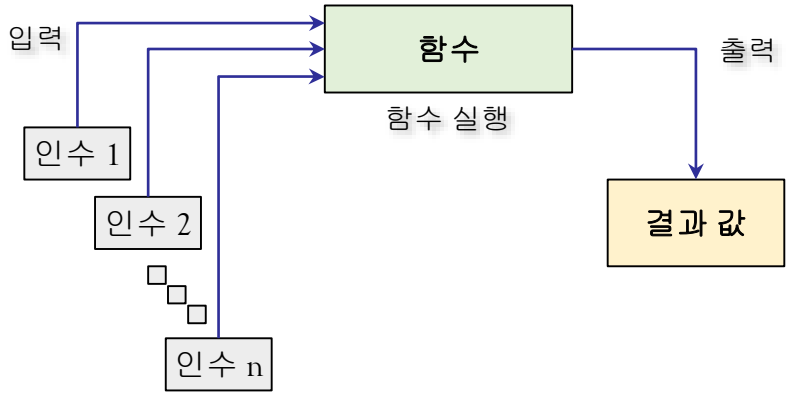
Chapter 3

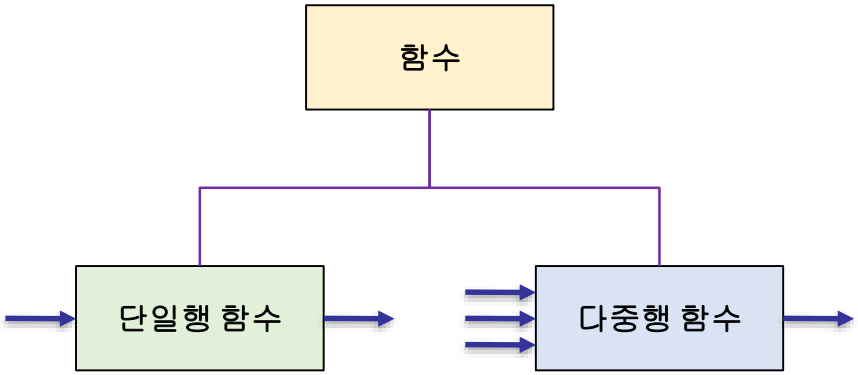
함수

문자 조작 함수
숫자 함수
날짜 함수
변환 함수
집합 연산자
분석 함수

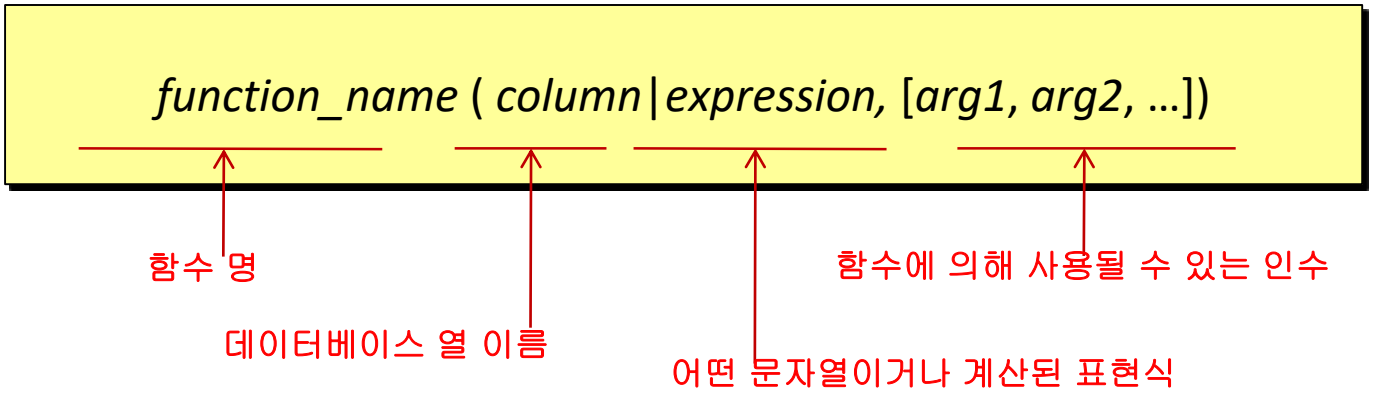
- 함수는 SQL의 아주 강력한 특징이며 다음을 위해서 사용할 수 있습니다.
  - 데이터 계산 수행
  - 개별적인 데이터 항목 수정
  - 행의 그룹에 대해 결과 조작
  - 출력을 위한 날짜와 숫자 형식 설정
  - 열의 데이터타입 변환



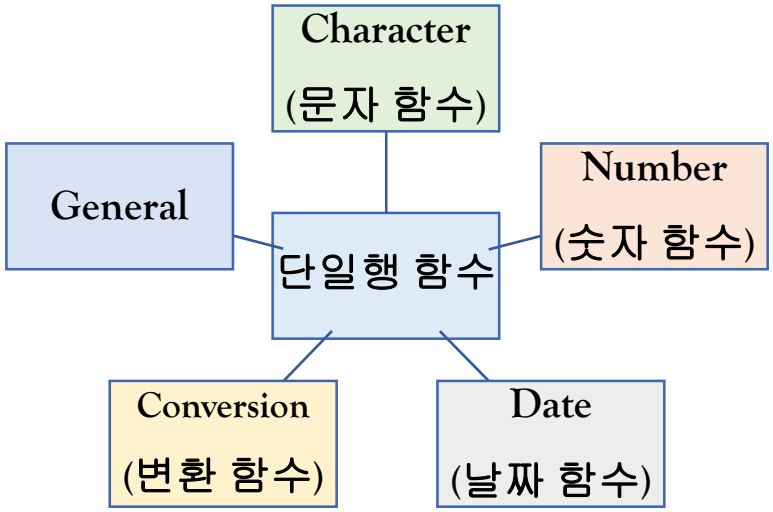
- 단일 행 함수
  - 이러한 함수는 오직 단일 행에서만 적용 가능하고 행별로 하나의 결과를 리턴합니다.
  - 단일 행 함수에는 여러 유형들이 있습니다. 본 과정은 아래에 나열된 것들을 다룹니다:
    - 문자
    - 숫자
    - 날짜
    - 변환
- 다중 행 함수
  - 이러한 함수는 복수의 행을 조작하여 행의 그룹당 하나의 결과를 리턴합니다.



- 데이터 값을 조작합니다.
- 인수(argument)를 받고 하나의 결과를 리턴합니다.
- 리턴될 각각의 행에 적용됩니다.
- 행별로 하나의 결과를 리턴합니다.
- 데이터타입을 수정할 수 있습니다.
- 중첩(nested)될 수 있습니다.



- 단일 행 함수 종류에는 문자 함수, 숫자 함수, 날짜 함수, 변환 함수 등이 있습니다.
- 문자 함수는 문자 입력을 받고 문자와 숫자 값 모두를 리턴할 수 있습니다.
  숫자 함수는 숫자 입력을 받고 숫자 값을 리턴합니다.
  날짜 함수는 날짜 데이터타입의 값에 대해 수행합니다.
  숫자를 리턴하는 MONTHS_BETWEEN 함수를 제외한 모든 날짜 함수는 날짜 데이터타입의 값을 리턴합니다.
  변환 함수는 어떤 데이터타입의 값을 다른 데이터타입의 값으로 변환합니다.
  그 외 함수에는 NVL, NVL2, LNNVL, NULLIF, COALESCE, DECODE 등이 있습니다.



