
그림 3-13 SQL 문의 내부적 실행 순서

03. 데이터 조작어 - 검색

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

03. 데이터 조작어 - 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
축구아는 여자	13000
축구의 이해	22000
골프 바이블	35000
피겨 교본	8000
역도 단계별기술	6000
야구의 추억	20000
야구를 부탁해	13000
올림픽 이야기	7500
Olympic Champions	13000

(질의 3-1 변형) 모든 도서의 가격과 이름을 검색하시오.

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

PRICE	BOOKNAME
7000	축구의 역사
13000	축구아는 여자
22000	축구의 이해
35000	골프 바이블
8000	피겨 교본
6000	역도 단계별기술
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	축구아는 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

```
SELECT    *
FROM      Book;
```

BOOKID	BOOKNAME	PUBLISHER	PRICE
1	축구의 역사	굿스포츠	7000
2	축구아는 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

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
이상미디어
나무수
대한미디어
Pearson
삼성당

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	축구아는 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	축구아는 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	축구아는 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 도서를 검색하시오.

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

BOOKID	BOOKNAME	PUBLISHER	PRICE
2	축구아는 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
축구아는 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	축구아는 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
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	축구아는 여자	나무수	13000
1	축구의 역사	굿스포츠	7000
3	축구의 이해	대한미디어	22000
5	피겨 교본	굿스포츠	8000

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

SUM(SALEPRICE)
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;
```

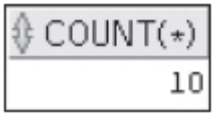


표 3-4 집계 함수의 종류

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
4	2	33000
3	3	31000

ORDERID	CUSTID	BOOKID	SALEPRICE	ORDERDATE
2	1	3	21000	14/07/03
6	1	2	12000	14/07/07
1	1	1	6000	14/07/01
9	2	10	7000	14/07/09
3	2	5	8000	14/07/03
4	3	6	6000	14/07/04
10	3	8	13000	14/07/10
8	3	10	12000	14/07/08
7	4	8	13000	14/07/07
5	4	7	20000	14/07/05

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
4	2
3	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>	GROUP BY로 투플을 그룹으로 묶은 후 SELECT 절에는 GROUP BY에서 사용한 <속성>과 집계함수만 나올 수 있음 ▪ 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>	WHERE 절과 HAVING 절이 같이 포함된 SQL 문은 검색조건이 모호해질 수 있음. HAVING 절은 ① 반드시 GROUP BY 절과 같이 작성해야 하고 ② WHERE 절보다 뒤에 나와야 함. 그리고 ③ <검색조건>에는 SUM, AVG, MAX, MIN, COUNT와 같은 집계함수가 와야 함. • 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 투플의 개수는 고객이 다섯 명이고 주문이 열 개이므로 5×10 해서 50이 된다.**

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

CUSTID	NAME	ADDRESS	PHONE	ORDERID	CUSTID_1	BOOKID	SALEPRICE	ORDERDATE
1	박지성	영국 맨체스타	000-5000-0001	1	1	1	6000	14/07/01
1	박지성	영국 맨체스타	000-5000-0001	2	1	3	21000	14/07/03
1	박지성	영국 맨체스타	000-5000-0001	3	2	5	8000	14/07/03
1	박지성	영국 맨체스타	000-5000-0001	4	3	6	6000	14/07/04
1	박지성	영국 맨체스타	000-5000-0001	5	4	7	20000	14/07/05
1	박지성	영국 맨체스타	000-5000-0001	6	1	2	12000	14/07/07
1	박지성	영국 맨체스타	000-5000-0001	7	4	8	13000	14/07/07
1	박지성	영국 맨체스타	000-5000-0001	8	3	10	12000	14/07/08
1	박지성	영국 맨체스타	000-5000-0001	9	2	10	7000	14/07/09
1	박지성	영국 맨체스타	000-5000-0001	10	3	8	13000	14/07/10
2	김연아	대한민국 서울	000-6000-0001	1	1	1	6000	14/07/01
2	김연아	대한민국 서울	000-6000-0001	2	1	3	21000	14/07/03
2	김연아	대한민국 서울	000-6000-0001	3	2	5	8000	14/07/03
2	김연아	대한민국 서울	000-6000-0001	4	3	6	6000	14/07/04
2	김연아	대한민국 서울	000-6000-0001	5	4	7	20000	14/07/05
2	김연아	대한민국 서울	000-6000-0001	6	1	2	12000	14/07/07
2	김연아	대한민국 서울	000-6000-0001	7	4	8	13000	14/07/07
2	김연아	대한민국 서울	000-6000-0001	8	3	10	12000	14/07/08

… 중략 …

5	박세리	대한민국 대전	(null)	5	4	7	20000	14/07/05
5	박세리	대한민국 대전	(null)	6	1	2	12000	14/07/07
5	박세리	대한민국 대전	(null)	7	4	8	13000	14/07/07
5	박세리	대한민국 대전	(null)	8	3	10	12000	14/07/08
5	박세리	대한민국 대전	(null)	9	2	10	7000	14/07/09
5	박세리	대한민국 대전	(null)	10	3	8	13000	14/07/10

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_1	BOOKID	SALEPRICE	ORDERDATE
1	박지성	영국 맨체스타	000-5000-0001	2	1	3	21000	14/07/03
1	박지성	영국 맨체스타	000-5000-0001	6	1	2	12000	14/07/07
1	박지성	영국 맨체스타	000-5000-0001	1	1	1	6000	14/07/01
2	김연아	대한민국 서울	000-6000-0001	9	2	10	7000	14/07/09
2	김연아	대한민국 서울	000-6000-0001	3	2	5	8000	14/07/03
3	장미란	대한민국 강원도	000-7000-0001	4	3	6	6000	14/07/04
3	장미란	대한민국 강원도	000-7000-0001	10	3	8	13000	14/07/10
3	장미란	대한민국 강원도	000-7000-0001	8	3	10	12000	14/07/08
4	추신수	미국 클리블랜드	000-8000-0001	7	4	8	13000	14/07/07
4	추신수	미국 클리블랜드	000-8000-0001	5	4	7	20000	14/07/05

3.3.1 조인_2개의 테이블을 합체해보자

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 맨체스타	000-5000-0001	2	1	3	21000	14/07/03
1	박지성	영국 맨체스타	000-5000-0001	6	1	2	12000	14/07/07
1	박지성	영국 맨체스타	000-5000-0001	1	1	1	6000	14/07/01
2	김연아	대한민국 서울	000-6000-0001	9	2	10	7000	14/07/09
2	김연아	대한민국 서울	000-6000-0001	3	2	5	8000	14/07/03
3	장미란	대한민국 강원도	000-7000-0001	4	3	6	6000	14/07/04
3	장미란	대한민국 강원도	000-7000-0001	10	3	8	13000	14/07/10
3	장미란	대한민국 강원도	000-7000-0001	8	3	10	12000	14/07/08
4	추신수	미국 클리블랜드	000-8000-0001	7	4	8	13000	14/07/07
4	추신수	미국 클리블랜드	000-8000-0001	5	4	7	20000	14/07/05

3.3.1 조인_2개의 테이블을 합체해보자

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

3.3.1 조인_2개의 테이블을 합체해보자

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)

BOOKID	BOOKNAME	PUBLISHER	PRICE
1	축구의 역사	굿스포츠	7000
2	축구아는 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

ORDERID	CUSTID	BOOKID	SALEPRICE	ORDERDATE
1	1	1	6000	14/07/01
2	1	3	21000	14/07/03
3	2	5	8000	14/07/03
4	3	6	6000	14/07/04
5	4	7	20000	14/07/05
6	1	2	12000	14/07/07
7	4	8	13000	14/07/07
8	3	10	12000	14/07/08
9	2	10	7000	14/07/09
10	3	8	13000	14/07/10

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	축구아는 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 Book.price =20000;
```

