

Day 25

☰ Tags	
📅 Date	@2022년 8월 3일
📍 강의 번호	AUS 101

-Algorithm-

< Sieve of Eratosthenes (에라토스테네스의 체) >

- 소수 (prime number)를 찾아내는 알고리즘
- 소수는 2 이상의 정수에서 1과 그 수 자체로만 나눌 수 있는 수
- 소수는 나열되어 있는 구간이 불규칙하므로 임의로 찾기가 힘들다.

[소수 prime number]

: 소수는 2 이상의 정수 중에서 1과 그 수 자신 외는 나눌 수 없는 숫자.

: 10 이하에서는 2, 3, 5, 7 이 소수에 해당한다.

→ 4는 2와 2로, 6은 2와 3으로 나눌 수 있다. 따라서 소수가 아니다. 이러한 숫자들은 합성수 즉, 숫자와 숫자가 합성된 것이라고 할 수 있다. 반대로 말하면, 합성수가 아닌 숫자들이 바로 소수인 것이다.

- 소수 prime에서의 ' 소 '. 합성되지 않은 소박한 숫자 의미의 ' 소 '를 뜻한다.

: 모든 수의 소 (근본)을 의미하기도 한다.

! 소수인지 아닌지를 구분하는 것은 의외로 어렵다.

: 소수의 내용으로만 보면 어렵지 않아보이지만 사실은 의외로 아주 어렵다. 무엇이 어려운지도 바로 찾아내기 힘들다.

- 예로 3의 배수는 3, 6, 9, 12, 15 처럼 3개의 간격으로 나열된다. 따라서 1에서 100까지의 사이에 있는 3의 배수를 찾는 일은 간단하다. 하지만 소수는 규칙성이 없기 때문에 간격이 불규칙하고 랜덤하다. 즉, 소수는 한 번에 열거하기가 어렵다.

→ 그러면 어떻게 구할 수 있을까, 가장 먼저 떠오르는 방법은 하나하나 그 수보다 작은 숫자로 나누어보고 나눌 수 있는지의 여부를 확인하는 것이다.

: 예를 들어, 2에서 100까지의 소수를 찾으려면 먼저 2로 나눌 수 있는 수를 모두 지우고 그 다음 3으로 나눌 수 있는 숫자를 모두 지우고 그 다음 4로 나눌 수 있는 수를 모두 지우고..... 마지막에는 99로 나눌 수 있는 수를 지우는 방법이다.

하지만, 이 방법은 비효율적(컴퓨터로 실행할 경우 많은 데이터의 소모가 필요하기 때문에 비효율적이라고 보는 것)이고 수의 범위가 커지는 경우에는 많은 시간이 소모될 것이다.

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

사진 출처 - <https://dolphins-it.tistory.com/105>

고대 그리스의 과학자인 에라토스테네스는 위의 방법을 개선하여 소수를 효율적으로 발견하는 방법을 알아냈다. 그의 이름을 따서 ‘ 에라토스테네스의 체 ’ 라고 부른다.

[에라토스테네스의 체]

: 어떤 수 이하의 범위에 존재하는 소수를 찾고 싶은 경우

⇒ ‘ 그 수의 제곱근 보다 작은 소수의 배수만 없애면 남은 수가 소수다 ’ .

라는 생각을 바탕으로 소수를 찾는 방법이다.

: 예를 들어 100이하의 소수를 모두 찾아내려면 먼저 100의 제곱근 이하 소수를 선택한다.

$\sqrt{100}$ 의 제곱근은 10이다. 즉, 제곱하면 100이 되는 수는 10이다. 10 이하에서의 소수는 2, 3, 5, 7 네 개이다.

우선 2에서 100까지의 표에서 2로 나눌 수 있는 즉, 2의 배수 중 2를 제외한 나머지를 모두 삭제한다.

맨 처음에 소수인 2를 발견한 후 2의 배수를 모두 지운다.

	2	3		5		7		9
11		13		15		17		19
21		23		25		27		29
31		33		35		37		39
41		43		45		47		49
51		53		55		57		59
61		63		65		67		69
71		73		75		77		79
81		83		85		87		89
91		93		95		97		99

이런 방식으로 3보다 큰 수에 대해서도 $n \times n$ 부터 지워주면 된다.

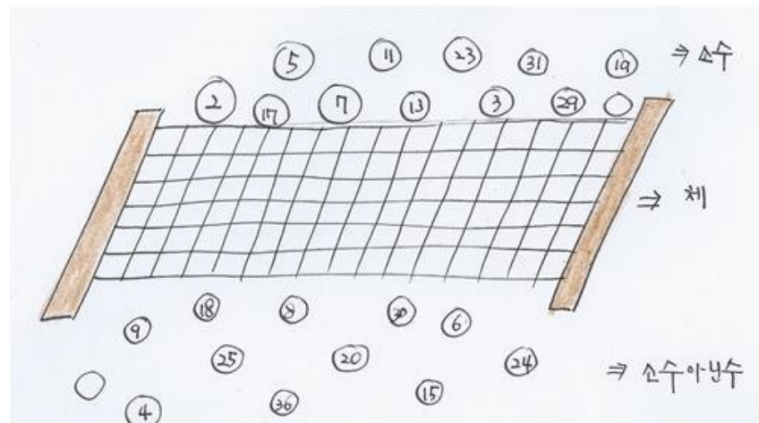
	2	3		5		7		
11		13				17		19
		23		25				29
31				35		37		
41		43				47		49
		53		55				59
61				65		67		
71		73				77		79
		83		85				89
91				95		97		

같은 방식으로 5와 7의 배수들도 모두 삭제한 결과는 아래와 같다.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

사진 출처 - 눈높이 대백과

에라토스테네스의 체



이 방법을 사용하면 1에서 100까지의 모든 숫자를 해당 숫자보다 작은수로 나눌 수 있는지 하나하나 전부 순서대로 확인하는 것보다는 훨씬 더 빨리 소수를 찾을 수 있다.

[에라토스테네스의 체 알고리즘]

- 에라토스테네스의 체는 크게 3개의 처리로 구성한다.

: 어떤 수 이하의 모든 정수 데이터를 준비

: 어떤 수의 제곱근 보다 작은 소수의 배수들을 차례대로 제거한다.
 : 마지막까지 남은 수들을 출력한다.

- 우선 10 이하의 소수를 구하는 경우로 생각해보자.

: 10 이하의 정수 데이터들을 준비한다.
 : 10의 제곱근 약 3.16 보다 작은 소수의 배수들을 차례대로 제거한다.
 : 마지막까지 남은 수들을 출력한다.

1. 10 이하의 정수 데이터들을 준비한다.

: 먼저 체의 대상이 되는 10까지의 정수를 데이터로 준비하자. 여기에서는 11개의 요소를 가지는 정수형 배열을 준비한다. 배열의 이름은 arr로 정하자. 첨자는 0부터 시작하기 때문에 첨자를 10까지로 동일하게 사용하려면 요소를 11개 준비해야 한다.

0	1	2	3	4	5	6	7	8	9	10

11개의 배열을 준비한다.

요소는 “ 그 첨자가 소수인지의 여부를 판정하는 데이터를 넣는 것으로 사용하자.

즉, 소수가 아닌 것으로 판정된 첨자의 요소에 ‘ 소수가 아니다 ‘라는 것을 나타내는 데이터를 넣자.

