오라클로 배우는

데이터베이스 개론과 실습

2판

박우창, 남송휘, 이현룡 지음

[강의교안 이용 안내]

오라클로 배우는

데이터베이스 개론과 실습

2판

박우창, 남송휘, 이현룡 지음

Chapter 04. SQL 고급

목차

학습목표

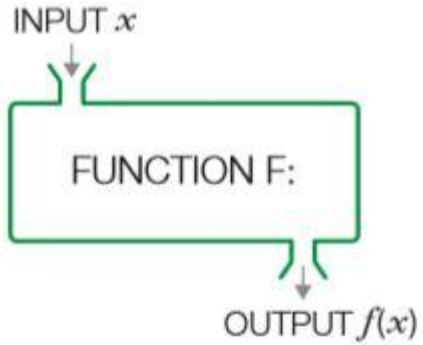
- 내장 함수의 의미를 알아보고 자주 사용되는 내장 함수 몇 가지를 직접 실습해 본다
- 부속질의의 의미와 종류를 알아보고 직접 실습해 본다.
- 뷰의 의미를 알아보고 뷰를 직접 생성, 수정, 삭제해 본다.
- 데이터베이스의 저장 구조와 인덱스의 관계를 알아보고 인덱스를 직접 생성, 수정, 삭제해 본다.

01. 내장함수

- SQL 내장 함수
- NULL 값 처리
- ROWNUM

01. 내장함수

- SQL에서는 함수의 개념을 사용하는데 수학의 함수와 마찬가지로 특정 값이나 열의 값을 입력받아 그 값을 계산하여 결과 값을 돌려줌.
  - 함수의 원리



- SQL의 함수는 DBMS가 제공하는 내장 함수(built-in function)와 사용자가 필요에 따라 직접 만드는 사용자 정의 함수(user-defined function)로 나뉨.

1.1 SQL 내장함수

- SQL 내장 함수는 상수나 속성 이름을 입력 값으로 받아 단일 값을 결과로 반환함.
- 모든 내장 함수는 최초에 선언될 때 유효한 입력 값을 받아야 함.

표 4-1 오라클에서 제공하는 주요 내장 함수

구분		함수
단일행 함수	숫자 함수	ABS, CEIL, COS, EXP, FLOOR, LN, LOG, MOD, POWER, ROUND(number), SIGN, TRUNC(number)
	문자 함수 (문자 반환)	CHR, CONCAT, LOWER, LPAD, LTRIM, STR, REPLACE, RPAD, RTRIM, SUBSTR, TRIM, UPPER
	문자 함수 (숫자 반환)	ASCII, INSTR, LENGTH
	날짜 · 시간 함수	ADD_MONTHS, LAST_DAY, NEXT_DAY, ROUND(date), SYSDATE, TO_CHAR(datetime)
	변환 함수	ASCIISTR, CONVERT, TO_BINARY_DOUBLE, TO_BINARY_FLOAT, TO_CHAR(character), TO_CHAR(datetime), TO_CHAR(number), TO_DATE, TO_NUMBER
	인코딩과 디코딩	DECODE, DUMP, VSIZE
	NULL 관련 함수	COALESCE, NULLIF, NVL
집계 함수		AVG, COUNT, CUME_DIST, FIRST, LAST, MAX, MEDIAN, MIN, PERCENT_RANK, PERCENTILE_CONT, SUM
분석 함수		AVG, CORR, COUNT, CUME_DIST, DENSE_RANK, FIRST, FIRST_VALUE, LAST_VALUE, LEAD, MAX, MIN, RANK, SUM

숫자 함수

표 4-2 숫자 함수의 종류

함수	설명	예
ABS(숫자)	숫자의 절댓값 계산	ABS(−4.5) = 4.5
CEIL(숫자)	숫자보다 크거나 같은 최소의 정수	CEIL(4.1) = 5
FLOOR(숫자)	숫자보다 작거나 같은 최소의 정수	FLOOR(4.1) = 4
ROUND(숫자, m)	m 자리를 기준으로 숫자 반올림	ROUND(5.36, 1) = 5.40
LOG(n, 숫자)	숫자의 자연로그 값 반환	LOG(10) = 2.30259
POWER(숫자, n)	숫자의 n제곱 값 계산	POWER(2, 3) = 8
SQRT(숫자)	숫자의 제곱근 값 계산(숫자는 양수)	SQRT(9.0) = 3.0
SIGN(숫자)	숫자가 음수이면 −1, 0이면 0, 양수이면 1	SIGN(3.45) = 1

숫자 함수

- ABS 함수 : 절댓값을 구하는 함수

질의 4-1 -78과 +78의 절댓값을 구하시오.

```
SELECT   ABS(-78), ABS(+78)
FROM     Dual;
```

ABS(-78)	ABS(+78)
78	78

- ROUND 함수 : 반올림한 값을 구하는 함수

질의 4-2 4.875를 소수 첫째 자리까지 반올림한 값을 구하시요.

```
SELECT   ROUND(4.875, 1)
FROM     Dual;
```

ROUND(4.875,1)
4.9

- 숫자 함수의 연산

질의 4-3 고객별 평균 주문 금액을 배 원 단위로 반올림한 값을 구하시요.

```
SELECT   custid "고객번호", ROUND(SUM(saleprice)/COUNT(*), -2) "평균금액"
FROM     Orders
GROUP BY custid;
```

고객번호	평균금액
1	13000
2	7500
4	16500
3	10300

문자 함수

표 4-3 문자 함수의 종류: 문자값 반환 함수(s는 문자열, c는 문자, n과 k는 정수)

함수	설명과 예
CHR(k)	정수 아스키코드를 문자로 반환 예 CHR(68) = 'D'
CONCAT(s1, s2)	두 문자열을 연결 예 CONCAT('마당', '서점') = '마당서점'
INITCAP(s)	문자열의 첫 번째 알파벳을 대문자로 변환 예 INITCAP('the soap') = 'The Soap'
LOWER(s)	대상 문자열을 모두 소문자로 변환 예 LOWER('MR. SCOTT') = 'mr. scott'
LPAD(s, n, c)	대상 문자열의 왼쪽부터 지정한 자릿수까지 지정한 문자로 채움 예 LPAD('Page 1', 10, '*') = '****Page 1'
LTRIM(s1, s2)	대상 문자열의 왼쪽부터 지정한 문자들을 제거 예 LTRIM('<==>BROWNING<==>', '<>=') = 'BROWNING<==>'
REPLACE(s1, s2, s3)	대상 문자열의 지정한 문자를 원하는 문자로 변경 예 REPLACE('JACK and JUE', 'J', 'BL') = 'BLACK and BLUE'
RPAD(s, n, c)	대상 문자열의 오른쪽부터 지정한 자릿수까지 지정한 문자로 채움 예 RPAD('AbC', 5, '*') = 'AbC**'
RTRIM(s1, s2)	대상 문자열의 오른쪽부터 지정한 문자들을 제거 예 RTRIM('<==>BROWNING<==>', '<>=') = '<==>BROWNING'
SUBSTR(s, n, k)	대상 문자열의 지정된 자리에서부터 지정된 길이만큼 잘라서 반환 예 SUBSTR('ABCDEFG', 3, 4) = 'CDEF'
TRIM(c FROM s)	대상 문자열의 양쪽에서 지정된 문자를 삭제(문자열만 넣으면 기본값으로 공백 제거) 예 TRIM('=' FROM '==>BROWNING<==') = '>BROWNING<'
UPPER(s)	대상 문자열을 모두 대문자로 변환 예 UPPER('mr. scott') = 'MR. SCOTT'
ASCII(c)	대상 알파벳 문자의 아스키코드 값을 반환 예 ASCII('D') = 68
INSTR(s1, s2, n, k)	문자열 중 n번째 문자부터 시작하여 찾고자 하는 문자열 s2가 k번째 나타나는 문자열 위치 반환, 예제에서 3번째부터 OR가 2번째 나타나는 자릿수 예 INSTR('CORPORATE FLOOR', 'OR', 3, 2) = 14
LENGTH(s)	대상 문자열의 글자 수를 반환 예 LENGTH('CANDIDE') = 7

문자 함수

- REPLACE : 문자열을 치환하는 함수

질의 4-4 도서제목에 야구가 포함된 도서를 농구로 변경한 후 도서 목록을

```
SELECT    bookid, REPLACE(bookname, '야구', '농구') 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	농구의 추억	이상미디어	20000
8	농구를 부탁해	이상미디어	13000
9	올림픽 이야기	삼성당	7500
10	Olympic Champions	Pearson	13000

문자 함수

- LENGTH : 글자의 수를 세어주는 함수 (단위가 바이트(byte)가 아닌 문자 단위)

질의 4-5 굿스포츠에서 출판한 도서의 제목과 제목의 글자 수를 확인하시오.
(한글은 2바이트 혹은 UNICODE 경우는 3바이트를 차지함)

```
SELECT    bookname "제목", LENGTH(bookname) "글자수",
          LENGTHB(bookname) "바이트수"
FROM      Book
WHERE     publisher='굿스포츠';
```

제목	글자수	바이트수
축구의 역사	6	16
피겨 교본	5	13
역도 단계별기술	8	22

- SUBSTR : 지정한 길이만큼의 문자열을 반환하는 함수

질의 4-6 마당서점의 고객 중에서 같은 성(姓)을 가진 사람이 몇 명이나 되는지 성별 인원수를 구하시오.