Chapter 3

함수
문자 조작 함수
숫자 함수
날짜 함수
변환 함수
집합 연산자
분석 함수

함수	설명	결과
initcap('javaspecialist')	첫 문자만 대문자로, 나머지는 소문자로 출력합니다.	Javaspecialist
lower('JavaSpecialist')	모두 소문자로 출력합니다.	javaspecialist
upper('JavaSpecialist')	모두 대문자로 출력합니다.	JAVASPECIALIST
length('JavaSpecialist')	문자열의 길이(문자의 수)를 출력합니다.	14
length('자바전문가그룹')	문자열의 길이(문자의 수)를 출력합니다.	7
lengthb('자바전문가그룹')	바이트 수를 출력합니다.	21
concat('Java', 'Specialist')	||와 동일한 의미입니다. 두 문자열을 연결합니다.	JavaSpecialist
substr('JavaSpecialist', 5, 7)	부분 문자열을 출력합니다. 예는 5번째 문자부터 7문자	Special
substr('자바전문가그룹', 3, 3)	부분 문자열을 출력합니다. 예는 3번째 문자부터 3문자	전문가
substrb('자바전문가그룹', 7, 9)	바이트를 기준으로 부분 문자열을 출력합니다. 예는 7바이트부터 9바이트	전문가
instr('JavaSpecialist', 'S')	찾는 문자의 위치를 반환합니다.	5
instr('JavaSpecialist', 'b')	찾는 문자의 위치를 반환합니다. 해당 문자를 차지 못하면 0을 반환합니다.	0
instr('자바전문가그룹', '전')	찾는 문자의 위치를 반환합니다.	3
instrb('자바전문가그룹', '전')	찾는 문자열의 위치를 바이트 수로 반환합니다.	7
lpad(17600, 10, '*')	주어진 자릿수만큼 왼쪽(lpad)에 채웁니다. 예는 10자리로 출력하고 남은 자리는 왼쪽에 “*”로 채웁니다.	*****17600
rpad('Java', 10, '-')	주어진 자릿수만큼 오른쪽(rpad)에 채웁니다. 예는 10자리로 출력하고 남은 자리는 오른쪽에 “-”로 채웁니다.	Java------
ltrim('JavaSpecialist', 'Java')	첫 번째 문자열의 왼쪽부터 두 번째 문자열을 지웁니다. 예는 JavaSpecialist에서 왼쪽 Java를 지웁니다.	Specialist
ltrim(' JavaSpecialist')	왼쪽 공백을 제거해줍니다.	JavaSpecialist
trim(' JavaSpecialist ')	양쪽 공백을 제거해준다.	JavaSpecialist
replace('JavaSpecialist', 'Java', 'BigData')	주어진 문자열에서 두 번째 인수의 문자열을 세 번째 인수의 문자열로 변경합니다.	BigDataSpecialist
replace('Java Specialist', ' ', '')	주어진 문자열에 포함된 공백 문자를 제거합니다.	JavaSpecialist

```
SQL> SELECT last_name, LOWER(last_name), INITCAP(last_name), UPPER(last_name)
  2  FROM   employees;
```

	LAST_NAME	LOWER(LAST_NAME)	INITCAP(LAST_NAME)	UPPER(LAST_NAME)
1	Abel	abel	Abel	ABEL
2	Ande	ande	Ande	ANDE
3	Atkinson	atkinson	Atkinson	ATKINSON
4	Austin	austin	Austin	AUSTIN
5	Baer	baer	Baer	BAER

데이터의 대/소문자 구분이 명확하지 않을 경우 WHERE 조건절에 LOWER 또는 UPPER 함수가 유용하게 사용될 수 있습니다.

```
SQL> SELECT last_name, LOWER(last_name), INITCAP(last_name), UPPER(last_name)
  2  FROM   employees
  3  WHERE  LOWER(last_name)='austin';
```

	LAST_NAME	LOWER(LAST_NAME)	INITCAP(LAST_NAME)	UPPER(LAST_NAME)
1	Austin	austin	Austin	AUSTIN

LENGTH는 문자열의 길이를 반환합니다. INSTR은 주어진 문자의 위치를 반환합니다. 다음 SQL 코드는 FIRST_NAME의 길이와 FIRST_NAME에서 문자 'a'의 위치를 반환합니다.

```
SQL> SELECT first_name, LENGTH(first_name), INSTR(first_name, 'a')
  2  FROM   employees;
```

	FIRST_NAME	LENGTH(FIRST_NAME)	INSTR(FIRST_NAME,'A')
1	Ellen	5	0
2	Sundar	6	5
3	Mozhe	5	0
4	David	5	2
5	Hermann	7	5
6	Shelli	6	0
7	Amit	4	0
8	Elizabeth	9	5
9	Sarah	5	2
10	David	5	2

...

SUBSTR은 주어진 문자열에서 주어진 시작 위치에서부터 지정한 개수만큼 부분 문자열을 반환합니다. CONCAT 함수는 두 문자열을 연결합니다.

```
SQL> SELECT first_name, SUBSTR(first_name, 1, 3), CONCAT(first_name, last_name)
  2  FROM   employees;
```

	FIRST_NAME	SUBSTR(FIRST_NAME,1,3)	CONCAT(FIRST_NAME,LAST_NAME)
1	Ellen	Ell	EllenAbel
2	Sundar	Sun	SundarAnde
3	Mozhe	Moz	MozheAtkinson
4	David	Dav	DavidAustin
5	Hermann	Her	HermannBaer
6	Shelli	She	ShelliBaida
7	Amit	Ami	AmitBanda
8	Elizabeth	Eli	ElizabethBates
9	Sarah	Sar	SarahBell
10	David	Dav	DavidBernstein

...

LPAD는 왼쪽에, RPAD 오른쪽에 남은 부분을 주어진 문자로 채웁니다. LPAD 함수는 정의된 문장의 왼쪽 나머지 공간을 지정한 문자로 채웁니다. 오른쪽 공간에 채우는 함수에는 RPAD가 있습니다.

```
SQL> SELECT
  2      RPAD(first_name, 10, '-') AS name,
  3      LPAD(salary, 10, '*') AS sal
  4  FROM employees;
```

	NAME	SAL
1	Steven----	*****24000
2	Neena-----	*****17000
3	Lex-------	*****17000
4	Alexander-	******9000
5	Bruce-----	******6000
6	David-----	******4800
7	Valli-----	******4800
8	Diana-----	******4200
9	Nancy-----	*****12008
10	Daniel----	******9000

…

LTRIM 함수는 정의된 문자열의 왼쪽부터 지정된 단어가 발견되면 제거합니다. 반대로 오른쪽 단어를 제거하기 위해서는 RTRIM 함수를 사용합니다. 제거하는 문자를 지정하지 않으면 공백을 제거합니다.

```
SQL> SELECT LTRIM('JavaSpecialist', 'Java')
  2  FROM dual;
SQL> SELECT LTRIM(' JavaSpecialist')
  2  FROM dual;
SQL> SELECT TRIM(' JavaSpecialist ')
  2  FROM dual;
```

	LTRIM('JAVASPECIALIST','JAVA')
1	Specialist

	LTRIM('JAVASPECIALIST')
1	JavaSpecialist

	TRIM('JAVASPECIALIST')
1	JavaSpecialist

REPLACE 함수는 정의된 문자열에서 지정한 문자열을 새로운 문자열로 대체합니다. 이를 이용하면 정의된 문자열에 포함되어 있는 공백을 제거할 수 있습니다. TRANSLATE 함수는 정의된 문자열을 1대1로 대응시킵니다.

```
SQL> SELECT REPLACE('JavaSpecialist', 'Java', 'BigData') FROM dual;
SQL> SELECT REPLACE('Java Specialist', ' ', '') FROM dual;
SQL> SELECT TRANSLATE('javaspecialist', 'abcdefghijklmnopqrstuvwxyz',
     'defghijklmnopqrstuvwxyzabc') FROM dual;
```

	REPLACE('JAVASPECIALIST','JAVA','BIGDATA')
1	BigDataSpecialist

	REPLACE('JAVASPECIALIST','','')
1	JavaSpecialist

	TRANSLATE('JA'
1	mdydvshfldolvw

문제 1. EMPLOYEES 테이블에서 JOB_ID가 it_prog인 사원의 이름(first_name)과 급여(salary)를 출력하세요.