즉, 소수의 가능성이 있는 경우에는 1을 대입하고 소수가 아닌 경우에는 0을 대입한다.

위 개념 소개에서의 소수가 아닌 수를 제거하는 방식을 0을 대입하는 방식으로 처리한다.

따라서, 초기값을 1로 전부 대입해둔다. 그리고 소수가 아닌 것으로 판정된 수 (첨자)의 요소에는 0을 대입한다.

이러한 방식을 통해 소수의 배수를 모두 제거 (0을 대입)한 후 마지막에 ‘ 1 ‘이 남아 있는 요소들의 첨자는 모두 소수이고 0이 대입되어 있는 요소의 첨자는 소수가 아니다. “라고 구별할 수 있다.

1	1	1	1	1	1	1	1	1	1	1
0	1	2	3	4	5	6	7	8	9	10

기본값 1로 전부 채운다.

10의 제곱근은 약 3.16... 이므로 제곱근 이하의 소수는 2와 3이다. 따라서 2의 배수들 모두와 3의 배수들 모두를 제거하면 된다.

2. 10의 제곱근 약 3.16 보다 작은 소수 (2, 3)의 배수를 차례대로 제거한다. (0으로 대입한다.)

먼저 2의 배수들을 모두 지운다. 이 때 지운 숫자의 의미를 0을 대입한다는 표현으로 적용하자.

1	1	1	1	0	1	0	1	0	1	0
0	1	2	3	4	5	6	7	8	9	10

2의 배수를 전부 0으로 대입

다음으로 3의 배수들을 모두 지운다. 이때 지운 숫자의 의미를 0을 대입한다는 표현으로 적용하자.

1	1	1	1	0	1	0	1	0	0	0
0	1	2	3	4	5	6	7	8	9	10

3의 배수를 전부 0으로 대입

3. 마지막까지 남은 수들을 출력한다.

: 제곱근 이하의 소수들의 모든 배수를 제거 (0으로 대입)했다면 나머지는 모두 소수뿐이다. (여전히 요소의 값이 1인) 그럼 이제 첨자가 2 이상이고 1이 들어있는 요소의 첨자들만 뽑아내면 그것들이 바로 10 이하의 소수가 된다.

1	1	1	1	0	1	0	1	0	0	0
0	1	2	3	4	5	6	7	8	9	10

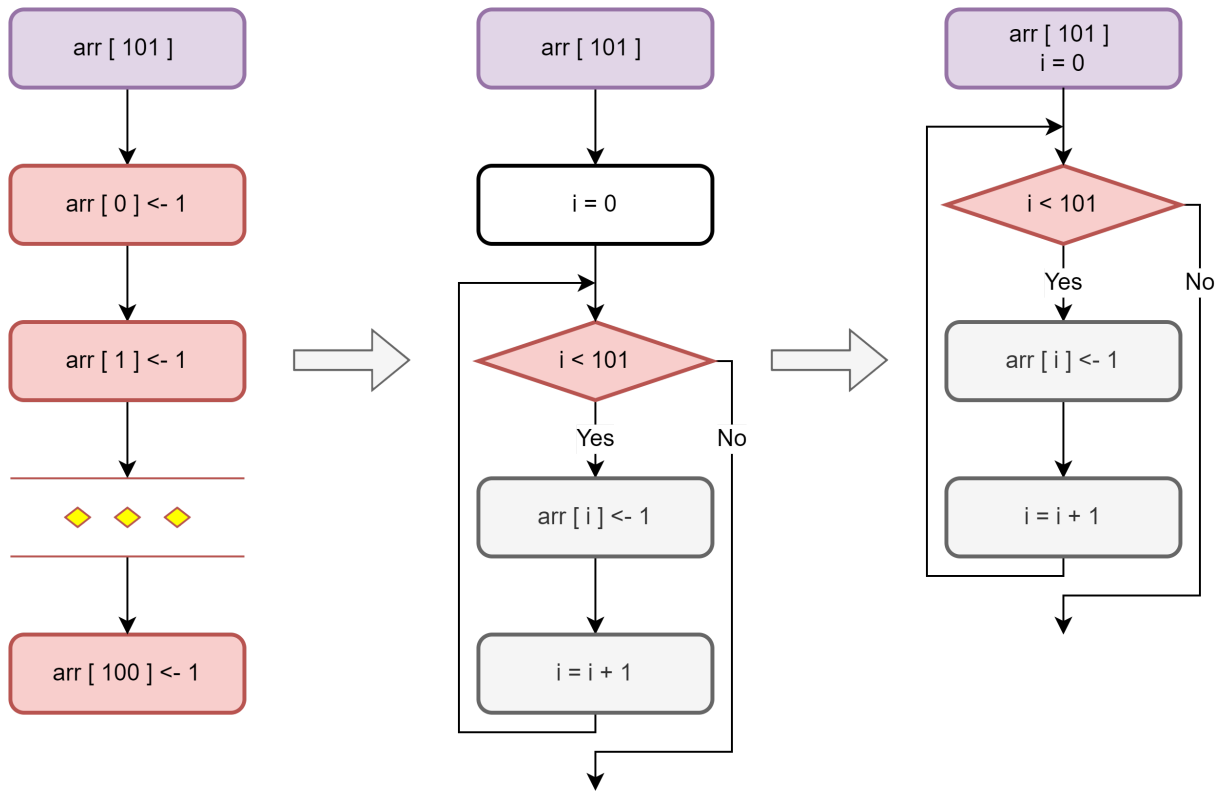
첨자가 2 이상이고, 1이 들어있는 요소의 첨자들만 뽑아내자.