```
SELECT    SUBSTR(name, 1, 1) "성", COUNT(*) "인원"
FROM      Customer
GROUP BY  SUBSTR(name, 1, 1);
```

성	
1	
1	
1	
2	

날짜 • 시간 함수

표 4-5 날짜 · 시간 함수의 종류

함수	설명과 예
TO_DATE (char, datetime)	문자형(CHAR) 데이터를 DATE형으로 반환 예 TO_DATE('2020-09-14', 'yyyy-mm-dd')=2020-09-14
TO_CHAR (date, datetime)	DATE형 데이터를 문자열(VARCHAR2)로 반환 예 TO_CHAR(TO_DATE('2020-09-14', 'yyyy-mm-dd'), 'yyyymmdd')='20200914'
ADD_MONTHS (date, 숫자)	날짜에 지정한 달을 더해 DATE형으로 반환(1: 다음 달, -1: 이전 달) 예 ADD_MONTHS(TO_DATE('2020-09-14', 'yyyy-mm-dd'), 12)=2021-09-14
LAST_DAY (date)	날짜에 달의 마지막 날을 DATE형으로 반환 예 LAST_DAY(TO_DATE('2020-09-14', 'yyyy-mm-dd'))=2020-09-30
SYSDATE	DBMS 시스템상의 당일 날짜를 DATE형으로 반환하는 인자가 없는 함수 예 SYSDATE=20/09/20

날짜 • 시간 함수

표 4-6 datetime의 주요 인자

인자	설명
d	요일 순서(1~7, 월=1)
day	요일(월요일~일요일)
dy	요일의 약자(월~일)
dd	1달 중 날짜(1~31)
ddd	1년 중 날짜(1~366)
hh, hh12	12시간(1~12)
hh24	24시간(0~23)
mi	분(0~59)
mm	월 순서(01~12, January=01)
mon	월 이름 약어(Jan~Dec)
month	월 이름(January~December)
ss	초(0~59)
yyyy	4자리 연도
yyy, yy, y	4자리 연도의 마지막 3, 2, 1자리

날짜 • 시간 함수

질의 4-7 마당서점은 주문일로부터 10일 후 매출을 확정한다. 각 주문의 확정일자를

```
SELECT    orderid "주문번호", orderdate "주문일", orderdate+10 "확정"
FROM      Orders;
```

주문번호	주문일	확정
1	20/07/01	20/07/11
2	20/07/03	20/07/13
3	20/07/03	20/07/13
4	20/07/04	20/07/14
5	20/07/05	20/07/15
6	20/07/07	20/07/17
7	20/07/07	20/07/17
8	20/07/08	20/07/18
9	20/07/09	20/07/19
10	20/07/10	20/07/20

날짜 • 시간 함수

- TO_DATE : 문자형으로 저장된 날짜를 날짜형으로 변환하는 함수
- TO_CHAR : 날짜형을 문자형으로 변환하는 함수

질의 4-8 마당서점이 2014년 7월 7일에 주문받은 도서의 주문번호, 주문일, 고객번호, 도서번호를 모두 보이시오. 단 주문일은 'yyyy-mm-dd 요일' 형태로 표시

```
SELECT    orderid "주문번호", TO_CHAR(orderdate, 'yyyy-mm-dd dy') "주문일",
          custid "고객번호", bookid "도서번호"
FROM      Orders
WHERE     orderdate=TO_DATE('20200707', 'yyyymmdd');
```

주문번호	주문일	고객번호	도서번호
6	2020-07-07 화	1	2
7	2020-07-07 화	4	8

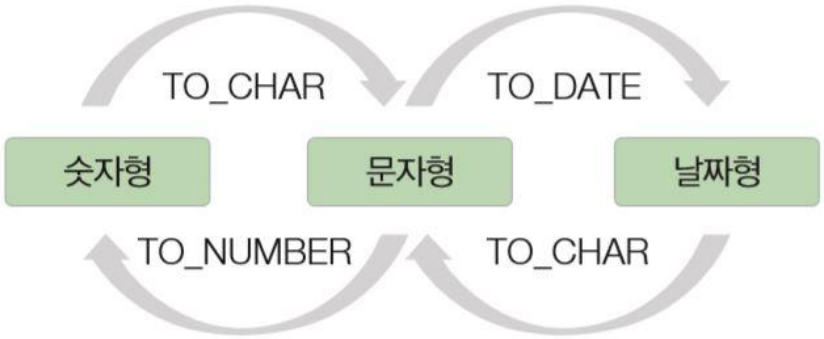


그림 4-1 형 변환 함수 TO_CHAR, TO_DATE, TO_NUMBER

날짜 • 시간 함수

- SYSDATETIME : 오라클의 현재 날짜와 시간을 반환하는 함수
- SYSTIMESTAMP : 현재 날짜, 시간과 함께 초 이하의 시간과 서버의 TIMEZONE까지 출력함

질의 4-9 DBMS 서버에 설정된 현재 시간과 오늘 날짜를 확인하시오.

```
SELECT    SYSDATE,
          TO_CHAR(SYSDATE, 'yyyy/mm/dd dy hh24:mi:ss') "SYSDATE_1"
FROM      Dual;
```

SYSDATE	SYSDATE_1
20/05/28	2020/05/28 목 00:18:48

연습문제

1. 다음 내장 함수의 결과를 적으시오.

- ABS(-15)
- CEIL(15.7)
- COS(3.14159)
- FLOOR(15.7)
- LOG(10,100)
- MOD(11,4)
- POWER(3,2)
- ROUND(15.7)
- SIGN(-15)
- TRUNC(15.7)
- CHR(67)
- CONCAT('HAPPY ', 'Birthday')
- LOWER('Birthday')
- LPAD('Page 1', 15, '*.')
- LTRIM('Page 1', 'ae')
- REPLACE('JACK', 'J', 'BL')
- RPAD('Page 1', 15, '*.')
- RTRIM('Page 1', 'ae')
- SUBSTR('ABCDEFG', 3, 4)
- TRIM(LEADING 0 FROM '00AA00')
- UPPER('Birthday')
- ASCII('A')
- INSTR('CORPORATE FLOOR','OR', 3, 2)
- LENGTH('Birthday')
- ADD_MONTHS('14/05/21', 1)
- LAST_DAY(SYSDATE)
- NEXT_DAY(SYSDATE, '화')
- ROUND(SYSDATE)
- SYSDATE
- TO_CHAR(SYSDATE)
- TO_CHAR(123)
- TO_DATE('12 05 2014', 'DD MM YYYY')
- TO_NUMBER('12.3')
- DECODE(1,1,'aa','bb')
- NULLIF(123, 345)
- NVL(NULL, 123)

1.2 NULL 값 처리

NULL 값이란?

- 아직 지정되지 않은 값
- NULL 값은 '0', '' (빈 문자), ' ' (공백) 등과 다른 특별한 값
- NULL 값은 비교 연산자로 비교가 불가능함.
- NULL 값의 연산을 수행하면 결과 역시 NULL 값으로 반환됨.

집계 함수를 사용할 때 주의할 점

- 'NULL+숫자' 연산의 결과는 NULL
- 집계 함수 계산 시 NULL이 포함된 행은 집계에서 빠짐
- 해당되는 행이 하나도 없을 경우 SUM, AVG 함수의 결과는 NULL이 되며, COUNT 함수의 결과는 0.

1.2 NULL 값 처리

- **NULL 값에 대한 연산과 집계 함수**
  - (* Mybook 테이블 생성은 웹페이지 스크립트 참조)

```
SELECT   price+100
FROM     Mybook
WHERE    bookid=3;
```

Mybook

bookid	price
1	10000
2	20000
3	NULL

PRICE+100
(null)

```
SELECT   SUM(price), AVG(price), COUNT(*), COUNT(price)
FROM     Mybook;
```

SUM(PRICE)	AVG(PRICE)	COUNT(*)	COUNT(PRICE)
30000	15000	3	2

```
SELECT   SUM(price), AVG(price), COUNT(*)
FROM     Mybook
WHERE    bookid >=4;
```

SUM(PRICE)	AVG(PRICE)	COUNT(*)
(null)	(null)	0

1.2 NULL 값 처리

- **NULL 값을 확인하는 방법 – IS NULL, IS NOT NULL**
  - NULL 값을 찾을 때는 '=' 연산자가 아닌 'IS NULL'을 사용,
  - NULL이 아닌 값을 찾을 때는 '< >' 연산자가 아닌 'IS NOT NULL'을 사용함

Mybook

bookid	price
1	10000
2	20000
3	NULL

```
SELECT   *
FROM     Mybook
WHERE    price IS NULL;
```

BOOKID	PRICE
3	(null)

(a) 옳은 예

```
SELECT   *
FROM     Mybook
WHERE    price='';
```

BOOKID	PRICE

(b) 틀린 예

그림 4-2 NULL 값 찾기의 옳은 예와 틀린 예

1.2 NULL 값 처리

- **NVL : NULL 값을 다른 값으로 대치하여 연산하거나 다른 값으로 출력**
  - NVL(속성, 값) /* 속성 값이 NULL이면 '값'으로 대치한다 */

질의 4-10 이름, 전화번호가 포함된 고객목록을 보이시오.
단, 전화번호가 없는 고객은 '연락처없음' 으로 표시한다.

```
SELECT    name "이름", NVL(phone, '연락처없음') "전화번호"
FROM      Customer;
```

이름	전화번호
박지성	000-5000-0001
김연아	000-6000-0001
장미란	000-7000-0001
추신수	000-8000-0001
박세리	연락처없음

1.3 ROWNUM

- 내장 함수는 아니지만 자주 사용되는 문법임.
- 오라클에서 내부적으로 생성되는 가상 컬럼으로 SQL 조회 결과의 순번을 나타냄.
- 자료를 일부분만 확인하여 처리할 때 유용함.

질의 4-11 고객 목록에서 고객번호, 이름, 전화번호를 앞의 두 명만 보이시오.