조건 1) 비교하기 위한 값은 소문자로 입력해야 합니다.(힌트 : lower 이용)

조건 2) 이름은 앞 3문자까지 출력하고 나머지는 *로 출력합니다.
이 열의 열 별칭은 name입니다.(힌트 : rpad와 substr 또는 substr 그리고 length 이용)

조건 3) 급여는 전체 10자리로 출력하되 나머지 자리는 *로 출력합니다.
이 열의 열 별칭은 salary입니다.(힌트 : lpad 이용)

문제 1.

EMPLOYEES 테이블 에서 이름, 입사일자 컬럼으로 변경해서 이름순으로 오름차순 출력 합니다.

조건 1) 이름 컬럼은 first_name, last_name을 붙여서 출력합니다.

조건 2) 입사일자 컬럼은 xx/xx/xx로 저장되어 있습니다. xxxxxx형태로 변경해서 출력합니다.

문제 2.

EMPLOYEES 테이블 에서 phone_numbe컬럼은 ###.###.####형태로 저장되어 있다

여기서 처음 세 자리 숫자 대신 서울 지역번호 (02)를 붙여 전화 번호를 출력하도록 쿼리를 작성하세요

문제 3.

EMPLOYEES 테이블에서 JOB_ID가 it_prog인 사원의 이름(first_name)과 급여(salary)를 출력하세요.

조건 1) 비교하기 위한 값은 소문자로 입력해야 합니다.(힌트 : lower 이용)

조건 2) 이름은 앞 3문자까지 출력하고 나머지는 *로 출력합니다.
이 열의 열 별칭은 name입니다.(힌트 : rpad와 substr 또는 substr 그리고 length 이용)

조건 3) 급여는 전체 10자리로 출력하되 나머지 자리는 *로 출력합니다.
이 열의 열 별칭은 salary입니다.(힌트 : lpad 이용)

Chapter 3

함수
문자 조작 함수
숫자 함수
날짜 함수
변환 함수
집합 연산자
분석 함수

함수명	설명
ROUND(column|expression, n)	열, 표현식 또는 값을 소수점 n 자리까지 반올림합니다. n이 생략되면 소수점이 없어집니다. n이 음수이면 소수점의 왼쪽 자리 수만큼 반올림 됩니다.
TRUNC(column|expression, n)	열, 표현식 또는 값을 소수점 n 자리까지 절삭합니다. n이 생략되면 소수점이 없어집니다. n이 음수이면, 소수점의 왼쪽 자리 수만큼 절삭됩니다.
ABS(column|expression)	열, 표현식 또는 값의 절대값을 반환하는 함수입니다.
SIGN(column|expression)	열, 표현식 또는 값이 양수인지 음수인지를 반환합니다. 0보다 클 경우 1을, 작을 경우 -1을, 그리고 0일 경우에는 0을 리턴합니다.
CEIL(column|expression)	열, 표현식 또는 값과 같거나 큰 가장 작은 정수를 리턴합니다.
FLOOR(column|expression)	열, 표현식 또는 값보다 작거나 같은 가장 큰 정수를 리턴합니다.
REMAINDER(m, n)	MOD함수와 마찬가지로 m을 n으로 나눈 나머지를 리턴합니다. MOD와 다른 점은 REMAINDER 함수는 NUMBER타입뿐만 아니라 BINARY_DOUBLE 타입도 올 수 있습니다.
POWER(m, n)	m의 n제곱값을 리턴합니다.
SQRT(n)	n의 제곱근 값을 리턴합니다.
SIN(radian), COS(radian) TAN(radian)	radian값을 이용하여 sine, cosine, tangent 값을 계산합니다. radian = degree * (3.141592/180)
SINH(radian), COSH(radian), TANH(radian)	radian 값을 이용하여 Hyperbolic sine, Hyperbolic cosine, Hyperbolic tangent값을 반환합니다.
EXP(n)	지수값 e를 밑으로 하는 e의 n승 값을 리턴합니다. exp(1) 는 2.71828183입니다.
LN(n)	자연 log값을 리턴합니다. 밑수 e 지수 n
LOG(m,n)	m을 기수로 이용하여 n의 대수를 리턴합니다.
MOD(m, n)	m을 n으로 나눈 나머지를 리턴합니다. 자바 언어의 m%n과 동일합니다.

ROUND 함수는 열, 표현식 또는 값을 소수점 n 자리로 반올림합니다. 두 번째 인자가 0이거나 생략되면, 값은 소수점 위치가 0으로 반올림됩니다. 두 번째 인자가 2이면, 값은 소수점 아래 두 번째 위치로 반올림됩니다. 반대로, 두 번째 인자가 -2이면, 값은 소수점 좌측의 두 번째 위치로 반올림됩니다.

다음 구문은 ROUND 함수를 사용하고 있습니다.

```
SQL> SELECT ROUND(45.923,2), ROUND(45.923,0), ROUND(45.923,-1)
  2  FROM   DUAL;
```

	ROUND(45.923,2)	ROUND(45.923,0)	ROUND(45.923,-1)
1	45.92	46	50

ROUND 함수는 날짜 함수에 함께 사용될 수도 있습니다.

TRUNC 함수는 열, 표현식 또는 값을 소수점 n 자리로 절삭합니다. TRUNC 함수는 ROUND 함수와 유사한 인수로 수행합니다. 두 번째 인자가 Ø이거나 생략되면, 값은 소수점 위치를 Ø으로 절삭합니다. 두 번째 인수가 2이면, 값은 두 개의 소수점 아래 위치로 절삭합니다. 반대로 두 번째 인자가 -2이면, 값은 소수점 좌측부터 두 자리까지 절삭합니다.

다음 구문은 TRUNC 함수를 사용하고 있습니다.

```
SQL> SELECT TRUNC(45.923,2), TRUNC(45.923), TRUNC(45.923,-1)
  2  FROM   DUAL;
```

	TRUNC(45.923,2)	TRUNC(45.923)	TRUNC(45.923,-1)
1	45.92	45	40

ROUND 함수처럼 TRUNC 함수도 날짜 함수와 함께 사용될 수 있습니다.

Chapter 3

함수
문자 조작 함수
숫자 함수
날짜 함수
변환 함수
집합 연산자
분석 함수

- 오라클은 세기, 년, 월, 일, 시간, 분, 초의 내부 숫자형식으로 날짜를 저장합니다.
- 디폴트 날짜 형식은 'DD-MON-YY'입니다. 시스템에 따라 'YY/MM/DD'가 될 수 있습니다.
- SYSDATE는 현재의 날짜를 반환하는 함수입니다.
- SYSTIMESTAMP는 현재의 날짜와 시간을 반환하는 함수입니다.
- DUAL은 SYSDATE를 보기 위해 사용된 dummy 테이블입니다.