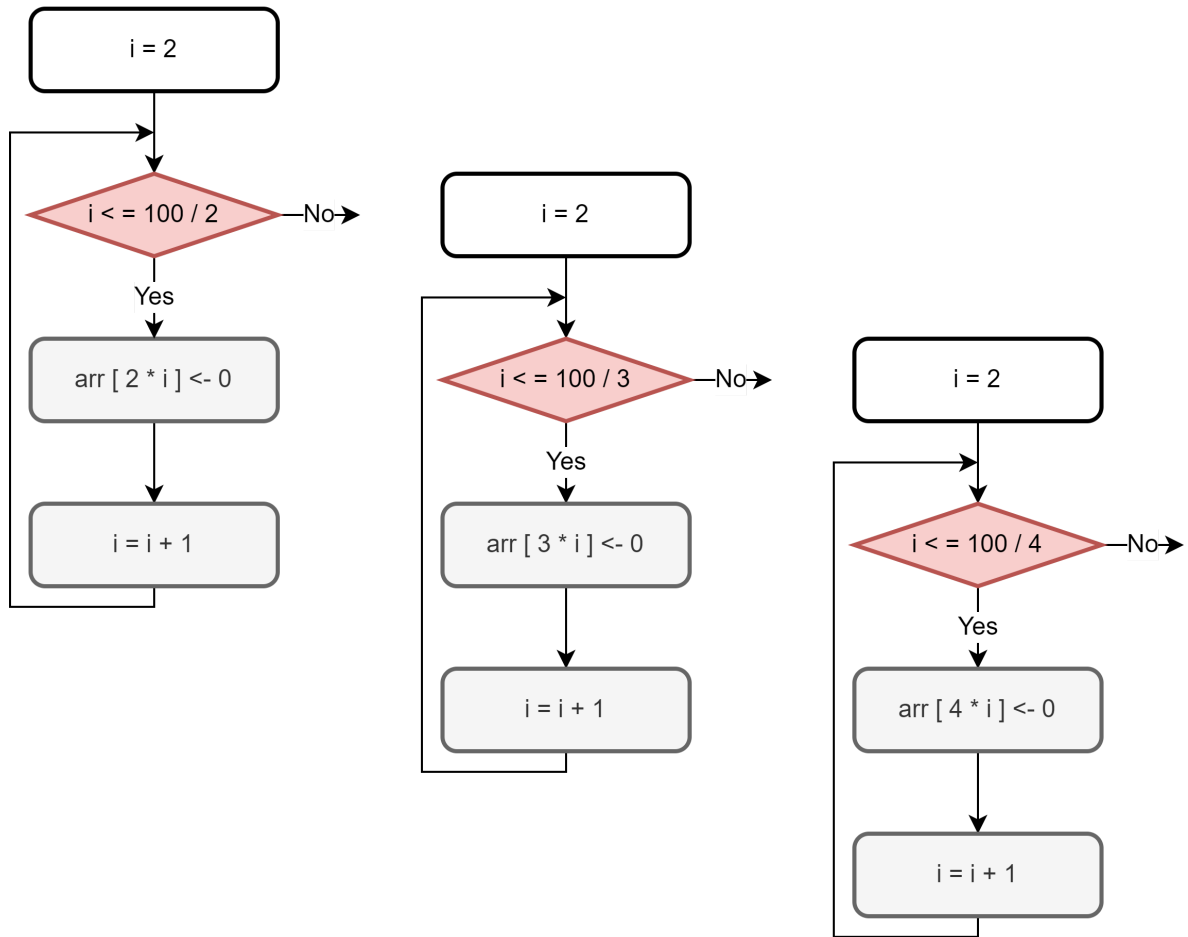
[에라토스테네스의 체 순서도]

: 이번에는 수를 조금 늘려서 100 이하의 소수를 모두 구하는 경우로 생각해보자.

: 따라서, 배열의 요소수는 101로 한다. 앞에서 생각한 것과 같이 첨자들을 ‘ 수 ’ 로 사용하고자 하기 때문이다. 첨자는 0부터 시작하기 때문에 요소의 수는 101개 필요하다.

: 모든 요소에는 초기값으로 1을 대입하여 초기화 하자.



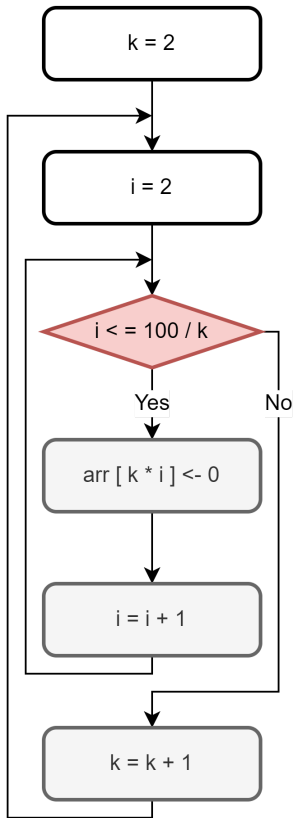


변수 k는 하나씩 증가한다. 2인 경우에는 2의 배수들을 전부 제거 즉, 0으로 바꾸고, 3인 경우에는 3의 배수들을 전부 제거하고 즉, 0으로 바꾸고, 계속 반복 시킨다. 그러나 4는 소수가 아니다.

3 다음의 소수는 5이므로 5의 배수를 제거해야만 한다. k 값이 소수인지를 판단하는 처리를 추가해야 한다.

2와 3의 배수 제거 처리가 끝난 단계에서의 배열을 생각해보면 이미 2의 배수와 3의 배수는 모두 제거된 즉, 0인 상태가 되었다.

⇒ 이 시점에서 첨자가 소수가 아닌 요소에는 0이 대입되어 있다. 즉, 4에는 0이 대입된 상태이다.



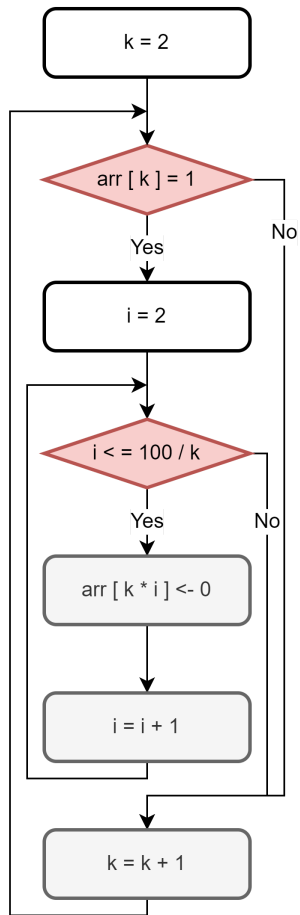
k에 대입된 값이 소수인지의 여부에 따라 반복문으로의 진입을 판단해야 한다. 따라서 $arr[k] = 1$ 이면 소수, $arr[k] = 0$ 이면 소수가 아니라는 것이다.

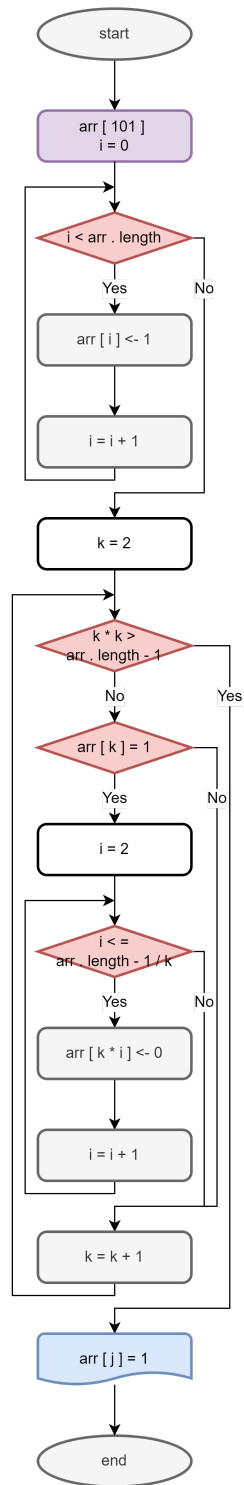
$arr[k] = 1$ 이 Yes인 경우, k는 소수이므로 k의 배수를 제거하는 반복 처리가 실행된다. 이와는 반대로, $arr[k] = 0$ 이 No인 경우, k는 소수가 아니므로 k를 하나 늘려 다음 k가 소수인지를 판명하게 된다.

k값은 이제 2, 3으로 증가하고 k = 4인 경우에는 $arr[4] = 0$ 이므로 배수를 제거하는 반복처리로 들어가지 않고 k만 하나 증가 시키게 된다.

그 다음은 k = 5인 경우, $arr[5] = 1$ 이므로 5의 배수들을 모두 제거하고... 계속 반복 된다. 따라서 무한루프에 빠지게 된다...

에라토스테네스의 체는 ‘어떤 숫자의 제곱근 보다 작은 소수의 배수를 제거하면 남은 수가 소수이다.’라는 이론이다. 따라서 이번 예제에서는 그 어떤 수가 100이므로 k는 $\sqrt{100}$ 의 제곱근 즉, 10 이하가 된다. 따라서, k가 $\sqrt{100}$ 의 제곱근 이하인가? 라는 판별식을 추가하여 반복을 마치고 소수들을 출력하자.