NAME	BOOKNAME
추신수	야구의 추억

04. 데이터 정의어

- CREATE 문
- ALTER 문
- DROP 문

4.1 CREATE 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 투플 삭제 시 동작을 나타냄.

- 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 DELETE [CASCADE | SET NULL]
     }
 )
```

4.1 CREATE 문

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

- 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 생성한다.

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

=

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

4.1 CREATE 문

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

```
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 기본 값 10000을 저장한다. 또 가격은 최소 1,000원 이상으로 한다.

※ NewBook 테이블의 CREATE 문에 좀 더 복잡한 제약사항을 추가한다.

```
CREATE TABLE NewBook (
  bookname      VARCHAR(20)      NOT NULL,
  publisher     VARCHAR(20)      UNIQUE,
  price         NUMBER           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) - 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) );
```

4.1 CREATE 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 );
```

4.1 CREATE 문

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

표 3-9 속성의 데이터 타입 종류

데이터 타입	설명	비슷한 타입
NUMBER(p, s)	실수형 p자리 정수, s자리 소수 부분. P와 s를 생략하여 NUMBER라고 쓰면 NUMBER(8, 2)로 저장됨.	DECIMAL(p, s) NUMBER[(p,s)] INTEGER, INT SMALLINT
CHAR(n)	문자형 고정길이. 문자를 저장하고 남은 공간은 공백으로 채움.	CHARACTER(n) CHAR(n)
VARCHAR2(n)	문자형 가변길이. 4000바이트까지 저장됨.	CHARACTER(n) VARYING(n) CHAR(n) VARYING(n)
DATE	날짜형, 연도/월/날/시간을 지정함.	

4.2 ALTER 문

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

- **ALTER 문의 기본 문법**

```
ALTER TABLE 테이블이름
        [ADD 속성이름 데이터타입]
        [DROP COLUMN 속성이름]
        [MODIFY 속성이름 데이터타입]
        [MODIFY 속성이름 [NULL | NOT NULL]]
        [ADD PRIMARY KEY(속성이름)]
        [[ADD | DROP] 제약이름]
```

4.2 ALTER 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);**

4.3 DROP 문

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

- **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;
```

05. 데이터 조작어 – 삽입, 수정, 삭제

- INSERT 문
- UPDATE 문
- DELETE 문

5.1 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	축구아는 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

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 (14, '스포츠 의학', '한솔의학서적');
```

BOOKID	BOOKNAME	PUBLISHER	PRICE
11	스포츠 의학	한솔의학서적	90000
14	스포츠 의학	한솔의학서적	(null)
1	축구의 역사	굿스포츠	7000
2	축구아는 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

5.1 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.
(Imported_book 테이블은 스크립트 Book 테이블과 같이 이미 만들어져 있음)

```
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	축구아는 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

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 변경하시오.

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

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 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 LIKE '박세리';
```

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 서울	(null)

5.3 DELETE 문

- **DELETE 문은 테이블에 있는 기존 투플을 삭제하는 명령임.**
- **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 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

질의 3-50 모든 고객을 삭제하시오.

```
DELETE    FROM    Customer;
```

```
명령의 1 행에서 시작하는 중 오류 발생 -
DELETE  FROM  Customer
오류 보고 -
SQL 오류: ORA-02292: 무결성 제약조건(MADANG.SYS_C0011117)이 위배되었습니다- 자식 레코드가 발견되었습니다
02292. 00000 - "integrity constraint (%s.%s) violated - child record found"
*Cause:    attempted to delete a parent key value that had a foreign
           dependency.
*Action:   delete dependencies first then parent or disable constraint.
```