```
SELECT    ROWNUM "순번", custid, name, phone
FROM      Customer
WHERE     ROWNUM <=2;
```

순번	CUSTID	NAME	PHONE
1	1	박지성	000-5000-0001
2	2	김연아	000-6000-0001

연습문제

2. Mybook 테이블을 생성하고 NULL에 관한 다음 SQL 문에 답하시오.

질의의 결과를 보면서 NULL에 대한 개념을 정리해보시오.

Mybook

bookid	price
1	10000
2	20000
3	NULL

SQL 문 **결과**

(1)
```
SELECT  *
FROM    Mybook;
```

(2)
```
SELECT  bookid, NVL(price, 0)
FROM    Mybook;
```

(3)
```
SELECT  *
FROM    Mybook
WHERE   price IS NULL;
```

(4)
```
SELECT  *
FROM    Mybook;
WHERE   price='';
```

(5)
```
SELECT  bookid, price+100
FROM    Mybook;
```

(6)
```
SELECT  SUM(price), AVG(price), COUNT(*)
FROM    Mybook
WHERE   bookid >= 4;
```

(7)
```
SELECT  COUNT(*), COUNT(price)
FROM    Mybook;
```

(8)
```
SELECT  SUM(price), AVG(price)
FROM    Mybook;
```

연습문제

3. ROWNUM에 관한 다음 SQL 문에 답하시오. 데이터는 마당서점 데이터베이스를 이용

SQL 문 **결과**

(1)
```
SELECT  *
FROM    Book;
```

(2)
```
SELECT  *
FROM    Book
WHERE   ROWNUM <= 5;
```

(3)
```
SELECT  *
FROM    Book
WHERE   ROWNUM <= 5
ORDER BY price;
```

(4)
```
SELECT  *
FROM    (SELECT * FROM Book ORDER BY price) b
WHERE   ROWNUM <= 5;
```

(5)
```
SELECT  *
FROM    (SELECT * FROM Book WHERE ROWNUM<=5) b
ORDER BY price;
```

(6)
```
SELECT  *
FROM (SELECT * FROM Book WHERE ROWNUM <= 5 ORDER BY price) b;
```

02. 부속질의

- 중첩질의 – WHERE 부속질의
- 스칼라 부속질의 – SELECT 부속질의
- 인라인 뷰 – FROM 부속질의

02. 부속질의

부속질의(subquery)란?

- 하나의 SQL 문 안에 다른 SQL 문이 중첩된nested 질의를 말함.
- 다른 테이블에서 가져온 데이터로 현재 테이블에 있는 정보를 찾거나 가공할 때 사용함.
- 보통 데이터가 대량일 때 데이터를 모두 합쳐서 연산하는 조인보다 필요한 데이터만 찾아서 공급해주는 부속질의가 성능이 더 좋음.
- 주 질의(main query, 외부질의)와 부속질의(sub query, 내부질의)로 구성됨.

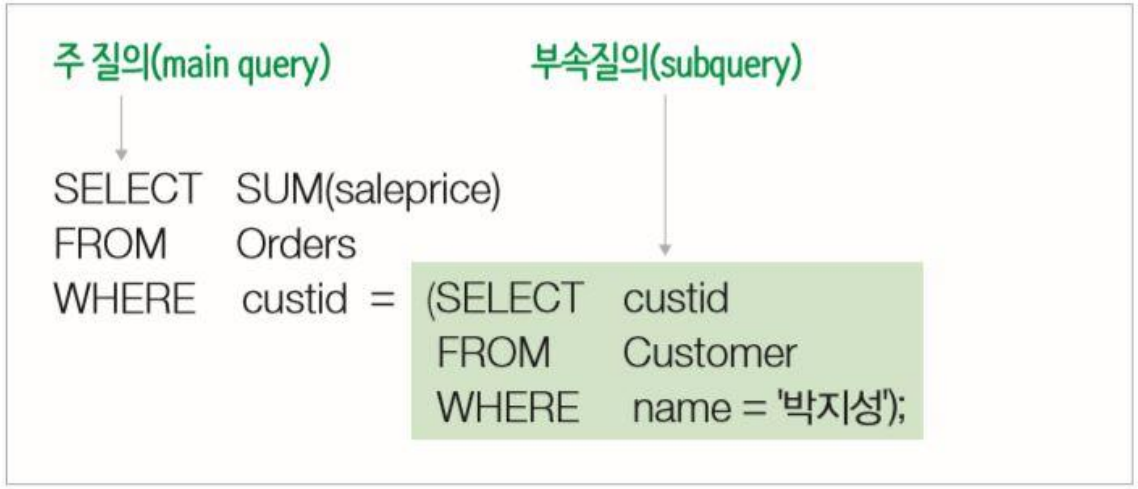


그림 4-3 부속질의

02. 부속질의

표 4-8 부속질의의 종류

명칭	위치	영문 및 동의어	설명
중첩질의	WHERE 절	nested subquery, predicate subquery	WHERE 절에 술어와 같이 사용되며 결과를 한정시키기 위해 사용된다. 상관 혹은 비상관 형태이다.
스칼라 부속질의	SELECT 절	scalar subquery	SELECT 절에서 사용되며 단일 값을 반환하기 때문에 스칼라 부속질의라고 한다.
인라인 뷰	FROM 절	inline view, table subquery	FROM 절에서 결과를 뷰(view) 형태로 반환하기 때문에 인라인 뷰라고 한다.

2.1 중첩질의 – WHERE 부속질의

- 중첩질의(nested subquery)는 WHERE 절에서 사용되는 부속질의.
- WHERE 절은 보통 데이터를 선택하는 조건 혹은 술어(predicate)와 같이 사용됨. 그래서 중첩질의를 술어 부속질의(predicate subquery)라고도 함.

표 4-9 중첩질의 연산자의 종류

술어	연산자	반환 행	반환 열	상관
비교	=, >, <, >=, <=, <>	단일	단일	가능
집합	IN, NOT IN	다중	다중	가능
한정	ALL, SOME(ANY)	다중	단일	가능
존재	EXISTS, NOT EXISTS	다중	다중	필수

2.1 중첩질의 – WHERE 부속질의

■ 비교 연산자

- 부속질의가 반드시 단일 행, 단일 열을 반환해야 하며, 아닐 경우 질의를 처리할 수 없음.

질의 4-12 평균 주문금액 이하의 주문에 대해서 주문번호와 금액을 보이시오.

```
SELECT    orderid, saleprice
FROM      Orders
WHERE     saleprice <=(SELECT AVG(saleprice)
                       FROM   Orders);
```

ORDERID	SALEPRICE
1	6000
3	8000
4	6000
9	7000

질의 4-13 각 고객의 평균 주문금액보다 큰 금액의 주문 내역에 대해서 주문번호, 고객번호,

```
SELECT    orderid, custid, saleprice
FROM      Orders md
WHERE     saleprice > (SELECT     AVG(saleprice)
                       FROM       Orders so
                       WHERE      md.custid=so.custid);
```

ORDERID	CUSTID	SALEPRICE
2	1	21000
3	2	8000
5	4	20000
8	3	12000
10	3	13000

2.1 중첩질의 – WHERE 부속질의

IN, NOT IN

- IN 연산자는 주질의 속성 값이 부속질의에서 제공한 결과 집합에 있는지 확인(check)하는 역할을 함. IN 연산자는 부속질의의 결과 다중 행을 가질 수 있음. 주질의는 WHERE 절에 사용되는 속성 값을 부속질의의 결과 집합과 비교해 하나라도 있으면 참이 된다. NOT IN은 이와 반대로 값이 존재하지 않으면 참이 됨.

질의 4-14 대한민국에 거주하는 고객에게 판매한 도서의 총판매액을

```
SELECT   SUM(saleprice) "total"
FROM     Orders
WHERE    custid IN (SELECT   custid
                    FROM     Customer
                    WHERE    address LIKE '%대한민국%');
```

total
46000

2.1 중첩질의 – WHERE 부속질의

- ALL, SOME(ANY)
  - ALL은 모두, SOME(ANY)은 어떠한(최소한 하나라도)이라는 의미를 가짐.

- 구문 구조

```
scalar_expression { 비교 연산자 ( =, <>, !=, >, >=, !>, <, <=, !< ) }
{ ALL | SOME | ANY } (부속질의)
```

질의 4-15 3번 고객이 주문한 도서의 최고 금액보다 더 비싼 도서를 구입한 주문의

```
SELECT    orderid, saleprice
FROM      Orders
WHERE     saleprice > ALL (SELECT    saleprice
                           FROM      Orders
                           WHERE     custid='3');
```

ORDERID	SALEPRICE
5	20000
2	21000

2.1 중첩질의 – WHERE 부속질의

EXISTS, NOT EXISTS

- 데이터의 존재 유무를 확인하는 연산자
- 주질의에서 부속질의로 제공된 속성의 값을 가지고 부속질의에 조건을 만족하여 값이 존재하면 참이 되고, 주질의는 해당 행의 데이터를 출력함.
- NOT EXIST의 경우 이와 반대로 동작함.

구문 구조

질의 4-16 EXISTS 연산자로 대한민국에 거주하는 고객에게 판매한 도서의 총 판매액을

```
SELECT    SUM(saleprice) "total"
FROM      Orders od
WHERE     EXISTS (SELECT *
                  FROM   Customer cs
                  WHERE  address LIKE '%대한민국%' AND cs.custid=od.custid);
```

total
46000

2.2 스칼라 부속질의 – SELECT 부속질의

- **스칼라 부속질의(scalar subquery)란?**
  - SELECT 절에서 사용되는 부속질의로, 부속질의의 결과 값을 단일 행, 단일 열의 스칼라 값으로 반환함.
  - 스칼라 부속질의는 원칙적으로 스칼라 값이 들어갈 수 있는 모든 곳에 사용 가능하며, 일반적으로 SELECT 문과 UPDATE SET 절에 사용됨.
  - 주질의와 부속질의와의 관계는 상관/비상관 모두 가능함.

표 4-4 문자 함수의 종류: 숫자값 반환 함수

함수	설명
ASCII(c)	대상 알파벳 문자의 아스키코드 값을 반환 예 ASCII('D') = 68
INSTR(s1, s2, n, k)	문자열 중 n번째 문자부터 시작하여 찾고자 하는 문자열 s2가 k번째 나타나는 문자열 위치 반환, 예제에서 3번째부터 OR가 2번째 나타나는 자릿수 예 INSTR('CORPORATE FLOOR', 'OR', 3, 2) = 14
LENGTH(s)	대상 문자열의 글자 수를 반환 예 LENGTH('CANDIDE') = 7

2.2 스칼라 부속질의 – SELECT 부속질의

질의 4-17 마당서점의 고객별 판매액을 보이시오(결과는 고객이름과 고객별 판매액을

```
SELECT   (SELECT      name
          FROM        Customer cs
          WHERE       cs.custid=od.custid) "name", SUM(saleprice) "total"
FROM      Orders od
GROUP BY  od.custid;
```

name	total
박지성	39000
김연아	15000
추신수	33000
장미란	31000

2.2 스칼라 부속질의 – SELECT 부속질의

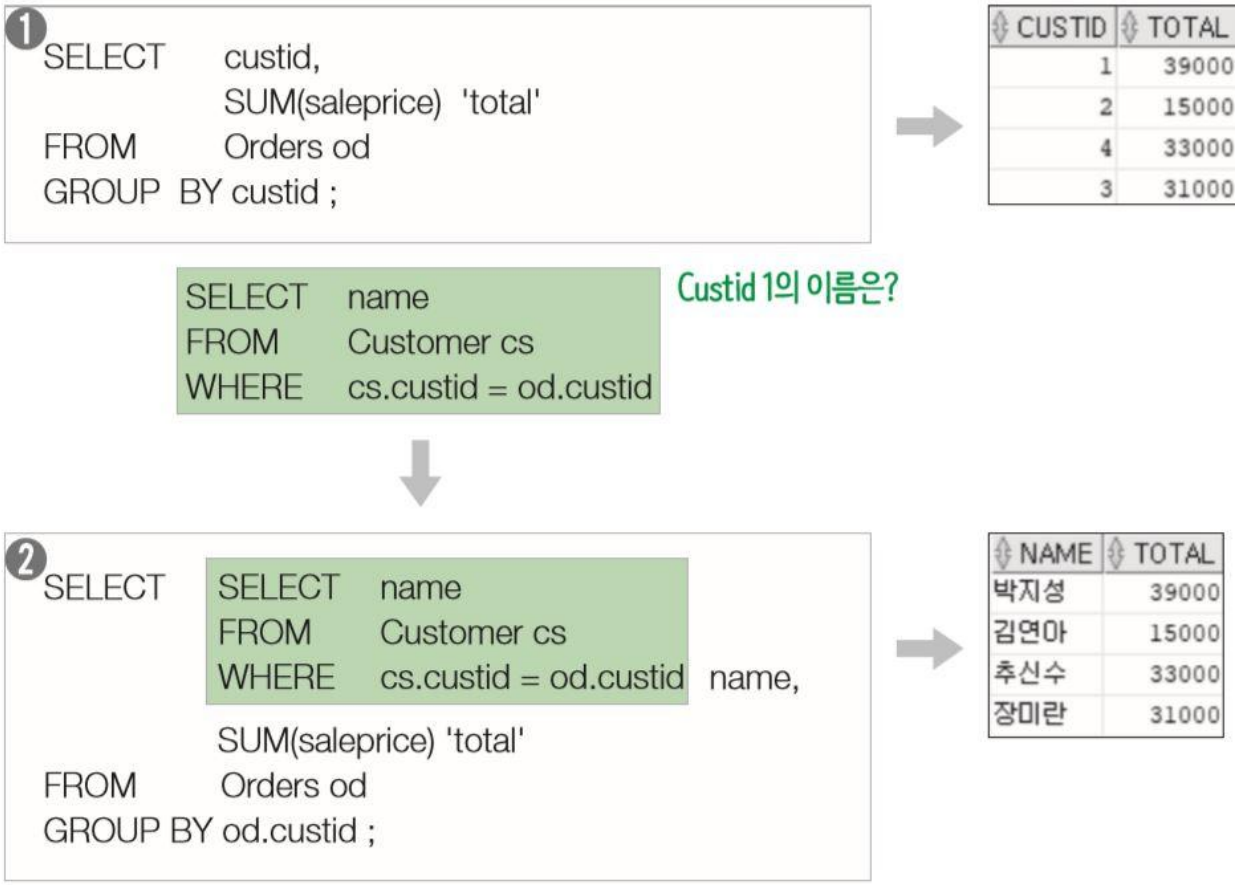


그림 4-5 마당서점의 고객별 판매액

2.2 스칼라 부속질의 – SELECT 부속질의

질의 4-18 Orders 테이블에 각 주문에 맞는 도서이름을 입력하시오.

- 먼저 Orders 테이블에 bookname 속성을 추가