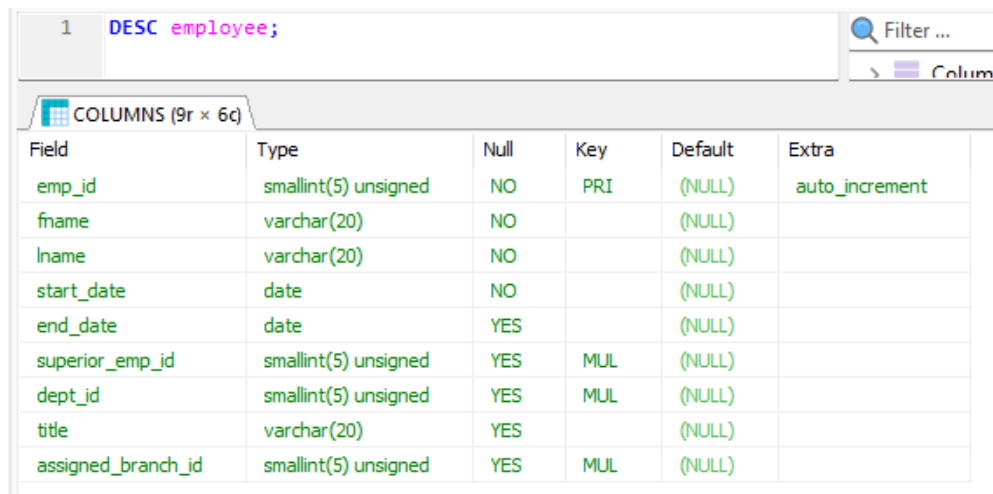




-DB-

< 다중 테이블 질의 (쿼리) > - chapter 5.

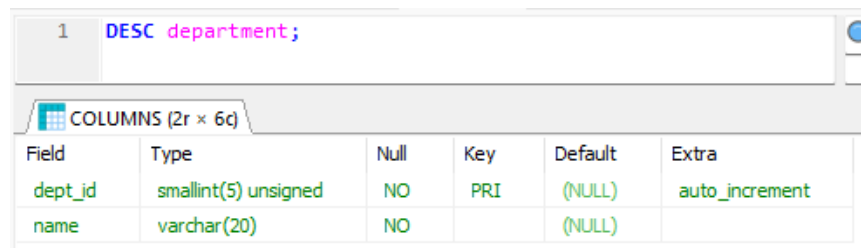
-. 하나의 테이블에만 쿼리를, 질의를 던지는 것이 적지 않다. 두 개, 세 개 .. 더 많은 테이블의 질의를 던질 수 있어야 한다.



The screenshot shows a database query tool interface. At the top, a text box contains the command `DESC employee;`. Below it, a tab labeled "COLUMNS (9r x 6c)" is active, displaying a table with the following data:

Field	Type	Null	Key	Default	Extra
emp_id	smallint(5) unsigned	NO	PRI	(NULL)	auto_increment
fname	varchar(20)	NO		(NULL)	
lname	varchar(20)	NO		(NULL)	
start_date	date	NO		(NULL)	
end_date	date	YES		(NULL)	
superior_emp_id	smallint(5) unsigned	YES	MUL	(NULL)	
dept_id	smallint(5) unsigned	YES	MUL	(NULL)	
title	varchar(20)	YES		(NULL)	
assigned_branch_id	smallint(5) unsigned	YES	MUL	(NULL)	

```
DESC employee;
```



The screenshot shows a database query tool interface. At the top, a text box contains the command `DESC department;`. Below it, a tab labeled "COLUMNS (2r x 6c)" is active, displaying a table with the following data:

Field	Type	Null	Key	Default	Extra
dept_id	smallint(5) unsigned	NO	PRI	(NULL)	auto_increment
name	varchar(20)	NO		(NULL)	

```
DESC department;
```

두 개 테이블에 동시에 던진다.

```
mysql> SELECT e.fname, e.lname, d.name
-> FROM employee e JOIN department d;
```

fname	lname	name
Michael	Smith	Operations
Michael	Smith	Loans
Michael	Smith	Administration
Susan	Barker	Operations
Susan	Barker	Loans
Susan	Barker	Administration
Robert	Tyler	Operations
Robert	Tyler	Loans

```
1 SELECT e.fname,e.lname,d.name
2 FROM employee e JOIN department d;
```

Result #1 (54r x 3c)

fname	lname	name
Michael	Smith	Operations
Michael	Smith	Loans
Michael	Smith	Administration
Susan	Barker	Operations
Susan	Barker	Loans
Susan	Barker	Administration
Robert	Tyler	Operations
Robert	Tyler	Loans
Robert	Tyler	Administration
Susan	Hawthorne	Operations
Susan	Hawthorne	Loans
Susan	Hawthorne	Administration
John	Gooding	Operations
John	Gooding	Loans
John	Gooding	Administration

```
SELECT e.fname, e.lname, d.name
FROM employee e JOIN department d;
```

=내부조인= (INNER JOIN)

```
mysql> SELECT e.fname, e.lname, d.name
-> FROM employee e JOIN department d
-> ON e.dept_id = d.dept_id;
```

```

1 SELECT e.fname,e.lname,d.name
2 FROM employee e JOIN department d
3 ON e.dept_id=d.dept_id;

```

Result #1 (18r x 3c)		
fname	lname	name
Susan	Hawthorne	Operations
Helen	Fleming	Operations
Chris	Tucker	Operations
Sarah	Parker	Operations
Jane	Grossman	Operations
Paula	Roberts	Operations

```

SELECT e.fname,e.lname,d.name
FROM employee e JOIN department d
ON e.dept_id=d.dept_id;

```

같은 테이블을 두번 사용 (테이블 재사용)

```

mysql> SELECT a.account_id, e.emp_id,
-> b_a.name open_branch, b_e.name emp_branch
-> FROM account a INNER JOIN branch b_a
-> ON a.open_branch_id = b_a.branch_id
-> INNER JOIN employee e
-> ON a.open_emp_id = e.emp_id
-> INNER JOIN branch b_e
-> ON e.assigned_branch_id = b_e.branch_id
-> WHERE a.product_cd = 'CHK';

```

account_id	emp_id	open_branch	emp_branch
10	1	Headquarters	Headquarters
14	1	Headquarters	Headquarters
21	1	Headquarters	Headquarters
1	10	Woburn Branch	Woburn Branch
4	10	Woburn Branch	Woburn Branch

1	SELECT a.account_id,e.emp_id,
2	b_a.name open_branch,b_e.name emp_branch
3	FROM account a INNER JOIN branch b_a
4	ON a.open_branch_id=b_a.branch_id
5	INNER JOIN employee e
6	ON a.open_emp_id=e.emp_id
7	INNER JOIN branch b_e
8	ON e.assigned_branch_id=b_e.branch_id
9	WHERE a.product_cd='CHK';