오라클은 세기, 년, 월, 일, 시간, 분, 초의 내부 숫자형식으로 날짜를 저장합니다.
날짜에 대한 디폴트 출력과 입력 형식은 'DD-MON-YY'입니다. 적절한 오라클 날짜의 범위는 January 1, 4712 B.C. 와 December 31, 9999 A.D. 사이입니다.

SYSDATE는 현재의 날짜를 반환하는 함수입니다. 다른 열 이름을 사용하듯이 SYSDATE를 사용할 수 있습니다. 예를 들면, 테이블로부터 SYSDATE를 선택하여 현재 날짜를 출력할 수 있습니다. DUAL[18] 이라는 더미 테이블로부터 SYSDATE를 선택하는 것이 관례입니다.

다음 구문은 현재 날짜를 출력합니다.

```
SQL> SELECT     SYSDATE
  2  FROM       DUAL;
```

	SYSDATE
1	17/04/04

SYSDATE는 현재 날짜를 출력합니다. 현재 날짜와 함께 시간에 대한 정보까지 출력하고 싶다면 아래 구문처럼 SYSTIMESTAMP를 이용합니다.

```
SQL> SELECT     SYSTIMESTAMP
  2  FROM       DUAL;
```

	SYSTIMESTAMP
1	17/04/04 13:34:20.626000000 +09:00

- 날짜에서 숫자를 더하거나 빼 날짜 결과를 반환합니다.
- 날짜 사이의 일(DAY) 수를 알기 위해서 두 개의 날짜를 뺍니다.
- 시간을 24로 나누어 날짜에 더합니다.

표 6. 날짜 연산

작업	결과	설명
날짜 + 숫자	날짜	일(DAY) 수를 날짜에 더합니다.
날짜 - 숫자	날짜	날짜에서 일(DAY) 수를 뺍니다.
날짜 - 날짜	일수	어떤 날짜에서 다른 날짜를 뺍니다.
날짜 + 숫자/24	날짜	시간을 날짜에 더합니다.

데이터베이스는 날짜를 숫자로 저장하므로, 더하기와 빼기 같은 산술 연산자를 사용하여 계산을 수행할 수 있습니다. 날짜뿐만 아니라 숫자 상수를 더하거나 뺄 수도 있습니다.

```
SQL> SELECT first_name, (SYSDATE - hire_date)/7 AS "Weeks"
  2  FROM   employees
  3  WHERE  department_id=60;
```

위의 예는 부서 60에 속하는 모든 사원에 대해서 이름과 근무한 주의 합계를 출력합니다. 현재 날짜(SYSDATE[19])에서 사원이 입사한 날짜를 빼고 근무한 주의 합계를 구하기 위해서 7로 나눕니다.

	FIRST_NAME	Weeks
1	Alexander	587.112194113756613756613756613756613757
2	Bruce	515.255051256613756613756613756613756614
3	David	614.540765542328042328042328042328042328
4	Valli	582.397908399470899470899470899470899471
5	Diana	529.969336970899470899470899470899470899

```
ROUND(date[, 'fmt'])
TRUNC(date[, 'fmt'])
```

ROUND 함수는 포맷 모델 fmt에 명시된 단위에 대해 반올림한 date를 반환합니다. fmt가 생략되면, date를 가장 가까운 날짜로 반올림 됩니다.

TRUNC 함수는 포맷 모델 fmt에 명시된 단위에 대해 절삭한 date를 반환합니다. fmt가 생략되면, date는 가장 가까운 날짜로 절삭됩니다.[20)]

```
SQL> SELECT SYSDATE, ROUND(SYSDATE), TRUNC(SYSDATE)
  2  FROM   dual;
```

	SYSDATE	ROUND(SYSDATE)	TRUNC(SYSDATE)
1	17/04/04	17/04/05	17/04/04

ROUND와 TRUNC 함수에 반올림하거나 절삭 할 단위를 지정할 수 있습니다.

```
SQL> SELECT TRUNC(SYSDATE, 'Month')
  2  FROM   dual;
```

	TRUNC(SYSDATE,'MONTH')
1	17/04/01

```
SQL> SELECT TRUNC(SYSDATE, 'Year')
  2  FROM   dual;
```

	TRUNC(SYSDATE,'YEAR')
1	17/01/01

```
SQL> SELECT ROUND(TO_DATE('17/03/16'), 'Month')
  2  FROM   dual;
```

	ROUND(TO_DATE('17/03/16'),'MONTH')
1	17/04/01

```
SQL> SELECT TRUNC(TO_DATE('17/03/16'), 'Month')
  2  FROM   dual;
```

	TRUNC(TO_DATE('17/03/16'),'MONTH')
1	17/03/01

Chapter 3

함수
문자 조작 함수
숫자 함수
날짜 함수
변환 함수
집합 연산자
분석 함수

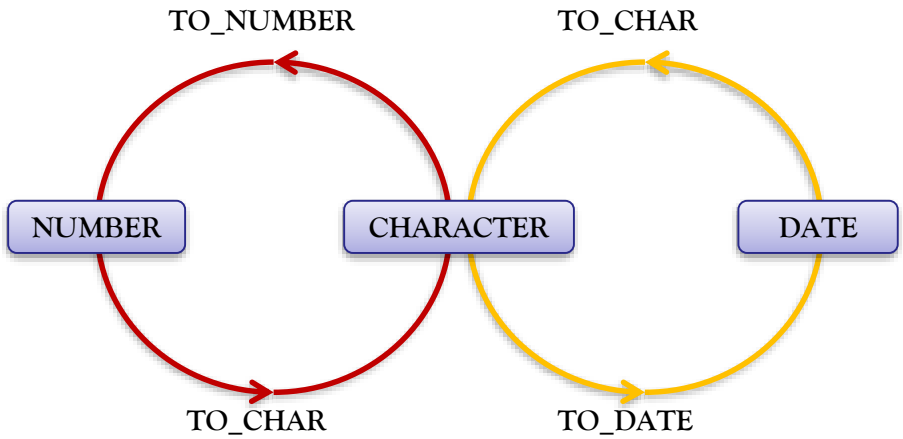
- 값 할당(assignment) 시, 오라클은 자동으로 다음을 변환할 수 있습니다.

표 8. 암시적 형 변환

From	To	비고
VARCHAR2 또는 CHAR	NUMBER	표현식 계산의 경우
VARCHAR2 또는 CHAR	DATE	표현식 계산의 경우
NUMBER	VARCHAR2	
DATE	VARCHAR2	

값 할당(assignment) 시, 오라클 서버는 다음을 자동으로 변환할 수 있습니다.

- VARCHAR2 또는 CHAR를 NUMBER로
- VARCHAR2 또는 CHAR를 DATE로
- NUMBER를 VARCHAR2로
- DATE를 VARCHAR2로



변환 함수	설명
TO_CHAR(number[, 'fmt'])	숫자를 포맷 모델 fmt를 사용하여 VARCHAR2 문자 스트링으로 변환합니다.
TO_CHAR(date[, 'fmt'])	날짜를 포맷 모델 fmt를 사용하여 VARCHAR2 문자 스트링으로 변환합니다.
TO_NUMBER(char[, 'fmt'])	숫자를 포함하는 문자 스트링을 숫자로 변환합니다.
TO_DATE(char[, 'fmt'])	날짜를 나타내는 문자 스트링을 명시된 fmt에 따라서 날짜 값으로 변환합니다.(fmt 가 생략되면, DD-MON-YY 형식입니다.)

날짜를 문자로 변환하기 위해서 TO_CHAR 함수를 사용합니다.