```
ALTER TABLE Orders ADD bookname VARCHAR2(40);
UPDATE    Orders
SET       bookname=(SELECT      bookname
                    FROM        Book
                    WHERE       Book.bookid=Orders.bookid);
```

ORDERID	CUSTID	BOOKID	SALEPRICE	ORDERDATE	BOOKNAME
1	1	1	6000	20/07/01	축구의 역사
2	1	3	21000	20/07/03	축구의 이해
3	2	5	8000	20/07/03	피겨 교본
4	3	6	6000	20/07/04	역도 단계별기술
5	4	7	20000	20/07/05	야구의 추억
6	1	2	12000	20/07/07	축구아는 여자
7	4	8	13000	20/07/07	야구를 부탁해
8	3	10	12000	20/07/08	Olympic Champions
9	2	10	7000	20/07/09	Olympic Champions
10	3	8	13000	20/07/10	야구를 부탁해

2.3 인라인 뷰- FROM 부속질의

인라인 뷰(inline view)란?

- FROM 절에서 사용되는 부속질의.
- 테이블 이름 대신 인라인 뷰 부속질의를 사용하면 보통의 테이블과 같은 형태로 사용할 수 있음.
- 부속질의 결과 반환되는 데이터는 다중 행, 다중 열이어도 상관없음.
- 다만 가상의 테이블인 뷰 형태로 제공되어 상관 부속질의로 사용될 수는 없음.

질의 4-19 고객번호가 2 이하인 고객의 판매액을 보이시오

(결과는 고객이름과 고객별 판매액 출력)

```
SELECT    cs.name, SUM(od.saleprice) "total"
FROM      (SELECT    custid, name
           FROM      Customer
           WHERE     custid <= 2) cs,
           Orders  od
WHERE     cs.custid=od.custid
GROUP BY  cs.name;
```

NAME	total
박지성	39000
김연아	15000

2.3 인라인 뷰- FROM 부속질의

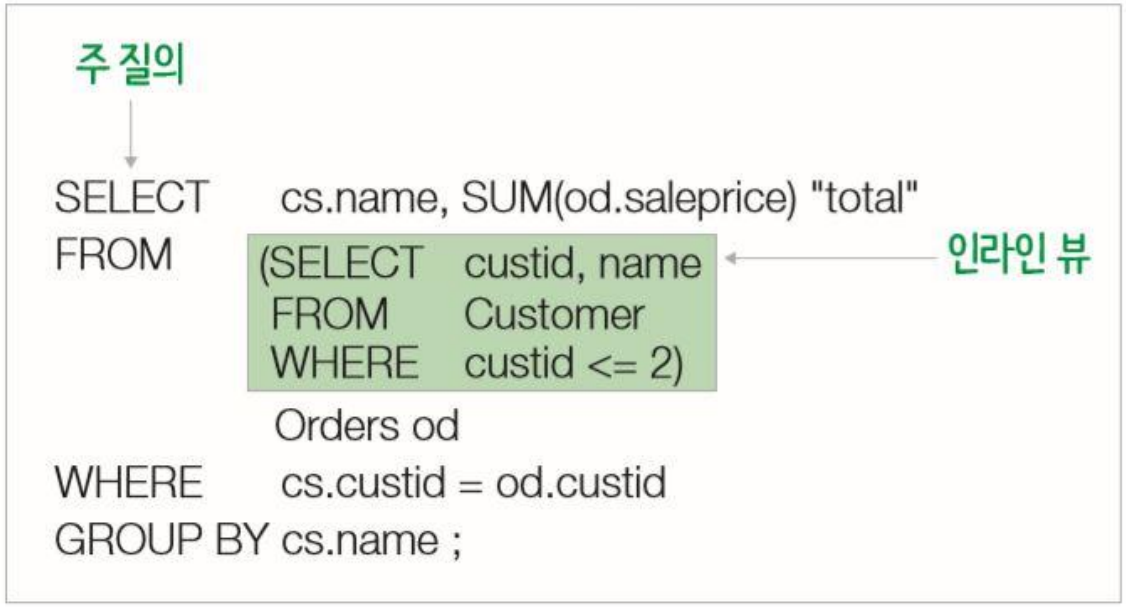


그림 4-6 인라인 뷰

연습문제

5. 부속질의에 관한 다음 SQL 문을 수행해보고 어떤 질의에 대한 답인지 설명하시오.

```
(1) SELECT      custid, (SELECT   address
                         FROM     Customer cs
                         WHERE    cs.custid = od.custid) "address",
                                  SUM(saleprice) "total"
    FROM        Orders od
    GROUP BY    od.custid;

(2) SELECT      cs.name, s
    FROM        (SELECT     custid, AVG(saleprice) s
                 FROM       Orders
                 GROUP BY custid) od, Customer cs
    WHERE       cs.custid = od.custid;

(3) SELECT      SUM(saleprice) "total"
    FROM        Orders od
    WHERE EXISTS (SELECT    *
                  FROM      Customer cs
                  WHERE     custid <= 3 AND cs.custid = od.custid);
```

03. 뷰

- 뷰의 생성
- 뷰의 수정
- 뷰의 삭제

03. 뷰

- **뷰(view)는 하나 이상의 테이블을 합하여 만든 가상의 테이블.**
- **장점**
  - 편리성(및 재사용성) : 자주 사용되는 복잡한 질의를 뷰로 미리 정의해 놓을 수 있음.

    => 복잡한 질의를 간단히 작성, 미리 정의된 뷰를 일반 테이블처럼 사용하여 편리함
  - 보안성 : 각 사용자별로 필요한 데이터만 선별하여 보여줄 수 있음. 중요한 질의의 경우 질의 내용을 암호화할 수 있음.

    => 개인정보(주민번호)나 급여, 건강 같은 민감한 정보를 제외한 테이블을 만들어 사용
  - 논리적 데이터 독립성 제공 :

    => 개념 스키마의 데이터베이스 구조가 변하여도 외부 스키마에 영향을 주지 않도록하는 논리적 데이터 독립성 제공

- **뷰의 특징**
  - 1. 원본 데이터 값에 따라 같이 변함
  - 2. 독립적인 인덱스 생성이 어려움
  - 3. 삽입, 삭제, 갱신 연산에 많은 제약이 따름

03. 뷰

뷰 Vorders

ORDERID	CUSTID	NAME	BOOKID	BOOKNAME	SALEPRICE	ORDERDATE
1	1	박지성	1	축구의 역사	6000	20/07/01
6	1	박지성	2	축구아는 여자	12000	20/07/07
2	1	박지성	3	축구의 이해	21000	20/07/03
3	2	김연아	5	피겨 교본	8000	20/07/03

뷰 생성문

```
CREATE VIEW Vorders
AS SELECT O.orderid, O.custid, C.name, O.bookid, B.bookname, O.saleprice, O.orderdate
    FROM Orders O, Customer C, Book B
    WHERE     O.custid=C.custid AND O.bookid=B.bookid;
```

기본 릴레이션 Customer, Orders, Book

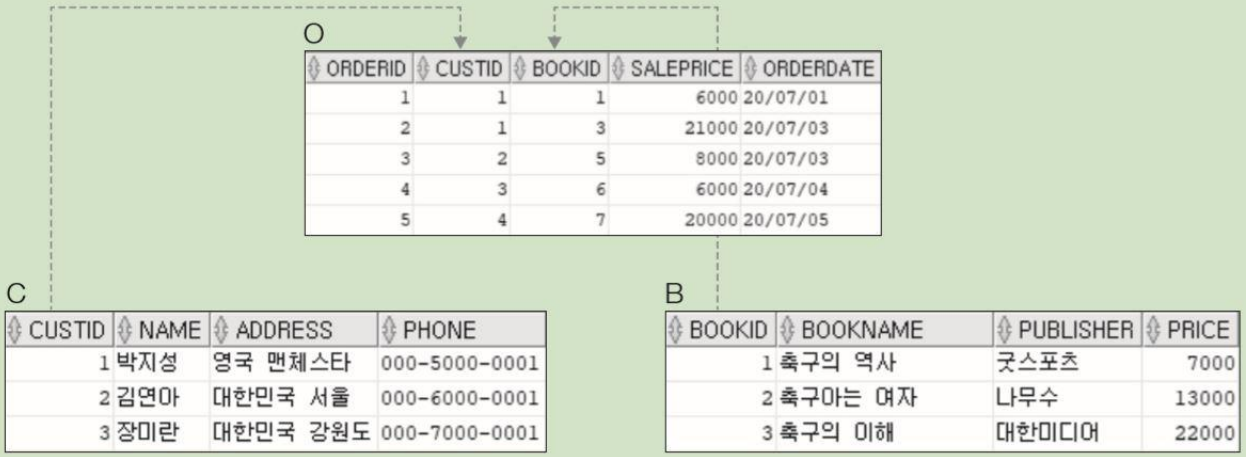


그림 4-7 뷰

3.1 뷰의 생성

- 기본 문법

```
CREATE  VIEW 뷰이름 [(열이름 [,... n])]
AS <SELECT 문>
```

- Book 테이블에서 '축구'라는 문구가 포함된 자료만 보여주는 뷰

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

- 위 SELECT 문을 이용해 작성한 뷰 정의문

```
CREATE VIEW  vw_Book
AS SELECT    *
    FROM     Book
    WHERE    bookname LIKE '%축구%';
```

3.1 뷰의 생성

질의 4-20 주소에 '대한민국'을 포함하는 고객들로 구성된 뷰를 만들고 조회하시오.
뷰의 이름은 vw_ Customer로 설정하시오

```
CREATE VIEW  vw_Customer
AS SELECT    *
    FROM     Customer
    WHERE    address LIKE '%대한민국%';
```

- 결과확인

```
SELECT       *
FROM         vw_Customer;
```

CUSTID	NAME	ADDRESS	PHONE
2	김연아	대한민국 서울	000-6000-0001
3	장미란	대한민국 강원도	000-7000-0001
5	박세리	대한민국 대전	(null)

3.1 뷰의 생성

질의 4-21 Orders 테이블에 고객이름과 도서이름을 바로 확인할 수 있는 뷰를 생성한 후, '김연아' 고객이 구입한 도서의 주문번호, 도서이름, 주문액을 보이시오.