Result #1 (10r × 4c)				
account_id	emp_id	open_branch	emp_branch	
10	1	Headquarters	Headquarters	
14	1	Headquarters	Headquarters	
21	1	Headquarters	Headquarters	
1	10	Woburn Branch	Woburn Branch	
4	10	Woburn Branch	Woburn Branch	
7	13	Quincy Branch	Quincy Branch	
13	16	So. NH Branch	So. NH Branch	
18	16	So. NH Branch	So. NH Branch	
24	16	So. NH Branch	So. NH Branch	
28	16	So. NH Branch	So. NH Branch	

```
SELECT a.account_id,e.emp_id,
b_a.name open_branch,b_e.name emp_branch
FROM account a INNER JOIN branch b_a
ON a.open_branch_id=b_a.branch_id
INNER JOIN employee e
ON a.open_emp_id=e.emp_id
INNER JOIN branch b_e
ON e.assigned_branch_id=b_e.branch_id
WHERE a.product_cd='CHK';
```

▼ 아래처럼 같은 테이블 다른 별칭으로 재사용 가능.

```
SELECT e.fname,a.lname
FROM employee AS e,employee AS a;
```

=문제=

사원 테이블의 이름을 출력하는데 첫 번째 컬럼은 이름을 대문자로 출력하고 두 번째 컬럼은 이름을 소문자로 출력하고 세 번째 컬럼은 이름의 첫 번째 철자는 대문자로 하고 나머지는 소문자로 출력해 보겠습니다.

Quiz 16.

UPPER(ENAME)	LOWER(ENAME)	INITCAP(ENAME)
KING	king	King
BLAKE	blake	Blake
:	:	:
ADAMS	adams	Adams
MILLER	miller	Miller

```
SELECT UPPER(ename), LOWER(ename), INITCAP(ename)
FROM emp;
```

1	SELECT UPPER(ename), LOWER(ename), INITCAP(ename)
2	FROM emp;
Result #1 (14r x 2c)	
UPPER(ename)	LOWER(ename)
KING	king
BLAKE	blake
CLARK	clark
JONES	jones
MARTIN	martin
ALLEN	allen
TURNER	turner
JAMES	james
WARD	ward
FORD	ford
SMITH	smith
SCOTT	scott
ADAMS	adams
MILLER	miller

오라클
함수

```
SELECT UPPER(ename), LOWER(ename), INITCAP(ename)
FROM emp;
```

SQL에서의 함수는 크게 2가지로 나눌 수 있다.

1. 단일행 함수 - 하나의 행을 입력받아 하나의 행으로 출력하는 함수
2. 다중행 함수 - 여러행의 값을 입력받아 하나의 행으로 출력하는 함수

1	SELECT UPPER(ename), LOWER(ename), CONCAT(UPPER(SUBSTR(ename,1,1)), LOWER(SUBSTR(ename,2)))	
2	FROM emp;	

emp (14r × 3c)		
UPPER(ename)	LOWER(ename)	CONCAT(UPPER(SUBSTR(ename,1,1)), LOWER(SUBSTR(e...
KING	king	King
BLAKE	blake	Blake
CLARK	clark	Clark
JONES	jones	Jones
MARTIN	martin	Martin
ALLEN	allen	Allen
TURNER	turner	Turner
JAMES	james	James
WARD	ward	Ward
FORD	ford	Ford
SMITH	smith	Smith
SCOTT	scott	Scott
ADAMS	adams	Adams
MILLER	miller	Miller

INITCAP 함수가 오라클에서만 작동하기 때문에 CONCAT을 이용해서 같은 결과가 나오도록 만들었다.

```
SELECT UPPER(ename), LOWER(ename), CONCAT(UPPER(SUBSTR(ename,1,1)), LOWER(SUBSTR(ename,2)))
FROM emp;
```

영어 단어 SMITH에서 SMI만 잘라내서 출력해 보겠습니다.

Quiz 17.

1	SELECT SUBSTR("SMITH",1,3)	
2	FROM dual;	

Result #1 (1r × 1c)	
SUBSTR("SMITH",1,3)	
SMI	

```
SELECT SUBSTR("SMITH",1,3)
FROM dual;
```

dual 테이블은 더미 테이블로 특정 테이블에 종속되지 않는 내용을 확인할 때 사용되는 일종의 가상의 연습용 테이블이다.

SQL은 다른 대부분의 컴퓨터 언어와는 달리 1에서 시작한다.

이름을 출력하고 그 옆에 이름의 철자 개수를 출력해 보겠습니다.

Quiz 18.

ENAME	LENGTH(ENAME)
KING	4
BLAKE	5
:	:
ADAMS	5
MILLER	6

```
SELECT ename, LENGTH(ename)
FROM emp;
```

사원 이름 SMITH에서 알파벳 철자 M이 몇 번째 자리에 있는지 출력해 보겠습니다.

Quiz 19.

```
INSTR('SMITH', 'M')
2
```

```
SELECT INSTR('SMITH', 'M')
FROM DUAL;
```

abcdefgh@naver.com 이메일에서 naver.com만 추출하고 싶다'

Quiz 19_1.

```

1 SELECT SUBSTR('abcdefgh@naver.com',10,18)
2 FROM dual;

```

Result #1 (1r x 1c)

SUBSTR('abcdefgh@naver.com',10,18) naver.com

쉬운 버전.

```

SELECT SUBSTR('abcdefgh@naver.com',10,18)
FROM dual;

```

```

1 SELECT SUBSTR('abcdefgh@naver.com',INSTR('abcdefgh@naver.com','@')+1)
2 FROM dual;

```

Result #1 (1r x 1c)

SUBSTR('abcdefgh@naver.com',INSTR('abcdefgh@naver... naver.com

심화 버전. @앞의 내용이 달라져도, 그 뒤의 주소를 어느 메일주소에서든지 추출해 낼 수 있다.

```

SELECT SUBSTR('abcdefgh@naver.com',INSTR('abcdefgh@naver.com','@')+1)
FROM dual;

```

이름과 월급을 출력하는데, 월급을 출력할 때 숫자 0을 *(별표)로 출력해 보겠습니다.

Quiz 20.

KING	5***
BLAKE	285*
:	:
ADAMS	11**
MILLER	13**

```

SELECT ename, REPLACE(sal, 0, '*')
FROM emp;

```

이름과 월급을 출력하는데 월급 컬럼의 자릿수를 10자리로 하고, 월급을 출력하고 남은 나머지 자리에 별표(*)를 채워서 출력해 보겠습니다.

Quiz 21.

ENAME	SALARY1	SALARY2
KING	*****5000	5000*****
BLAKE	*****2850	2850*****
CLARK	*****2450	2450*****
:	:	:
SCOTT	*****3000	3000*****
ADAMS	*****1100	1100*****
MILLER	*****1300	1300*****

```
SELECT ename, LPAD(sal,10,'*') as salary1, RPAD(sal,10,'*') as salary2
FROM emp;
```

1	SELECT ENAME,LPAD(sal,10,'*')AS SALARY1,RPAD(sal,10,'*')AS SALARY2
2	FROM emp;

ENAME	SALARY1	SALARY2
KING	*****5000	5000*****
BLAKE	*****2850	2850*****
CLARK	*****2450	2450*****
JONES	*****2975	2975*****
MARTIN	*****1250	1250*****
ALLEN	*****1600	1600*****
TURNER	*****1500	1500*****
JAMES	*****950	950*****
WARD	*****1250	1250*****
FORD	*****3000	3000*****
SMITH	*****800	800*****
SCOTT	*****3000	3000*****
ADAMS	*****1100	1100*****
MILLER	*****1300	1300*****

```
SELECT ENAME,LPAD(sal,10,'*')AS SALARY1,RPAD(sal,10,'*')AS SALARY2
FROM emp;
```

첫 번째 컬럼은 영어 단어 smith 철자를 출력하고, 두 번째 컬럼은 영어 단어 smith에서 s를 잘라서 출력하고, 세 번째 컬럼은 영어 단어 smith에서 h를 잘라서 출력하고, 네 번째 컬럼은 영어 단어 smiths의 양쪽에 s를 잘라서 출력해 보겠습니다.