날짜 형식을 지정하기 위해서 fmt(format)를 지정하며, fmt는 단일 인용부호로 둘러싸여 있어야 합니다.

어떤 타당한 날짜 형식 요소도 포함할 수 있습니다.

추가된 공백을 제거하거나 앞부분의 Ø을 없애기 위해 fm(fill mode)요소를 사용합니다.

```
TO_CHAR(date, 'fmt')
```

다음은 포맷 모델에 대한 지침입니다.

- 포맷 모델은 단일 인용부호로 둘러 싸여 있어야 하고 대소문자를 구분합니다.
- 포맷 모델은 어떤 타당한 날짜 형식도 포함할 수 있습니다. 포맷 모델의 날짜 값은 콤마로 구분합니다.
- 결과의 일 명과 월 명은 자동적으로 공백이 추가됩니다.
- 추가된 공백을 제거하거나 앞부분의 Ø을 없애기 위해서 fm[18] 요소를 사용합니다.
- SQL Plus COLUMN 명령어로 문자 필드 결과의 출력 폭의 크기를 조절할 수 있습니다.
- 결과 열의 길이는 디폴트로 80자 입니다.

```
SQL> SELECT first_name, TO_CHAR(hire_date, 'MM/YY') AS Month_Hired
  2  FROM   employees
  3  WHERE  first_name='Steven';
```

	FIRST_NAME	HIREDMONTH
1	Steven	06/03
2	Steven	03/08

요 소	설 명
SCC or CC	세기; BC 날짜에는 -를 붙입니다.
YYYY or SYYYY	년; BC 날짜에는 -를 붙입니다.
YYY or YY or Y	년의 마지막 3, 2 또는 1자리 수
Y,YYY	콤마가 있는 년
IYYY, IYY, IY, I	ISO 표준에 바탕을 둔 4, 3, 2 또는 1자리 수
RR	Y2K 문제와 관련 있는 년도를 나타내는 형식 문자
SYEAR or YEAR	문자로 표현된 년; BC 날짜에는 -를 붙입니다.
BC or AD	BC/AD 지시자
B.C. or A.D.	. 이 있는 BC/AD 지시자
Q	년의 4분의 1
MM	두 자리 값의 월
MONTH	9자리를 위해 공백을 추가한 월 이름
MON	세 자리의 약어로 된 월 이름
RM	로마 숫자 월
WW or W	년이나 월의 주
DDD or DD or D	년, 월 또는 주의 일
DAY	9자리를 위해 공백을 추가한 요일 이름
DY	세 자리 약어로 된 요일 이름
J	Julian day; BC 4713년 12월 31일 이후의 요일 수

요 소	설 명
AM or PM	정오 지시자
A.M. or P.M.	. 이 있는 정오 지시자
HH or HH12 or HH24	하루 중 시간 또는 시간(1-12) 또는 시간(0-23)
MI	분 (0-59)
SS	초 (0-59)
SSSSS	자정 이후의 초 (0-86399)

다음 표는 숫자 출력에 영향을 주는 접미사입니다.

요 소	설 명
TH	서수 (예, DDTH à 4TH)
SP	명시한 수 (예, DDSP à FOUR)
SPTH or THSP	명시한 서수 (예, DDSPTH à FOURTH)

다음 표의 요소는 사용한 문자가 결과에 다시 나타납니다.

요 소	설 명
/ . ,	사용 문자가 결과에 다시 나타납니다.
"of the"	인용 부호내의 스트링이 결과에 다시 나타납니다.

```
SQL> SELECT first_name,
  2    TO_CHAR(hire_date, 'YYYY"년" MM"월" DD"일"')  HIREDATE
  3  FROM   employees;
```

위의 SQL 문장은 모든 사원에 대해서 이름과 입사일을 출력 합니다. 입사일은 "2003년 06월 17일"처럼 나타납니다. 위의 예에서 포맷을 'fmYYYY"년" MM"월" DD"일"'으로 지정했을 경우에는 날짜 또는 월에 0이 제거됩니다.

	FIRST_NAME	HIREDATE
1	Steven	2003년 06월 17일
2	Neena	2005년 09월 21일
3	Lex	2001년 01월 13일
4	Alexander	2006년 01월 03일
5	Bruce	2007년 05월 21일
6	David	2005년 06월 25일
7	Valli	2006년 02월 05일
8	Diana	2007년 02월 07일
9	Nancy	2002년 08월 17일
10	Daniel	2002년 08월 16일

...

다음 구문은 날짜 형식이 "Seventeenth of June 2003 12:00:00 AM"처럼 나타나도록 위의 예를 수정합니다.

```
SQL> SELECT first_name,
  2    TO_CHAR(hire_date,
  3            'fmDdspth "of" Month YYYY fmHH:MI:SS AM',
  4            'NLS_DATE_LANGUAGE=english') AS HIREDATE
  5  FROM   employees;
```

TO_CHAR(*number*, '*fmt*')

요소	설 명	예	결과
9	숫자 위치(9의 수는 출력 폭을 결정합니다.)	999999	1234
0	앞에 Ø을 출력 합니다.	099999	001234
$	달러 기호를 나타냅니다.	$999999	$1234
L	지역 화폐 기호를 사용합니다.	L999999	₩1234
.	명시한 위치에 소수점을 출력합니다.	999999.99	1234.00
,	명시한 위치에 콤마를 출력합니다.	999,999	1,234
PR	음수를 괄호로 묶습니다.	999999PR	<1234>
EEEE	과학적인 부호표기(형식은 4개의 E를 명시해야 합니다.)	99.999EEEE	1.234E+03
V	10을 n 번 곱합니다. (n = V 뒤의 9의 수)	9999V99	123400
B	Ø값을 Ø이 아닌 공백으로 출력 합니다.	B9999.99	1234.00

```
SQL> SELECT first_name, last_name, TO_CHAR(salary, '$999,999') SALARY
  2  FROM   employees
  3  WHERE  first_name='David';
```

	FIRST_NAME	LAST_NAME	SALARY
1	David	Austin	$4,800
2	David	Bernstein	$9,500
3	David	Lee	$6,800

다음 구문은 주어진 자릿수에 숫자를 모두 표시할 수 없기 때문에 값을 모두 #으로 출력 합니다.

```
SQL> SELECT TO_CHAR(2000000, '$999,999') SALARY
  2  FROM   dual;
```

	SALARY
1	#########

다음 구문은 소수점 자리수로 반올림한 값을 출력 합니다.

```
SQL> SELECT first_name, last_name,
  2         salary*0.123456 SALARY1,
  3         TO_CHAR(salary*0.123456, '$999,999.99') SALARY2
  4  FROM   employees
  5  WHERE  first_name='David';
```

	FIRST_NAME	LAST_NAME	SALARY1	SALARY2
1	David	Austin	592.5888	$592.59
2	David	Bernstein	1172.832	$1,172.83
3	David	Lee	839.5008	$839.50

TO_NUMBER 함수를 사용하여 문자 스트링을 숫자형식으로 변환합니다.

```
TO_NUMBER(char, 'fmt')
```

오식 숫자로만 이루어진 문자 스트링은 숫자 형식으로 자동 형변환 됩니다. 그러나 $ 또는 , 등을 포함한 문자스트링은 fmt를 지정해서 형 변환해야 합니다.