```
CREATE VIEW  vw_Orders (orderid, custid, name, bookid, bookname, saleprice,
             orderdate)
AS SELECT    od.orderid, od.custid, cs.name,
             od.bookid, bk.bookname, od.saleprice, od.orderdate
    FROM     Orders od, Customer cs, Book bk
    WHERE    od.custid=cs.custid AND od.bookid=bk.bookid;
```

- 결과확인

```
SELECT       orderid, bookname, saleprice
FROM         vw_Orders
WHERE        name ='김연아';
```

ORDERID	BOOKNAME	SALEPRICE
3	피겨 교본	8000
9	Olympic Champions	7000

3.2 뷰의 수정

기본 문법

```
CREATE OR REPLACE VIEW 뷰이름 [(열이름 [,... n])]
AS SELECT 문
```

질의 4-22 [질의 4-20]에서 생성한 뷰 vw_Customer는 주소가 대한민국인 고객을 보여준다. 이 뷰를 영국을 주소로 가진 고객으로 변경하시오.

phone 속성은 필요 없으므로 포함시키지 마시오.

```
CREATE OR REPLACE VIEW vw_Customer (custid, name, address)
AS  SELECT      custid, name, address
    FROM        Customer
    WHERE       address LIKE '%영국%';
```

- 결과확인

```
SELECT      *
FROM        vw_Customer;
```

CUSTID	NAME	ADDRESS
1	박지성	영국 맨체스타

3.3 뷰의 삭제

- 기본 문법

```
DROP VIEW 뷰이름 [,... n];
```

질의 4-23 앞서 생성한 뷰 vw_Customer를 삭제하시오.