Quiz 22.

	smith	mith	smit	mith
1	<pre>SELECT 'smith',SUBSTR('smith',2),SUBSTR('smith',1,LENGTH('smith')-1),SUBSTR('smiths',2,LENGTH('smiths')-2) FROM dual;</pre>			
2				
Result #1 (1r x 4c)				
smith	SUBSTR('smith',2)	SUBSTR('smith',1,LENGTH('smith')-1)	SUBSTR('smiths',2,LENGTH('smiths')-2)	
smith	mith	smit	mith	

배운 것을 활용해보기 위해 돌아가는 방법.

```
SELECT 'smith',SUBSTR('smith',2),SUBSTR('smith',1,LENGTH('smith')-1),SUBSTR('smiths',2,LENGTH('smiths')-2)
FROM dual;
```

```
SELECT 'smith', LTRIM('smith','s'), RTRIM('smith','h'), TRIM('s' from
'smiths')
FROM dual;
```

⇒ 이름 사이의 공백을 지우기 위해서 주로 사용됨. 공백 전용.

TRIM(삭제옵션(선택), 삭제할문자(선택), 원본문자열 데이터 (필수))	특정 문자를 지움	SELECT trim(ENAME), trim('A' FROM ENAME), trim(LEADING 'A' FROM ENAME), trim(TRAILING 'S' FROM ENAME) FROM EMP;	
LTRIM(원본문자열, 삭제할문자(선택)) RTRIM(원본문자열, 삭제할문자(선택))	왼쪽, 오른쪽 지정문자를 삭제	SELECT Ltrim(ENAME), Rtrim(ENAME) FROM EMP;	선택문자열 없음 공백만 삭제

1	SELECT 'smith',LTRIM(' smith '),RTRIM(' smith '),TRIM(' 'from smiths ')		
2	FROM DUAL;		

Result #1 (1r x 4c)			
smith	LTRIM(' smith ')	RTRIM(' smith ')	TRIM(' 'from smiths ')
smith	smith	smith	smiths

LTRIM - 왼쪽 공백 삭제 , RTRIM - 오른쪽 공백 삭제 , TRIM - 양쪽 공백 삭제 .

```
SELECT 'smith',LTRIM(' smith '),RTRIM(' smith '),TRIM(' 'from' smiths ')
FROM DUAL;
```

⇒ 마지막 열이 My SQL이다.

함수명	설명	예제	MYSQL 확인
UPPER(문자열)	대문자로 변환하여 반환	select upper(ENAME), lower(ENAME), INITCAP(ENAME) FROM EMP;	O
LOWER(문자열)	소문자로 변환하여 반환		O
INITCAP(문자열)	괄호안 문자데이터중 첫글자는 대문자로, 나머지는 소문자로 변환후 반환		함수 없음!
SUBSTR(문자열데이터, 시작위치, 추출길이(선택))	문자열 데이터를 시작 위치부터 추출길이만큼 추출합니다. 추출 길이 없을경우 마지막까지 추출	select JOB, SUBSTR(JOB,2), SUBSTR(JOB,1,2) FROM EMP;	O
INSTR(문자열, 위치를 찾으려는 부분문자, 시작위치(선택), 몇번째인지지정(선택))	문자열이 어디에 포함되어있는지 확인시 사용	SELECT INSTR('HELLO WORLD','L'), INSTR('HELLO WORLD','L',5), INSTR('HELLO WORLD','L',2,2) FROM DUAL;	SELECT INSTR('FOOBAR','BAR');
REPLACE(문자열, 찾는 문자열, 대체할문자(선택))	다른문자로 바꾸는 함수!	select JOB , replace(JOB, 'A' ,'Q') FROM EMP;	대체할 문자 생략 불가.
LPAD(문자열,자리수,채울문자(선택)) RPAD(문자열,자리수,채울문자(선택))	데이터의 빈 공간을 설정한 자리수 만큼 설정한 문자로 채운다.	SELECT lpad(ENAME,10,'@'), RPAD(ENAME,10,'@') FROM EMP;	채울문자 필수입력
CONCAT(문자열,문자열)	두 문자열 데이터를 합친다.	SELECT concat(ENAME, JOB) FROM EMP;	O
TRIM(삭제옵션(선택), 삭제할문자(선택), 원본문자열 데이터 (필수))	특정 문자를 지움	SELECT trim(ENAME), trim('A' FROM ENAME), trim(LEADING 'A' FROM ENAME), trim(TRAILING 'S' FROM ENAME) FROM EMP;	O

LTRIM(원본문자열, 삭제할문자(선택)) RTRIM(원본문자열, 삭제할문자(선택))	왼쪽, 오른쪽 지정문자를 삭제	SELECT Ltrim(ENAME), Rtrim(E NAME) FROM EMP;	선택문자열 없음 공백만 삭제
ROUND(숫자, 반올림위치(선택))	특정위치에서반올림	select ROUND(SAL, 1), ROUND(S AL) FROM EMP;	O
TRUNC(숫자, 버림위치(선택))	특정위치에서 버림		TRUNCATE(숫자, 버림위치(필수))
CEIL(숫자)	지정한 숫자와 가까운 큰 정수를 찾을	SELECT CEIL(3.141592), FLOOR (3.141592);	O
FOOR(숫자)	지정한 숫자와 가까운 작은 정수		
MOD(나눗셈될숫자, 나눌숫자)	숫자를 나눈 나머지 값을 구함	SELECT MOD(3.141592,3);	O
날짜관련			
SYSDATE	날짜 데이터	SELECT SYSDATE AS NOW, SYSDATE+1 AS TOMORROW,SYSDATE -1 AS YESTERDAY FROM DUAL;	SELECT SYSDATE(), NOW() , NOW()+1, NOW()-1;
ADD_MONTH((날짜데이터), 더할 개월수)	몇개월 이후의 날짜를 구함	SELECT sysdate, ADD_MONTHS(SYSDATE,3) FROM DUAL;	없음 DATE_ADD사용 DATEADD(NOW(), INTERVAL 1 MONTH)
*mysql 함수 DATE_ADD(date, INTERVAL expr unit)	시간 더하기	*MYSQL select date_add(now(), interval 1 day), date_add(now(), interv al 1 hour), date_add('20211018', interval 1 month);	
*mysql 함수 DATE_SUB(date, INTERVAL expr unit)	시간빼기	*MYSQL select date_sub(now(), interval 1 day), date_sub(now(), interva l 1 hour), date_sub('20211018', interval 1 month);	