예를 들면 다음 구문을 실행 시 오류가 발생합니다. 문자열 '%5,500.00'은 숫자 5500으로 자동 형변환 되지 않습니다.

```
SQL> SELECT '$5,500.00' - 4000 FROM dual;
```

```
ORA-01722: invalid number
01722. 00000 -  "invalid number"
*Cause:    The specified number was invalid.
*Action:   Specify a valid number.
```

```
SQL> SELECT to_number('$5,500.00', '$99,999.99') - 4000
  2  FROM dual;
```

	TO_NUMBER('$5,500.00','$99,999.99')-4000
1	1500

TO_DATE 함수를 사용하여 문자 스트링을 날짜 형식으로 변환합니다.

```
TO_DATE(char, 'fmt')
```

문자 스트링을 숫자나 날짜 형식으로 변환하기를 원할 수도 있습니다. 이런 일을 수행하기 위해 TO_NUMBER 나 TO_DATE 함수를 사용합니다. 여러분이 선택할 형식 모델은 앞의 형식 모델 요소를 바탕으로 합니다.

다음 구문은 2003년 6월 17일에 입사한 사원의 이름과 입사일을 출력 합니다.

```
SQL> SELECT first_name, hire_date
  2  FROM   employees
  3  WHERE  hire_date=TO_DATE('2003/06/17', 'YYYY/MM/DD');
```

	FIRST_NAME	HIRE_DATE
1	Steven	03/06/17

Null 값을 실제 값으로 변환하기 위해서, NVL 함수 함수를 사용합니다. 사용될 수 있는 데이터타입은 날짜, 문자, 숫자입니다. *expr1*과 *expr2* 데이터타입은 일치해야 합니다.

```
NVL(expr1, expr2)
```

구문형식에서...
- *expr1* : Null을 포함할 수 있는 값이나 표현식입니다. Null이 아니면 해당 값을 출력합니다.
- *expr2* : Null 변환을 위한 목표(target) 값 입니다. expr1이 Null일 경우 출력됩니다.

다음 구문은 모든 사원의 급여를 보너스를 포함하여 출력 합니다.

```
SQL> SELECT first_name,
  2         salary + salary*NVL(commission_pct,0) AS ann_sal
  3  FROM   employees;
```

	FIRST_NAME	ANN_SAL
1	Steven	24000
2	Neena	17000
3	Lex	17000
4	Alexander	9000
5	Bruce	6000
6	David	4800

NVL2함수는 expr1의 값이 Null이 아닐 경우에는 expr2의 값을 반환 하고, Null일 경우에는 expr3의 값을 반환 합니다. expr1의 타입과 expr2, expr3의 타입은 같지 않아도 됩니다.

```
NVL2(expr1, expr2, expr3)
```

구문형식에서...

- *expr1* : Null을 포함할 수 있는 값이나 표현식입니다.
- *expr2* : Null 변환을 위한 목표(target) 값 입니다. expr1이 Null이 아닐 경우 출력됩니다.
- *expr3* : Null 변환을 위한 목표(target) 값 입니다. expr1이 Null일 경우 출력됩니다.

다음 구문은 모든 사원의 급여를 보너스를 포함하여 출력 합니다.

```
SQL> SELECT first_name,
  2    NVL2(commission_pct, salary+(salary*commission_pct), salary) ann_sal
  3  FROM employees;
```

	FIRST_NAME	ANN_SAL
1	Steven	24000
2	Neena	17000
3	Lex	17000
4	Alexander	9000
5	Bruce	6000

DECODE 함수는 여러 언어에서 사용하는 IF-THEN-ELSE 구문과 유사한 방법으로 표현식을 해독합니다. DECODE 함수는 expression을 해독한 후 각각의 *searchN*에 대해 비교합니다. 표현식의 결과가 *searchN*과 같으면 *resultN*이 반환됩니다. 기본값이 생략되면 검색값(*searchN*)이 결과 값(*resultN*)과 일치하지 않는 곳에 Null 값이 반환될 것입니다.

```
DECODE(column or expression, search1, result1
                           [, search2, result2, … , ]
                           [, default])
```

```
SQL> SELECT  job_id, salary,
  2          DECODE(job_id, 'IT_PROG',    salary*1.10,
  3                         'FI_MGR',     salary*1.15,
  4                         'FI_ACCOUNT', salary*1.20,
  5                                       salary)
  6          AS revised_salary
  7  FROM    employees;
```

	JOB_ID	SALARY	REVISED_SALARY
1	AD_PRES	24000	24000
2	AD_VP	17000	17000
3	AD_VP	17000	17000
4	IT_PROG	9000	9900
5	IT_PROG	6000	6600
6	IT_PROG	4800	5280
7	IT_PROG	4800	5280
8	IT_PROG	4200	4620
9	FI_MGR	12008	13809.2
10	FI_ACCOUNT	9000	10800

CASE ~ WHEN ~ THEN 구문은 자바언어의 if~else 문 또는 switch~case 문과 같은 기능을 합니다. CASE 절 뒤에 표현식이 있을 경우 WHEN 절에는 값이 오며, CASE 절 뒤에 아무것도 없을 경우 WHEN 절에 조건식이 옵니다.

```
CASE column or expression WHEN condition1 THEN result1
                          WHEN condition2 THEN result2
                                       ......
                          WHEN conditionN THEN resultN
                          ELSE result
END
```

```
SQL> SELECT job_id, salary,
  2    CASE job_id WHEN 'IT_PROG'    THEN salary*1.10
  3                WHEN 'FI_MGE'     THEN salary*1.15
  4                WHEN 'FI_ACCOUNT' THEN salary*1.20
  5           ELSE salary
  6    END AS REVISED_SALARY
  7  FROM   employees;
```

	JOB_ID	SALARY	REVISED_SALARY
1	AD_PRES	24000	24000
2	AD_VP	17000	17000
3	AD_VP	17000	17000

```
SQL> SELECT job_id, salary,
  2    CASE WHEN job_id='IT_PROG'    THEN salary*1.10
  3         WHEN job_id='FI_MGR'     THEN salary*1.15
  4         WHEN job_id='FI_ACCOUNT' THEN salary*1.20
  5         ELSE salary
  6    END AS revised_salary
  7  FROM   employees;
```

문제 1.
현재일자를 기준으로 EMPLOYEE테이블의 입사일자(hire_date)를 참조해서 근속년수가 10년 이상인 사원을 다음과 같은 형태의 결과를 출력하도록 쿼리를 작성해 보세요.

조건 1) 근속년수가 높은 사원 순서대로 결과가 나오도록 합니다

	사원번호	사원명	입사일자	근속년수
1	102	LexDe Haan	01/01/13	18
2	109	DanielFaviet	02/08/16	17
3	206	WilliamGietz	02/06/07	17
4	204	HermannBaer	02/06/07	17
5	205	ShelleyHiggins	02/06/07	17
6	108	NancyGreenberg	02/08/17	17
7	203	SusanMavris	02/06/07	17
8	100	StevenKing	03/06/17	16
9	114	DenRaphaely	02/12/07	16
10	115	AlexanderKhoo	03/05/18	16

문제 2.
EMPLOYEE 테이블의 manager_id컬럼을 확인하여 first_name, manager_id, 직급을 출력합니다.
100이라면 '사원',
120이라면 '주임'
121이라면 '대리'
122라면 '과장'
나머지는 '임원' 으로 출력합니다.