```
DROP VIEW    vw_Customer;
```

- 결과확인

```
SELECT       *
FROM         vw_Customer;
```

```
ORA-00942: 테이블 또는 뷰가 존재하지 않습니다
00942. 00000 - "table or view does not exist"
*Cause:
*Action:
4행, 6열에서 오류 발생
```

연습문제

08. 다음에 해당하는 뷰를 작성하시오. 데이터베이스는 마당서점 데이터베이스를 이용한다.

(1) 판매가격이 20,000원 이상인 도서의 도서번호, 도서이름, 고객이름, 출판사, 판매가격을 보여주는 highorders 뷰를 생성하시오.

(2) 생성한 뷰를 이용하여 판매된 도서의 이름과 고객의 이름을 출력하는 SQL 문을 작성하시오.

(3) highorders 뷰를 변경하고자 한다. 판매가격 속성을 삭제하는 명령을 수행하시오. 삭제 후 (2)번 SQL 문을 다시 수행하시오.

04. 인덱스

- 데이터베이스의 물리적 저장
- 인덱스와 B-tree
- 오라클 인덱스
- 인덱스의 생성
- 인덱스의 재구성과 삭제

4.1 데이터베이스의 물리적 저장

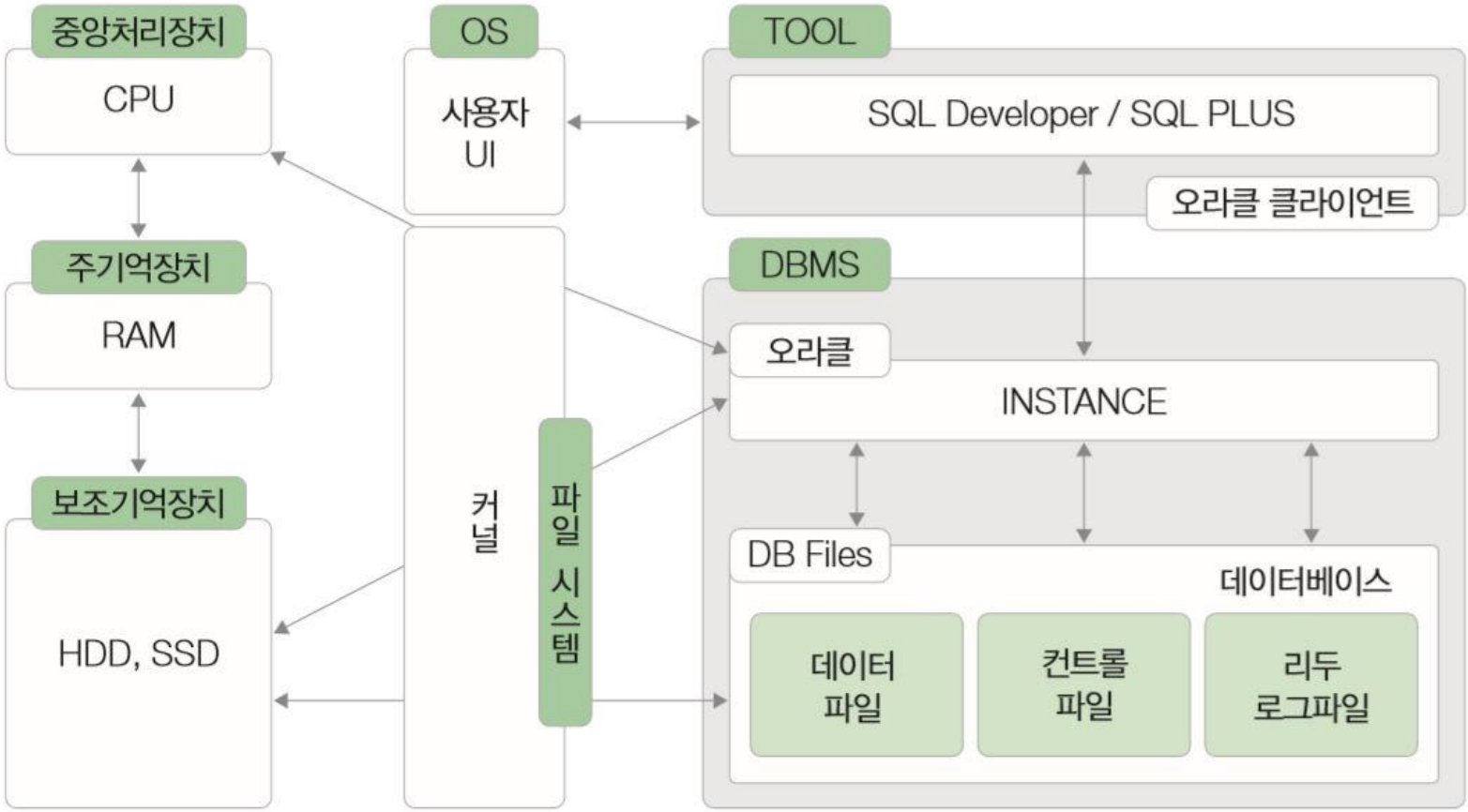


그림 4-8 DBMS와 데이터 파일

4.1 데이터베이스의 물리적 저장

- **실제 데이터가 저장되는 곳은 보조기억장치**
  - 하드디스크, SSD, USB 메모리 등

- **가장 많이 사용되는 장치는 하드디스크**
  - 하드디스크는 원형의 플레이트(plate)로 구성되어 있고, 이 플레이트는 논리적으로 트랙으로 나뉘며 트랙은 다시 몇 개의 섹터로 나뉨.
  - 원형의 플레이트는 초당 빠른 속도로 회전하고, 회전하는 플레이트를 하드디스크의 액세스 암(arm)과 헤더(header)가 접근하여 원하는 섹터에서 데이터를 가져옴.
  - 하드디스크에 저장된 데이터를 읽어 오는 데 걸리는 시간은 모터(motor)에 의해서 분당 회전하는 속도(RPM, Revolutions Per Minute), 데이터를 읽을 때 액세스 암이 이동하는 시간(latency time), 주기억장치로 읽어오는 시간(transfer time)에 영향을 받음.

4.1 데이터베이스의 물리적 저장

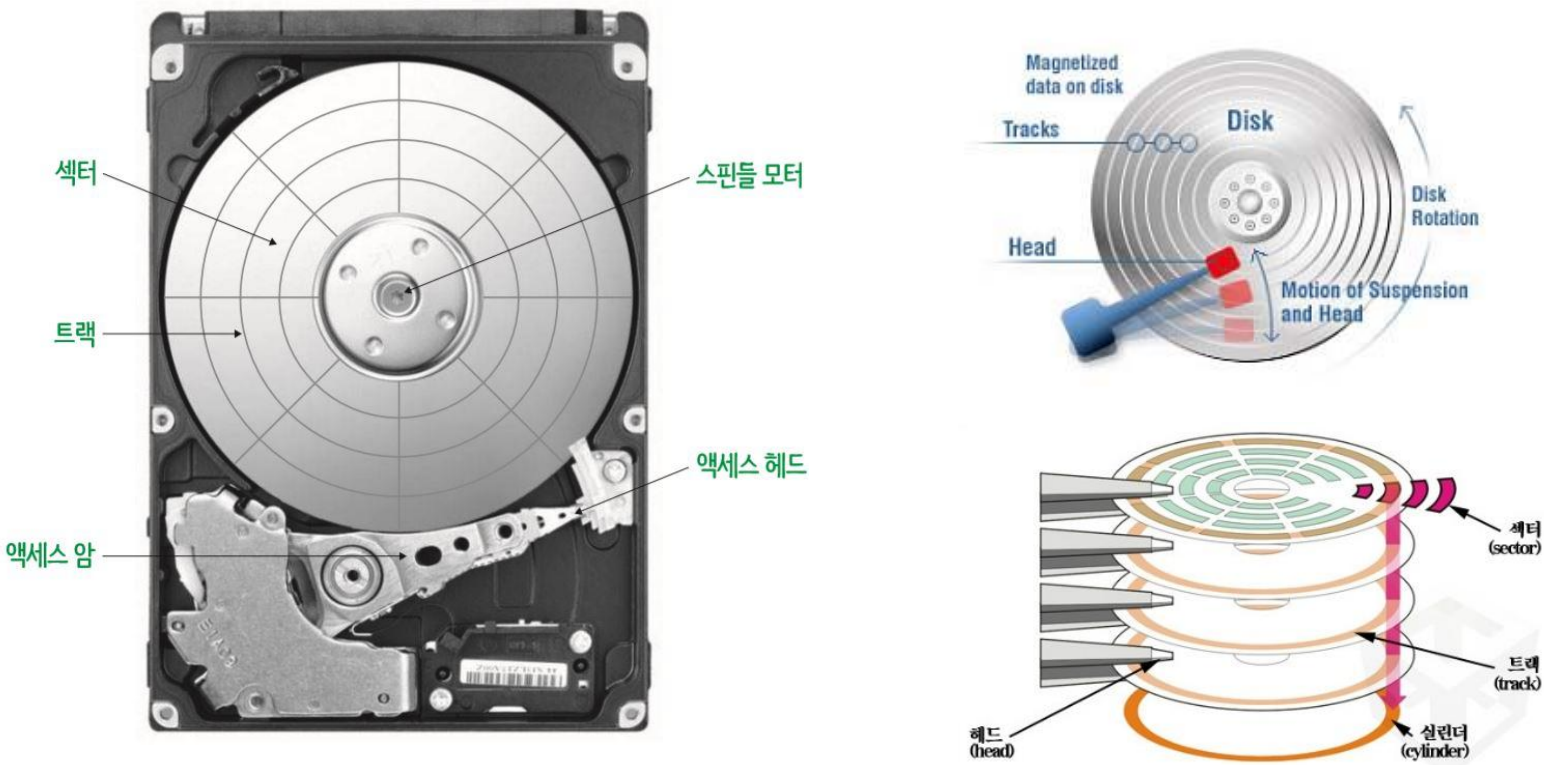


그림 4-9 하드디스크의 구조

- **액세스 시간(access time)**
  - 액세스 시간 = 탐색시간(seek time, 액세스 헤드를 트랙에 이동시키는 시간)
    + 회전지연시간(rotational latency time, 섹터가 액세스 헤드에 접근하는 시간)
    + 데이터 전송시간(data transfer time, 데이터를 주기억장치로 읽어오는 시간)

4.1 데이터베이스의 물리적 저장

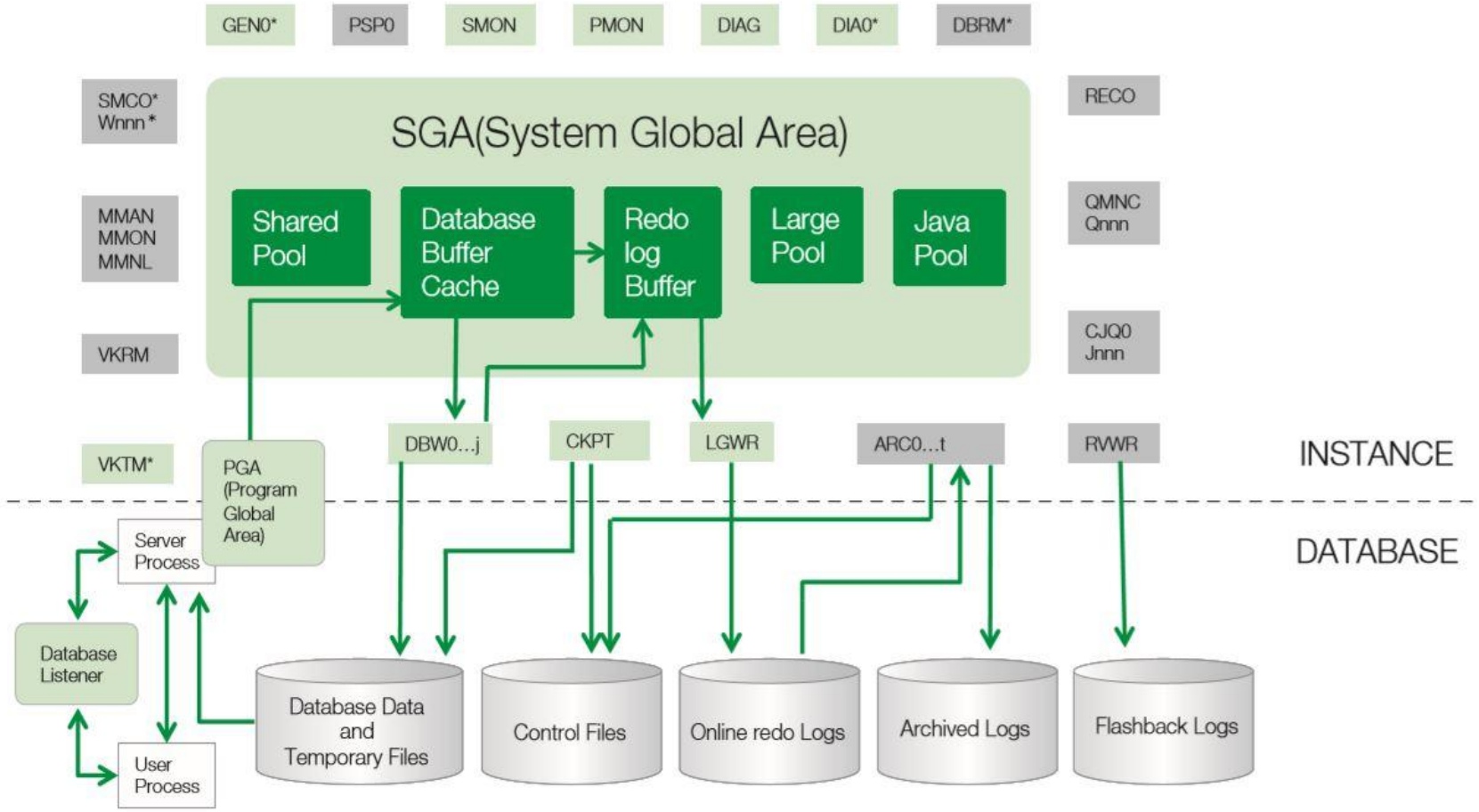


그림 4-10 오라클의 내부 구조

4.1 데이터베이스의 물리적 저장

표 4-10 오라클의 주요 파일

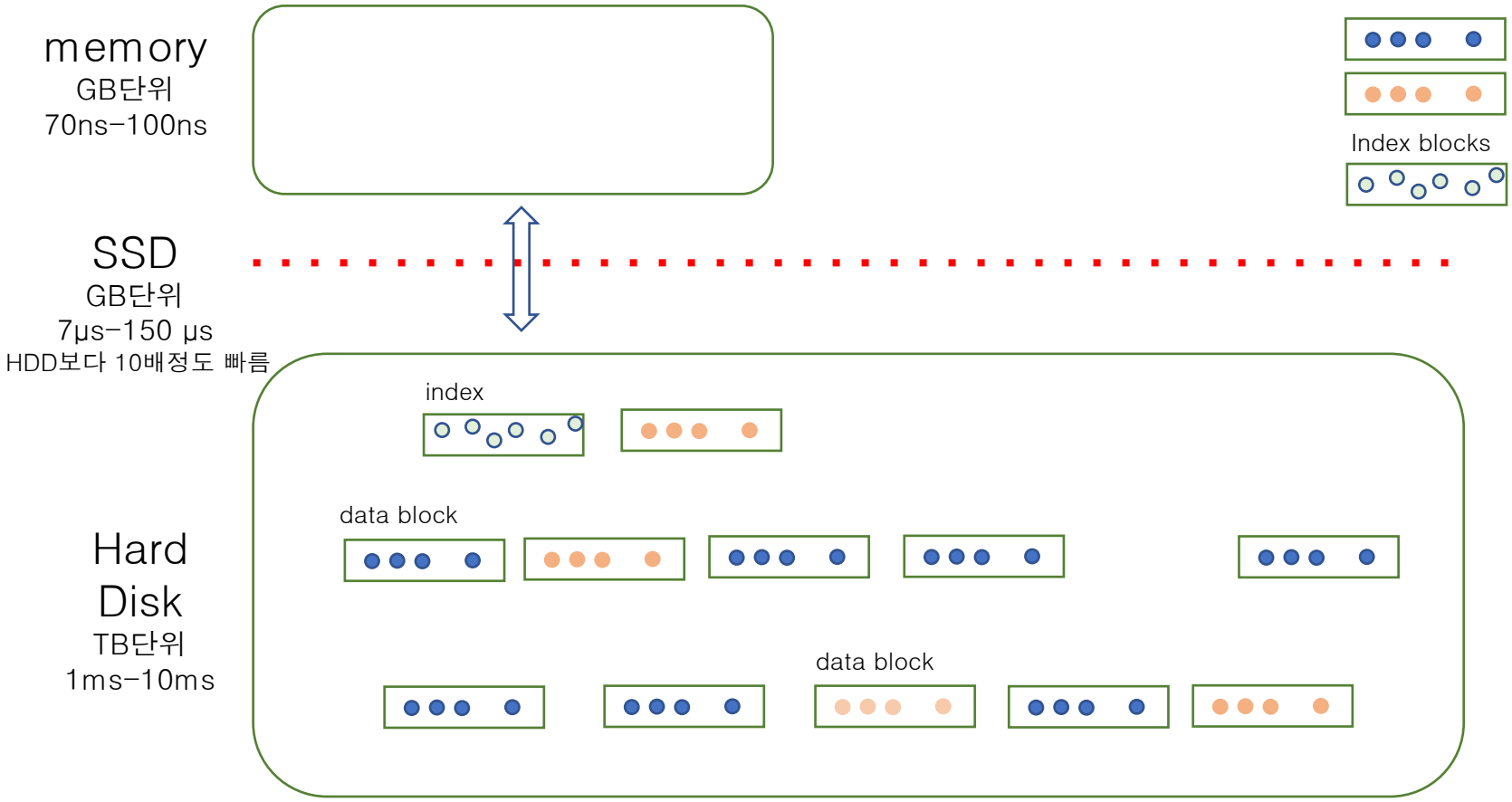
파일	설명
데이터 파일 (Data Files)	• 운영체제상에 물리적으로 존재 • 사용자 데이터와 개체를 저장 • 테이블과 인덱스로 구성
온라인 리두 로그 (Online Redo Log)	• 데이터의 모든 변경사항을 기록 • 데이터베이스 복구에 사용되는 로그 정보 저장 • 최소 두 개의 온라인 리두 로그 파일 그룹을 가짐