MONTHS_BETWEEN(날짜 데이터1, 날짜 데이터 2)	두 날짜 간의 개월수 차이를 구함.	select months_between(hiredate,sysdate) from emp;	없음
*mysql 함수 DATEDIFF(날짜데이터1, 날짜데이터2)	두 날짜 사이의 차이를 구함	*MYSQL SELECT datediff(NOW(),'20220620')	
NEXT_DAY(날짜데이터, 요일문자)	돌아오는 요일,달 마지막 날짜를 구함	SELECT SYSDATE, NEXT_DAY(SYSDATE,'월요일'), LAST_DAY(SYSDATE) FROM DUAL	없음
LAST_DAY(날짜데이터)	속한 달의 마지막 날짜를 출력		
ROUND(날짜, 포맷)	날짜 반올림	SELECT round(NOW() , 'CC'), round(NOW() , 'YYYY'),round(NOW() , 'Q'),round(NOW() , 'DDD'),round(NOW() , 'HH'), TRUNC(NOW() , 'CC'), TRUNC(NOW() , 'YYYY'),TRUNC(NOW() , 'Q'),TRUNC(NOW() , 'DDD'),TRUNC(NOW() , 'HH');	SELECT round(NOW() , 'CC'), round(NOW() , 'YYYY'),round(NOW() , 'Q'),round(NOW() , 'DDD'),round(NOW() , 'HH'), TRUNCATE(NOW() , 'CC'), TRUNCATE(NOW() , 'YYYY'),TRUNCATE(NOW() , 'Q'),TRUNCATE(NOW() , 'DDD'),TRUNCATE(NOW() , 'HH');
TRUNC(날짜, 포맷)	날짜 버림		
TO_CHAR		SELECT TO_CHAR(SYSDATE(), 'YYYY/MM/DD HH24:MI:SS')	없음
*MY SQL DATE_FORMAT(날짜, 포맷)	날짜 , 숫자데이터를 문자 데이터로 변환	*MYSQL SELECT DATE_FORMAT(SYSDATE(), '%Y-%M-%D');	
TO_NUMBER		SELECT TO_NUMBER('1') FROM DUAL;	CAST 함수써야함

TO_NUMBER		SELECT TO_NUMBER('1') FROM DUAL;	CAST 함수써야함
*MYSQL CAST(문자열 AS INTEGER)	문자데이터를 숫자데이터로 변환	*MYSQL SELECT CAST('1' AS INTEGER) AS NUM;	
TO_DATE	문자데이터를 날짜 데이터로 변환	TO_DATE	X DATE_FORMAT 함수 쓸것
NVL()	열또는 데이터를 입력하여 해당데이터가 NULL이 아닐 경우데이터를 그대로 반환하고, NULL인 경우지정한 데이터를 반환.	SELECT NVL(COMM,0) FROM EMP;	X IFNULL 함수 쓸것
NVL2()	데이터가 NULL이 아닐때 반환데이터를 추가로 지정가능	SELECT NVL2(COMM, '0','X') FROM EMP	X
*MYSQL IFNULL(비교군,'리턴내용')	NULL 일경우 사용자 지정내용 리턴	*MYSQL SELECT IFNULL(NAME,'값이없습니다') FROM TEST	

DECODE	<p>DECODE([검사대상이 될 열 또는 데이터, 연산이나 함수의 결과],[조건1],[데이터가 조건1과 일치할때 반환결과], [조건2],[데이터가 조건2와 일치할때 반환할 결과],</p> <p>.....</p> <p>[조건N],[데이터가 조건N와 일치할때 반환할 결과]</p> <p>],</p> <p>[위조건과 일치한 경우가 없을때 반환할 결과])</p> <p>IF나 /SWITCH-CASE 조건문과 같은 이 데이터를 조건에부합하는지 비교하고 부합하는 경우 설정한 결과를 반환한다.</p>	<p>SELECT</p> <p>EMPNO, ENAME, JOB, SAL,</p> <p>DECODE(</p> <p>JOB</p> <p>, 'MANAGER', SAL*1.1,</p> <p>'SALESMAN', SAL*1.1,</p> <p>'ANALYST', SAL</p> <p>,SAL* 1.03)</p> <p>FROM EMP;</p>	<p>없음 IF나 CASE문 쓸것</p> <p>ENCODE(문자열,키)</p> <p>암호화 함수</p> <p>DECODE(문자열, 키)</p> <p>복호화 함수</p> <p>사용 용도가 다름</p>
--------	--	---	--

CASE	<p>조건을 비교하되 각 조건에 사용하는 데이터가 서로 상관없어도 비교가능, 기준데이터 값이 같은 데이터외에 다양한 조건을 사용할 수있다.</p> <p>CASE [비교할 데이터] WHEN [조건1] THEN [조건1 True 반환할 결과] WHEN [조건2] THEN [조건2 True 반환할 결과] WHEN [조건3] THEN [조건3 True 반환할 결과] ... WHEN [조건n] THEN [조건n True 반환할 결과] ELSE [위조건에 일치하는경우가 없을때 바나한 결과] END</p>	<p>SELECT EMPNO, CASE JOB WHEN 'MANAGER' THEN SAL*1.1 WHEN 'SALESMAN' THEN SAL*1.05 ELSE SAL*1.03 END FROM EMP;</p>	O
* MYSQL IF문	<p>오라클 함수의 decode함수는 조건과 같을 겨우만 비교하지만 , 조건문을 적어 줄수있다.</p> <p>IF (조건, 조건이 일치할경우 VALUE, 조건이 일치하지 않을 경우 VALUE)</p>	<p>*MY SQL SELECT deptno, IF(deptno <= 20,'RESEARCH','OPERATIONS') deptno as "Dept Name" FROM dept;</p>	
다중행함수			

SUM(해당열)	합계	SELECT COUNT(SAL), COUNT(ALL SAL),COUNT(distinct SAL), SUM(SAL), MAX(SAL), MIN(SAL), AVG(SAL) FROM EMP;
COUNT(해당열)	데이터의 갯수를 구해줌	
MAX(),MIN()	최대값 , 최소값	
AVG()	평균값	

1	/* 반올림 ROUND
2	
3	ROUND(대상, 반올림 위치)
4	반올림 위치가 소수점 아래에서는 양수로 그 자리까지로 반올림
5	반올림 위치가 소수점 위에는 (즉 정수자리에서는) 음수로 그 자리에서 반올림
6	*/
7	SELECT '1234.5678', ROUND(1234.5678, 1), ROUND(1234.5678, -1), ROUND(1236.5678, -1)
8	FROM DUAL;

Result #1 (1r × 4c)			
1234.5678	ROUND(1234.5678,1)	ROUND(1234.5678,-1)	ROUND(1236.5678,-1)
1234.5678	1,234.6	1,230	1,240

반올림에 대해서.

```

/* 반올림 ROUND
ROUND(대상, 반올림 위치)
반올림 위치가 소수점 아래에서는 양수로 그 자리까지로 반올림
반올림 위치가 소수점 위에는 (즉 정수자리에서는) 음수로 그 자리에서 반올림
*/
SELECT '1234.5678', ROUND(1234.5678, 1), ROUND(1234.5678, -1), ROUND(1236.5678, -1)
FROM DUAL;

```

⇒ 반올림 : My SQL뿐만 아니라 컴퓨터 내에서는 동일한 개념으로 반올림이 적용된다.

876.567 숫자를 출력하는데 소수점 두 번째 자리인 6에서 반올림해서 출력해 보겠습니다.

Quiz 23.