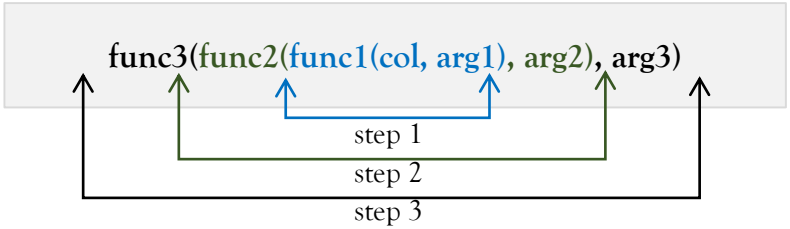
조건 1) manager_id가 50인 사람들을 대상으로만 조회합니다

문제 1. 1월부터 12월까지 각 월의 마지막 날짜를 출력하세요.

	1	2	3	4	5	6	7	8	9	10	11	12
1	31	28	31	30	31	30	31	31	30	31	30	31

정답 1.

```
SQL> SELECT
  2    TO_CHAR(LAST_DAY(TO_DATE('01', 'MM')), 'dd') AS "1",
  3    TO_CHAR(LAST_DAY(TO_DATE('02', 'MM')), 'dd') AS "2",
  4    TO_CHAR(LAST_DAY(TO_DATE('03', 'MM')), 'dd') AS "3",
  5    TO_CHAR(LAST_DAY(TO_DATE('04', 'MM')), 'dd') AS "4",
  6    TO_CHAR(LAST_DAY(TO_DATE('05', 'MM')), 'dd') AS "5",
  7    TO_CHAR(LAST_DAY(TO_DATE('06', 'MM')), 'dd') AS "6",
  8    TO_CHAR(LAST_DAY(TO_DATE('07', 'MM')), 'dd') AS "7",
  9    TO_CHAR(LAST_DAY(TO_DATE('08', 'MM')), 'dd') AS "8",
 10    TO_CHAR(LAST_DAY(TO_DATE('09', 'MM')), 'dd') AS "9",
 11    TO_CHAR(LAST_DAY(TO_DATE('10', 'MM')), 'dd') AS "10",
 12    TO_CHAR(LAST_DAY(TO_DATE('11', 'MM')), 'dd') AS "11",
 13    TO_CHAR(LAST_DAY(TO_DATE('12', 'MM')), 'dd') AS "12"
 14  FROM dual;
```



위 그림에서 func1 함수의 결과는 func2 함수의 인수로 사용됩니다. func2 함수의 결과는 다시 func3 함수의 인수로 사용됩니다. 함수들이 중첩 사용되었을 때는 가장 안쪽 함수부터 실행이 종료되어야 바깥 함수가 실행됩니다.

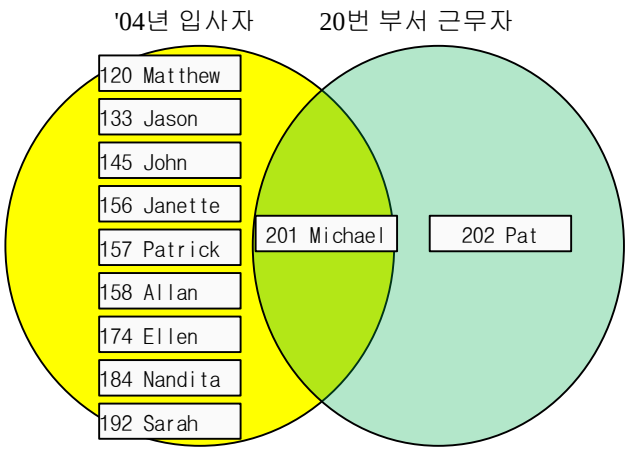
Chapter 3

함수
문자 조작 함수
숫자 함수
날짜 함수
변환 함수
집합 연산자
분석 함수

```
SQL> SELECT employee_id, first_name
  2    FROM   employees
  3    WHERE  hire_date LIKE '04%'
  4  UNION
  5  SELECT employee_id, first_name
  6    FROM   employees
  7    WHERE  department_id=20;
```

	EMPLOYEE_ID	FIRST_NAME
1	120	Matthew
2	133	Jason
3	145	John
4	156	Janette
5	157	Patrick
6	158	Allan
7	174	Ellen
8	184	Nandita
9	192	Sarah
10	201	Michael
11	202	Pat

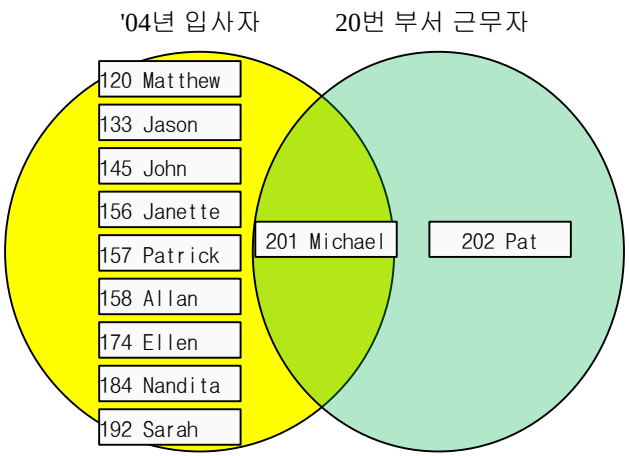
UNION- 합집합(중복x)




```
EMPLOYEE_ID FIRST_NAME
----------- ----------
        120 Matthew
        133 Jason
        145 John
        156 Janette
        157 Patrick
        158 Allan
        174 Ellen
        184 Nandita
        192 Sarah
        201 Michael
        202 Pat
```

```
SQL> SELECT employee_id, first_name
  2    FROM   employees
  3    WHERE  hire_date LIKE '04%'
  4  UNION  ALL
  5  SELECT employee_id, first_name
  6    FROM   employees
  7    WHERE  department_id=20;
```

UNION ALL-합집합(중복o)



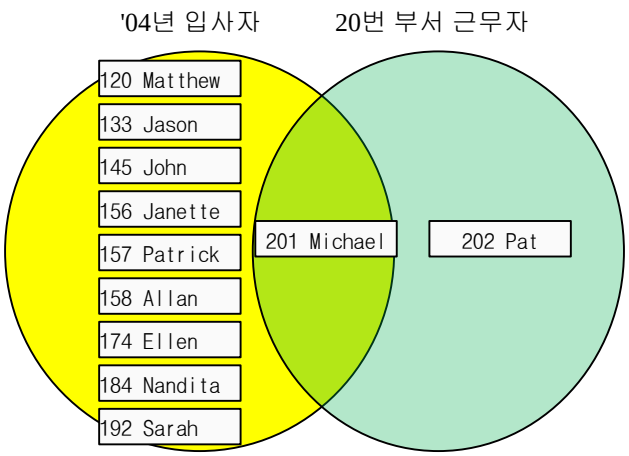

```
EMPLOYEE_ID FIRST_NAME
----------- ----------
        120 Matthew
        133 Jason
        145 John
        156 Janette
        157 Patrick
        158 Allan
        174 Ellen
        184 Nandita
        192 Sarah
        201 Michael
        201 Michael
        202 Pat
```

	EMPLOYEE_ID	FIRST_NAME
1	120	Matthew
2	133	Jason
3	145	John
4	156	Janette
5	157	Patrick
6	158	Allan
7	174	Ellen
8	184	Nandita
9	192	Sarah
10	201	Michael
11	201	Michael
12	202	Pat

```
SQL> SELECT employee_id, first_name
  2    FROM   employees
  3    WHERE  hire_date LIKE '04%'
  4  INTERSECT
  5  SELECT employee_id, first_name
  6    FROM   employees
  7    WHERE  department_id=20;
```

	EMPLOYEE_ID	FIRST_NAME
1	201	Michael

INTERSECT-(교집합)