컨트롤 파일 (Control File)	• 오라클이 필요로 하는 다른 파일들(데이터 파일, 로그 파일 등)의 위치 정보를 저장 • 데이터베이스 구조 등의 변경사항이 있을 때 자동으로 업데이트됨 • 오라클 DB의 마운트, 오픈의 필수 파일 • 복구 시 동기화 정보 저장

4.2 인덱스와 B-tree

- **인덱스의 필요성**

  질의 검색시 data block(여러 개의 record를 저장하는 저장단위, 2KB 4KB …) 읽는 횟수를 최소화 필요

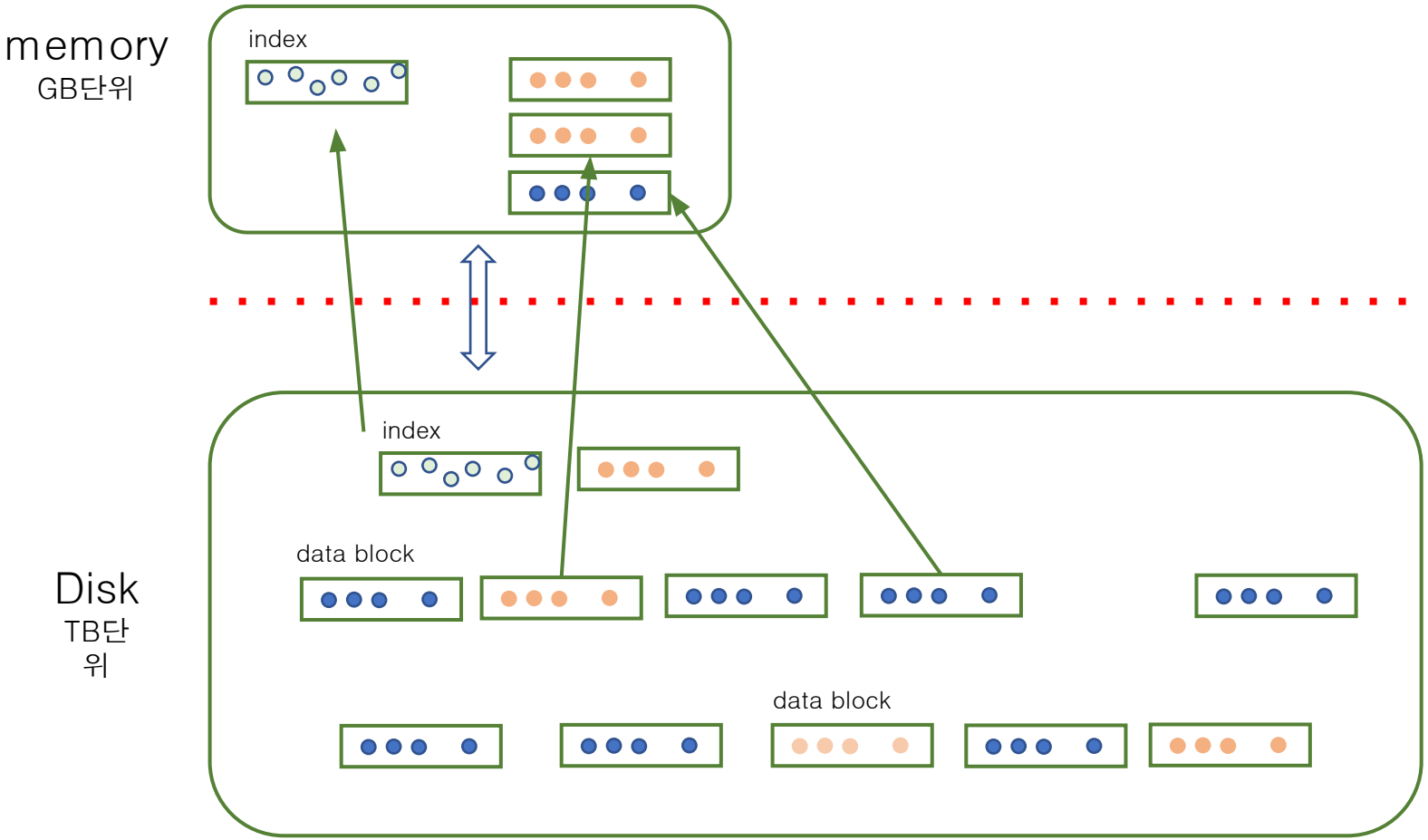
  disk에 있는 데이터는 memory에 있는 데이터에 비하여 읽어들이는 속도가 10000배 정도 소요된다.



㈜ https://www.formulusblack.com/blog/compute-performance-distance-of-data-as-a-measure-of-latency/

4.2 인덱스와 B-tree

- 인덱스의 필요성 memory



4.2 인덱스와 B-tree

- **인덱스(index, 색인)**
  - 도서의 색인이나 사전과 같이 데이터를 쉽고 빠르게 찾을 수 있도록 만든 데이터 구조

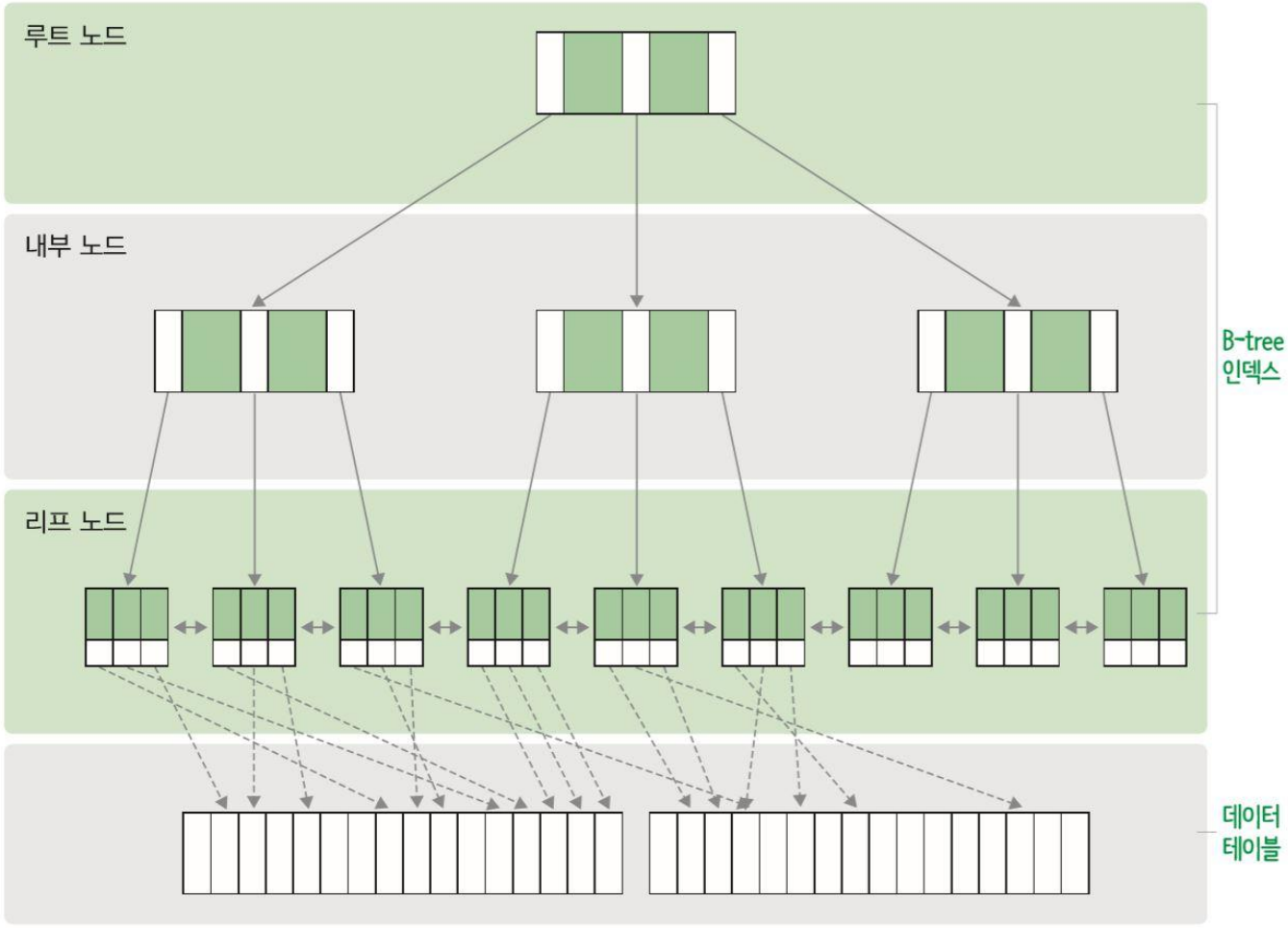


그림 4-12 B-tree의 구조

4.2 인덱스와 B-tree

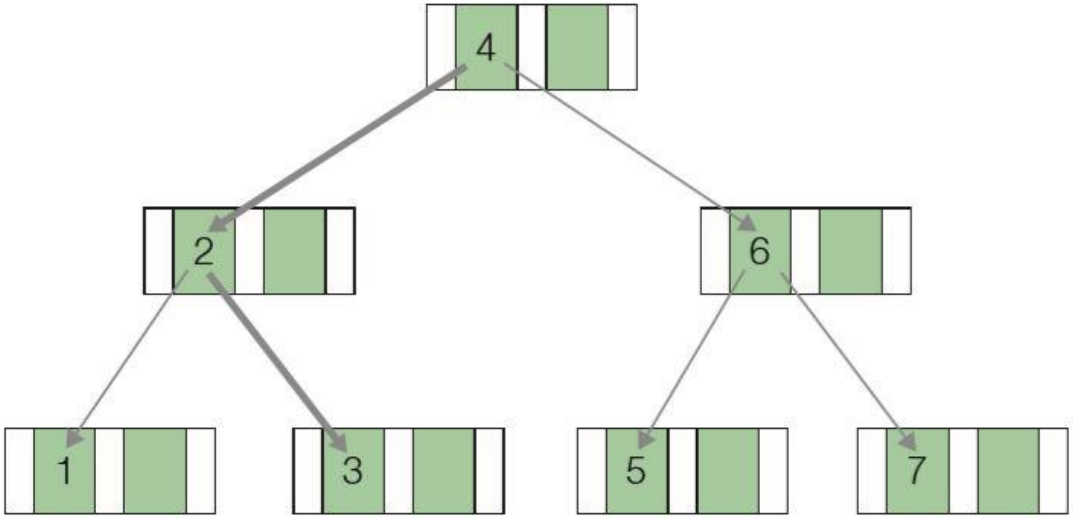


그림 4-13 B-tree에서 검색 예

인덱스의 특징

- 인덱스는 테이블에서 한 개 이상의 속성을 이용하여 생성함.
- 빠른 검색과 함께 효율적인 레코드 접근이 가능함.
- 순서대로 정렬된 속성과 데이터의 위치만 보유하므로 테이블보다 작은 공간을 차지함.
- 저장된 값들은 테이블의 부분집합이 됨.
- 일반적으로 B-tree 형태의 구조를 가짐.
- 데이터의 수정, 삭제 등의 변경이 발생하면 인덱스의 재구성이 필요함.

오라클 B-tree 인덱스

- 오라클 인덱스는 B-tree를 변형하여 사용하며 명칭은 B-tree로 동일한 이름으로 부름

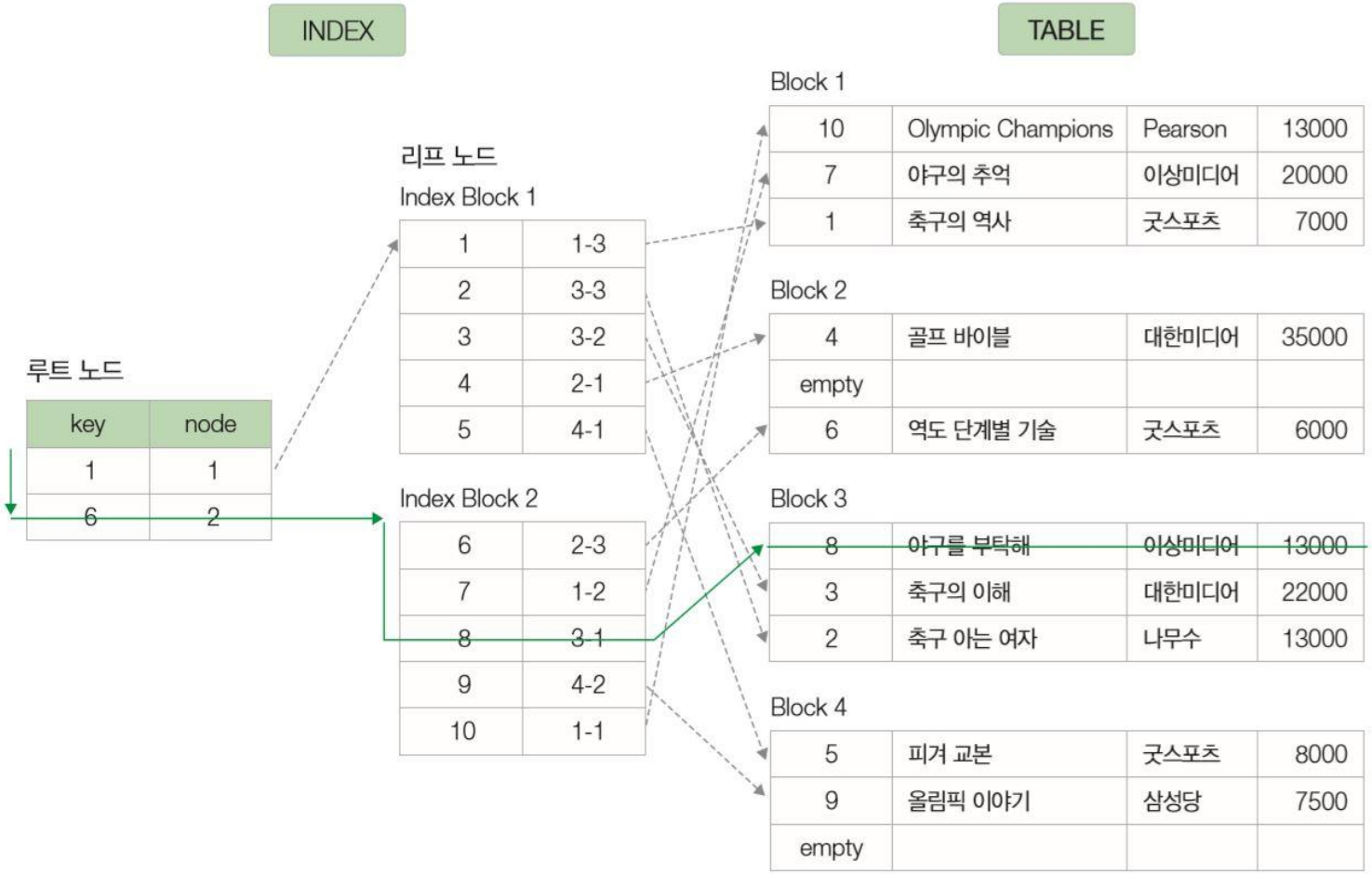


그림 4-14 인덱스의 예

4.2 인덱스와 B-tree

표 4-11 오라클 인덱스의 종류

인덱스 명칭	설명 / 생성 예
B-Tree 인덱스	기본적인 인덱스로, 하나의 리프 노드는 하나의 데이터에 대응 예 CREATE INDEX t_ix ON T(key1);
IOT (Index Organized Table, 인덱스 구조 테이블)	B-Tree 구조로 키값이 정렬되면서 인덱스 리프 노드에 실제 데이터가 같이 저장되는 테이블(인덱스+테이블 일체) 예 CREATE TABLE T (key1 NUMBER, BDATA VARCHAR2(10)) ORGANIZATION INDEX;
Bitmap Index (비트맵 인덱스)	비트맵을 사용하여 하나의 엔트리가 여러 행을 가리킬 수 있도록 생성 예 CREATE BITMAP INDEX bt_ix ON Emp(gender);
Function-Base Index (함수 기반 인덱스)	행과 열에 대한 함수의 결과를 저장한 인덱스 예 CREATE INDEX fbi_ix ON T(UPPER(name));

4.4 인덱스의 생성

인덱스 생성 시 고려사항

- 인덱스는 WHERE 절에 자주 사용되는 속성이어야 함.
- 인덱스는 조인에 자주 사용되는 속성이어야 함.
- 단일 테이블에 인덱스가 많으면 속도가 느려질 수 있음(테이블 당 4~5개 정도 권장).
- 속성이 가공되는 경우 사용하지 않음.
- 속성의 선택도가 낮을 때 유리함(속성의 모든 값이 다른 경우).

인덱스의 생성 문법