1	SELECT '876.567', ROUND(876.567, 1)
2	FROM DUAL;

Result #1 (1r × 2c)	
876.567	ROUND(876.567,1)
876.567	876.6

```

SELECT '876.567', ROUND(876.567, 1)
FROM DUAL;

```

```

1  /* 버림 TRUNC , TRUNCATE(My SQL)
2  -> My SQL에서는 TRUNCATE 로 사용 !
3
4  TRUNCATE(대상, 버림 위치)
5  버림 위치가 소수점 아래에서는 양수로 그 자리까지 남기고 버림
6  버림 위치가 소수점 위에는 (즉 정수 자리에서는) 음수로 그 자리에서 버림
7  */
8
9  SELECT '1234.5678' AS 숫자 , TRUNCATE(1234.5678,1), TRUNCATE(1234.5678,-2)
10 FROM DUAL;

```

Result #1 (1r x 3c)

숫자	TRUNCATE(1234.5678,1)	TRUNCATE(1234.5678,-2)
1234.5678	1,234.5	1,200

버리기에 대해서.

```

/* 버림 TRUNC , TRUNCATE(My SQL)
-> My SQL에서는 TRUNCATE 로 사용!

```

TRUNCATE(대상, 버림위치)

버림 위치가 소수점 아래에서는 양수로 그 자리까지 남기고 버림

버림 위치가 소수점 위에는 (즉 정수 자리에서는) 음수로 그 자리에서 버림

*/

```

SELECT '1234.5678' AS 숫자, TRUNCATE(1234.5678,1), TRUNCATE(1234.5678, -2)

```

사원 번호와 사원 번호가 홀수이면 1, 짝수이면 0을 출력

Quiz 25.

7839	1
7698	0
7782	0
:	:

1	/* 나머지 함수 MOD()
2	
3	SELECT MOD(대상 숫자, 나눌 숫자)
4	FROM DUAL;
5	*/
6	
7	SELECT EMPNO,MOD(EMPNO,2)
8	FROM emp;

emp (14r × 2c)	
EMPNO	MOD(EMPNO,2)
7,839	1
7,698	0
7,782	0
7,566	0
7,654	0
7,499	1
7,844	0
7,900	0
7,521	1
7,902	0
7,369	1
7,788	0
7,876	0
7,934	0

```

/* 나머지 함수 MOD()
SELECT MOD(대상 숫자, 나눌 숫자)
FROM DUAL;
*/
SELECT EMPNO, (MOD(EMPNO,2))
FROM emp;

```

사원 번호가 짝수인 사원들의 사원 번호와 이름을 출력하는 쿼리

EMPNO	ENAME
7698	BLAKE
7782	CLARK
7566	JONES
7654	MARTIN
:	:

1	/* 나머지 함수 MOD()
2	
3	SELECT MOD(대상 숫자, 나눌 숫자)
4	FROM DUAL;
5	*/
6	
7	SELECT EMPNO,ENAME
8	FROM emp
9	WHERE MOD(empno,2)=0;

emp (10r × 2c)	
EMPNO	ENAME
7,698	BLAKE
7,782	CLARK
7,566	JONES
7,654	MARTIN
7,844	TURNER
7,900	JAMES
7,902	FORD
7,788	SCOTT
7,876	ADAMS
7,934	MILLER

```

/* 나머지 함수 MOD()

SELECT MOD(대상 숫자, 나눌 숫자)
  FROM DUAL;
*/

SELECT EMPNO,ENAME
  FROM emp
 WHERE MOD(empno,2)=0;

```

다음의 쿼리는 10을 3으로 나눈 몫을 출력하는 쿼리입니다.

FLOOR(10/3)

3

1	SELECT FLOOR(10/3)
2	FROM DUAL;
Result #1 (1r x 1c)	
FLOOR(10/3)	3

근데 잘 안쓴다.

```
SELECT FLOOR(10/3)
FROM DUAL;
```

```

1  /* CASE 구문
2
3      CASE
4          WHEN sal>=3000 THEN 500
5          WHEN sal>=2000 THEN 300
6          WHEN sal>=1000 THEN 200
7          ELSE 0
8      END AS Bouns
9
10 */
11
12 SELECT ename,job,sal,
13        CASE
14            WHEN sal>=3000 THEN 500
15            WHEN sal>=2000 THEN 300
16            WHEN sal>=1000 THEN 200
17            ELSE 0
18        END AS Bouns
19 FROM emp;

```

emp (14r × 4c)			
ename	job	sal	Bouns
KING	PRESIDENT	5,000	500
BLAKE	MANAGER	2,850	300
CLARK	MANAGER	2,450	300
JONES	MANAGER	2,975	300
MARTIN	SALESMAN	1,250	200
ALLEN	SALESMAN	1,600	200
TURNER	SALESMAN	1,500	200
JAMES	CLERK	950	0
WARD	SALESMAN	1,250	200
FORD	ANALYST	3,000	500
SMITH	CLERK	800	0
SCOTT	ANALYST	3,000	500
ADAMS	CLERK	1,100	200
MILLER	CLERK	1,300	200

```

/* CASE 구문

CASE
    WHEN sal>=3000 THEN 500
    WHEN sal>=2000 THEN 300
    WHEN sal>=1000 THEN 200
    ELSE 0
END AS Bouns

*/

SELECT ename,job,sal,
       CASE
           WHEN sal>=3000 THEN 500
           WHEN sal>=2000 THEN 300
           WHEN sal>=1000 THEN 200
           ELSE 0
       END AS Bouns
FROM emp;

```

다음의 쿼리는 보너스를 출력할 때 직업이 SALESMAN, ANALYST이면 500을 출력하고, 직업이 CLERK, MANAGER이면 400을 출력하고, 나머지 직업은 0을 출력하는 쿼리입니다.

ENAME	JOB	보너스
KING	PRESIDENT	0
BLAKE	MANAGER	400
CLARK	MANAGER	400
JONES	MANAGER	400
MARTIN	SALESMAN	500
ALLEN	SALESMAN	500
TURNER	SALESMAN	500
JAMES	CLERK	400
WARD	SALESMAN	500
FORD	ANALYST	500
SMITH	CLERK	400
SCOTT	ANALYST	500
ADAMS	CLERK	400
MILLER	CLERK	400

```
SELECT ename, job, CASE WHEN job in ('SALESMAN','ANALYST') THEN 500
                        WHEN job in ('CLERK','MANAGER') THEN 400
                        ELSE 0 END as 보너스
FROM emp;
```



```

12 SELECT ename,job,
13        CASE
14            WHEN job IN('SALESMAN','ANALYST') THEN 500
15            WHEN job IN('CLERK','MANAGER') THEN 400
16            ELSE 0
17        END AS 보너스
18 FROM emp;

```

emp (14r x 3c)

ename	job	보너스
KING	PRESIDENT	0
BLAKE	MANAGER	400
CLARK	MANAGER	400
JONES	MANAGER	400
MARTIN	SALESMAN	500
ALLEN	SALESMAN	500
TURNER	SALESMAN	500
JAMES	CLERK	400
WARD	SALESMAN	500
FORD	ANALYST	500
SMITH	CLERK	400
SCOTT	ANALYST	500
ADAMS	CLERK	400
MILLER	CLERK	400

```

SELECT ename,job,
CASE
WHEN job IN('SALESMAN','ANALYST') THEN 500
WHEN job IN('CLERK','MANAGER') THEN 400
ELSE 0
END AS 보너스
FROM emp;

```