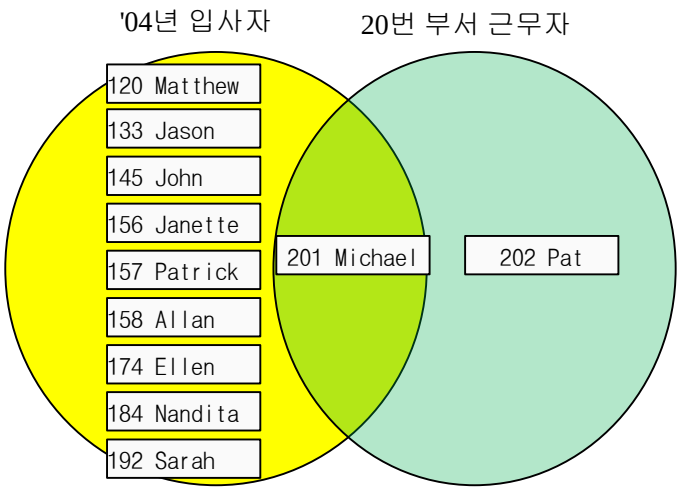



```
EMPLOYEE_ID FIRST_NAME
----------- ----------
        120 Matthew
        133 Jason
        145 John
        156 Janette
        157 Patrick
        158 Allan
        174 Ellen
        184 Nandita
        192 Sarah
        201 Michael
        201 Michael
        202 Pat
```

```
SQL> SELECT employee_id, first_name
  2    FROM   employees
  3    WHERE  hire_date LIKE '04%'
  4  MINUS
  5  SELECT employee_id, first_name
  6    FROM   employees
  7    WHERE  department_id=20;
```

	EMPLOYEE_ID	FIRST_NAME
1	120	Matthew
2	133	Jason
3	145	John
4	156	Janette
5	157	Patrick
6	158	Allan
7	174	Ellen
8	184	Nandita
9	192	Sarah

Minus - 차집합




```
EMPLOYEE_ID FIRST_NAME
----------- ----------
        120 Matthew
        133 Jason
        145 John
        156 Janette
        157 Patrick
        158 Allan
        174 Ellen
        184 Nandita
        192 Sarah
```

Chapter 3

함수
문자 조작 함수
숫자 함수
날짜 함수
변환 함수
집합 연산자
분석 함수

함 수 명	설 명
RANK	해당 값에 대한 우선순위를 결정(중복순위 계산함)
DENSE_RANK	해당 값에 대한 우선순위를 결정(중복순위 계산 안 함)
ROW_NUMBER	조건을 만족하는 모든 행의 번호를 제공
CUME_DIST	최대값 1을 기준으로 분산된 값을 제공
RATIO_TO_REPORT	해당 컬럼 값의 백분율을 소수점으로 제공
PERCENT_RANK	최대값 1을 기준으로 Percent값을 제공
NTILE(n)	전체 데이터 분포를 m-Buckets로 나누어 표시
FIRST_VALUE	현재 행까지 값 중에서 가장 첫 번째 값
LAST_VALUE	현재 행까지 값 중에서 가장 마지막 번째 값

함 수 명	설 명
RANK	해당 값에 대한 순위를 결정합니다.(중복순위를 계산합니다. 즉 2등이 2명이면 다음 순위는 4등입니다.)
DENSE_RANK	해당 값에 대한 순위를 결정합니다.(중복순위 계산하지 않습니다. 즉 2등이 2명이더라도 다음 순위는 3등입니다.)
ROW_NUMBER	조건을 만족하는 모든 행의 번호를 제공합니다. 일련번호를 생성합니다.

```
SQL> SELECT employee_id, department_id, salary,
  2       RANK()       OVER (ORDER BY salary DESC) sal_rank,
  3       DENSE_RANK() OVER (ORDER BY salary DESC) sal_dense_rank,
  4       ROW_NUMBER() OVER (ORDER BY salary DESC) sal_number
  5  FROM   employees
  6  ORDER BY sal_number;
```

분석함수() OVER (순위의 조건)

	EMPLOYEE_ID	DEPARTMENT_ID	SALARY	SAL_RANK	SAL_DENSE_RANK	SAL_NUMBER
1	100	90	24000	1	1	1
2	101	90	17000	2	2	2
3	102	90	17000	2	2	3
4	145	80	14000	4	3	4
5	146	80	13500	5	4	5
6	201	20	13000	6	5	6
7	108	100	12008	7	6	7
8	205	110	12008	7	6	8

함 수 명	설 명
CUME_DIST	최댓값 1을 기준으로 분산된 값을 제공합니다. 최솟값과 최댓값 사이의 상대적인 위치를 의미합니다. 예를 들면 위 예에서는 첫 번째 row는 1/행수, 두 번째는 2/행수 가 됩니다.
PERCENT_RANK	최댓값 1을 기준으로 데이터 집합에서 특정 값의 백분율(Percent) 순위를 제공합니다. CUME_DIST와 비슷하지만 식이 다릅니다. 첫 번째 위치가 0부터 시작하고 두 번째 row부터의 위치는 (row의 rank-1) / (전체 row 개수 -1)이 됩니다.

```
SQL> SELECT employee_id, department_id, salary,
  2      CUME_DIST()    OVER (ORDER BY salary DESC) sal_cume_dist,
  3      PERCENT_RANK() OVER (ORDER BY salary DESC) sal_pct_rank
  4  FROM   employees;
```

	EMPLOYEE_ID	DEPARTMENT_ID	SALARY	SAL_CUME_DIST	SAL_PCT_RANK
1	100	90	24000	0.009345794392523364485981308411214953271028	0
2	101	90	17000	0.028037383177570093457943925233644859813	0.009433962264150943396226415094339622641509
3	102	90	17000	0.028037383177570093457943925233644859813	0.009433962264150943396226415094339622641509
4	145	80	14000	0.037383177570093457943925233644859813084	0.028301886792452830188679245283018867924528
5	146	80	13500	0.046728971962616822429906542056074766355	0.037735849056603773584905660377358490566
6	201	20	13000	0.056074766355140186915887850467289719626	0.047169811320754716981132075471698113207
7	108	100	12008	0.074766355140186915887850467289719626168	0.056603773584905660377358490566037735849
8	205	110	12008	0.074766355140186915887850467289719626168	0.056603773584905660377358490566037735849
9	147	80	12000	0.084112149532710280373831775700934579439	0.075471698113207547169811320754716981132
10	168	80	11500	0.093457943925233644859813084112149532710	0.084905660377358490566037735849056603773
				...	
105	136	50	2200	0.990654205607476635514018691588785046729	0.981132075471698113207547169811320754717
106	128	50	2200	0.990654205607476635514018691588785046729	0.981132075471698113207547169811320754717
107	132	50	2100	1	1

함수명	설 명
RATIO_TO_REPORT	해당 컬럼 값의 백분율을 소수점으로 제공합니다. 이를 이용하면 그룹 내에서 해당 값의 백분율을 구할 수 있습니다. 결과 값은 0보다 크고 1.0보다 작거나 같은 값이 출력됩니다.

```
SQL> SELECT first_name, salary,
  2    ROUND(RATIO_TO_REPORT(salary) OVER (), 4) AS salary_ratio
  3  FROM   employees
  4  WHERE  job_id='IT_PROG';
```

	FIRST_NAME	SALARY	SALARY_RATIO
1	Alexander	9000	0.3125
2	Bruce	6000	0.2083
3	David	4800	0.1667
4	Valli	4800	0.1667
5	Diana	4200	0.1458