```
CREATE [REVERSE] [UNIQUE] INDEX [인덱스이름]
ON 테이블이름 (컬럼 [ASC | DESC] [{, 컬럼  [ASC | DESC]} ...])[;]
```

4.4 인덱스의 생성

질의 4-24 Book 테이블의 bookname 열을 대상으로 비 클러스터 인덱스 ix_Book을

```
CREATE INDEX ix_Book ON Book (bookname);
```

```
Index IX_BOOK이(가) 생성되었습니다.
```

질의 4-25 Book 테이블의 publisher, price 열을 대상으로 인덱스 ix_Book2를 생성하시오

```
CREATE INDEX ix_Book2 ON Book (publisher, price);
```

```
Index IX_BOOK2이(가) 생성되었습니다.
```

4.4 인덱스의 생성

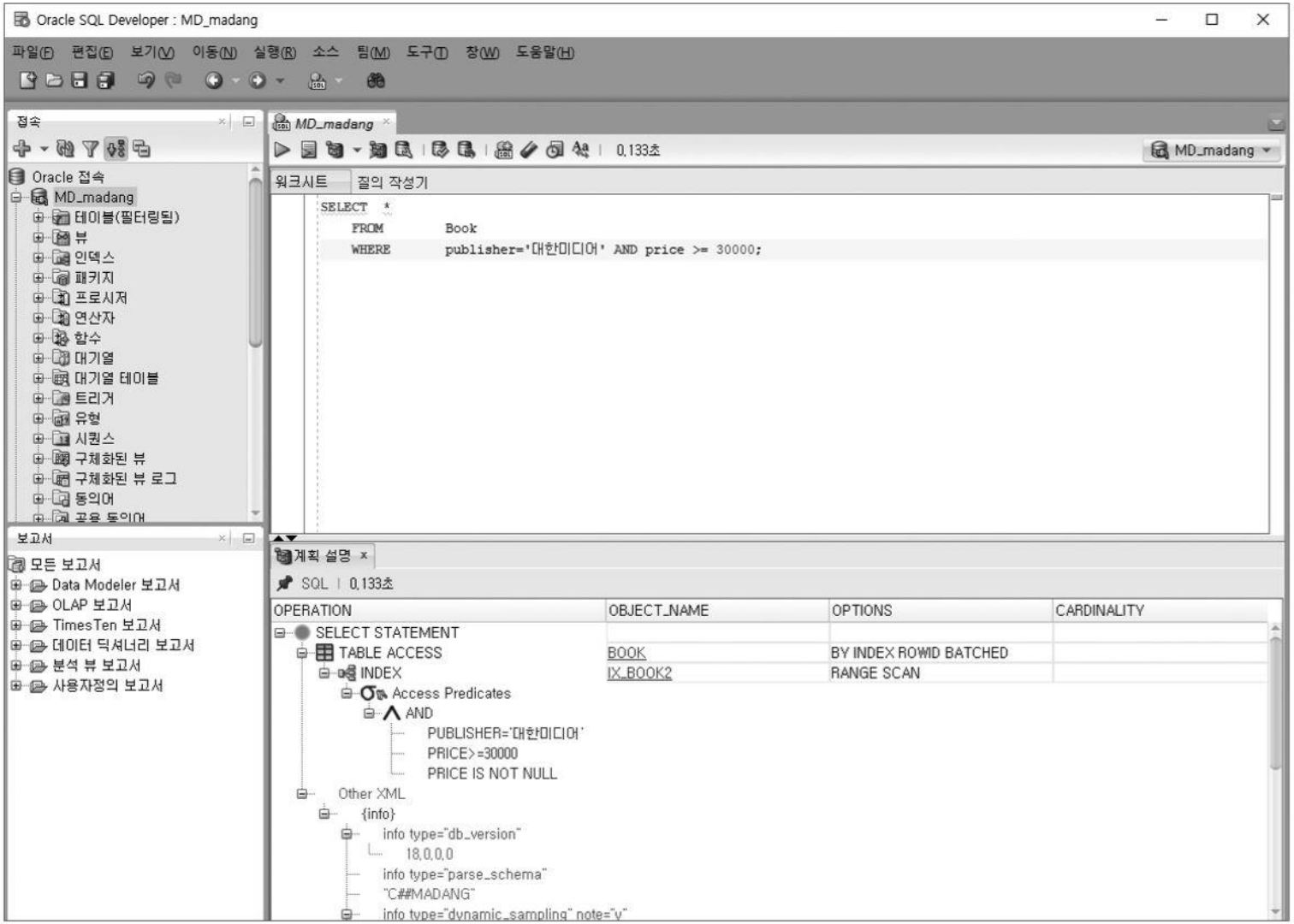


그림 4-15 생성된 인덱스 확인 및 실행 계획

4.4 인덱스의 생성

- **인덱스의 재구성은 ALTER INDEX 명령을 사용함.**
  - 생성 문법

```
ALTER [REVERSE] [UNIQUE] INDEX 인덱스이름
[ON {ONLY} 테이블이름 (컬럼이름 [{, 컬럼이름 } …)] REBUILD[;]
```

질의 4-26 인덱스 ix_Book을

```
ALTER INDEX ix_Book REBUILD;
```

```
Index IX_BOOK이(가) 변경되었습니다.
```

  - 삭제 문법

```
DROP      INDEX 인덱스이름[;]
```

질의 4-27 인덱스 ix_Book을 삭제하시오.

```
DROP INDEX ix_Book;
```

```
Index IX_BOOK이(가) 삭제되었습니다.
```

연습문제

16. **[마당서점 데이터베이스 인덱스] 마당서점 데이터베이스에서 다음 SQL 문을 수행하고 데이터베이스가 인덱스를 사용하는 과정을 확인하시오.**

(1) 다음 SQL 문을 수행해본다.

```
SELECT name FROM Customer WHERE name LIKE '박세리';
```

(2) 실행 계획을 살펴본다. 실행 계획은 [F10]키를 누른 후 [계획 설명]탭을 선택하면 표시된다.

(3) Customer 테이블에 name으로 인덱스를 생성하시오. 생성 후 (1)번의 질의를 다시 수행하고 실행 계획을 살펴보시오.

(4) 같은 질의에 대한 두 가지 실행 계획을 비교해보시오.

(5) (3)번에서 생성한 인덱스를 삭제하시오.

요약

1. 내장 함수
2. 부속질의
3. 뷰
4. 인덱스
5. B-tree
6. 오라클 인덱스의 종류