

## Chap07. 다중행 함수와 데이터 그룹화

- ▶ 07-1 하나의 열에 출력 결과를 담는 다중행 함수
- ▶ 07-2 결과 값을 원하는 열로 묶어 출력하는  
GROUP BY 절
- ▶ 07-3 GROUP BY절에 조건을 줄 때 사용하는  
HAVING절
- ▶ 07-4 그룹화와 관련된 여러 함수

## 07-1 하나의 열에 출력 결과를 담는 다중행 함수

- 
- ▶ **SUM** : 합계
  - ▶ **COUNT** : 데이터 개수
  - ▶ **MAX** : 최댓값
  - ▶ **MIN** : 최솟값
  - ▶ **AVG** : 평균값
- 



# 07-1 하나의 열에 출력 결과를 담는 다중행 함수

실습 7-1 SUM 함수를 사용하여 급여 합계 출력하기

```
01 SELECT SUM(SAL)
02 FROM EMP;
```

:: 결과 화면

SUM(SAL)
29025

실습 7-2 SUM 함수를 사용하여 사원 이름과 급여 합계 출력하기

```
01 SELECT ENAME, SUM(SAL)
02 FROM EMP;
```

:: 결과 화면

Errors
[Error] Execution (1: 8): ORA-00937: 단일 그룹의 그룹 함수가 아닙니다

```
SELECT SAL
FROM EMP;
```

SAL
800
1600
1250
2975
1250
2850
2450
3000
5000
1500
1100
950
3000
1300



```
SELECT SUM(SAL)
FROM EMP;
```

SUM(SAL)
29025

# 07-1 하나의 열에 출력 결과를 담는 다중행 함수

## ▶ SUM : 합계

SUM([DISTINCT, ALL 중 하나를 선택하거나 아무 값도 지정하지 않음(선택)])  
< [합계를 구할 열이나 연산자, 함수를 사용한 데이터(필수)]

기본 형식

번호	설명
①	합계를 구할 데이터를 지정합니다.

### 실습 7-3 추가 수당 합계 구하기

```
01 SELECT SUM(COMM)
02 FROM EMP;
```

:: 결과 화면

SUM(COMM)
2200

### 실습 7-4 급여 합계 구하기(DISTINCT, ALL 사용)

```
01 SELECT SUM(DISTINCT SAL),
02        SUM(ALL SAL),
03        SUM(SAL)
04 FROM EMP;
```

:: 결과 화면

SUM(DISTINCT SAL)	SUM(ALLSAL)	SUM(SAL)
24775	29025	29025

# 07-1 하나의 열에 출력 결과를 담는 다중행 함수

## ▶ COUNT : 데이터 개수

COUNT([DISTINCT, ALL 중 하나를 선택하거나 아무 값도 지정하지 않음(선택)]  
[개수를 구할 열이나 연산자, 함수를 사용한 데이터(필수)])  
OVER(분석을 위한 여러 문법 지정)(선택)

기본 형식

1

번호	설명
1	결과 행의 개수를 출력합니다.

### 실습 7-5 EMP 테이블의 데이터 개수 출력하기

```
01 SELECT COUNT(*)
02 FROM EMP;
```

:: 결과 화면

COUNT(*)
14

### 실습 7-6 부서 번호가 30번인 직원 수 구하기

```
01 SELECT COUNT(*)
02 FROM EMP
03 WHERE DEPTNO = 30;
```

:: 결과 화면

COUNT(*)
6

# 07-1 하나의 열에 출력 결과를 담는 다중행 함수

## ▶ COUNT : 데이터 개수

실습 7-7 COUNT 함수를 사용하기 급여 개수 구하기(DISTINCT, ALL 사용)

```
01 SELECT COUNT(DISTINCT SAL),  
02        COUNT(ALL SAL),  
03        COUNT(SAL)  
04 FROM EMP;
```



세 번째 열처럼 옵션을 지정 안 하면 ALL이 기본입니다.

:: 결과 화면

COUNT(DISTINCTSAL)	COUNT(ALLSAL)	COUNT(SAL)
12	14	14

실습 7-8 COUNT 함수를 사용하여 추가 수당 열 개수 출력하기

```
01 SELECT COUNT(COMM)  
02 FROM EMP;
```

실습 7-9 COUNT 함수와 IS NOT NULL을 사용하여 추가 수당 열 개수 출력하기

```
01 SELECT COUNT(COMM)  
02 FROM EMP  
03 WHERE COMM IS NOT NULL;
```

:: 결과 화면(실습 7-8, 7-9의 실행 결과가 같음)

COUNT(COMM)
4

# 07-1 하나의 열에 출력 결과를 담는 다중행 함수

## ▶ MAX : 최댓값

MAX([DISTINCT, ALL 중 하나를 선택하거나 아무 값도 지정하지 않음(선택)]  
[최댓값을 구할 열이나 연산자, 함수를 사용한 데이터(필수)])  
OVER(분석을 위한 여러 문법 지정)(선택)

기본 형식



번호	설명
①	결과 행에서 최댓값을 출력합니다.

실습 7-10 부서 번호가 10번인 직원들의 최대 급여 출력하기

```
01 SELECT MAX(SAL)
02 FROM EMP
03 WHERE DEPTNO = 10;
```

:: 결과 화면

MAX(SAL)
5000

실습 7-12 부서 번호가 20인 직원의 입사일 중 제일 최근 입사일 출력하기

```
01 SELECT MAX(HIREDATE)
02 FROM EMP
03 WHERE DEPTNO = 20;
```



입사 연도가 제일 큰 직원이  
최근입니다.

:: 결과 화면

MAX(HIREDATE)
1987/05/23

# 07-1 하나의 열에 출력 결과를 담는 다중행 함수

## ▶ MIN : 최솟값

MIN([DISTINCT, ALL 중 하나를 선택하거나 아무 값도 지정하지 않음(선택)]  
[최솟값을 구할 열이나 연산자, 함수를 사용한 데이터(필수)])  
OVER(분석을 위한 여러 문법을 지정)(선택)

기본 형식

①

번호	설명
①	결과 행에서 최솟값을 출력합니다.

실습 7-11 부서 번호가 10번인 직원들의 최소 급여 출력하기

```
01 SELECT MIN(SAL)
02 FROM EMP
03 WHERE DEPTNO = 10;
```

:: 결과 화면

MIN(SAL)
1300

실습 7-13 부서 번호가 20인 직원의 입사일 중 제일 오래된 입사일 출력하기

```
01 SELECT MIN(HIREDATE)
02 FROM EMP
03 WHERE DEPTNO = 20;
```



입사한 연도가 제일 작은 직원이 입사일이  
제일 오래된 사람입니다.

:: 결과 화면

MIN(HIREDATE)
1980/12/17



# 07-1 하나의 열에 출력 결과를 담는 다중행 함수

## ▶ **AVG** : 평균값

AVG([DISTINCT, ALL 중 하나를 선택하거나 아무 값도 지정하지 않음(선택)]  
[평균 값을 구할 열이나 연산자, 함수를 사용한 데이터(필수)])  
OVER(분석을 위한 여러 문법을 지정)(선택)

기본 형식

1

번호	설명
1	결과 행의 평균 값을 반환합니다.

### 실습 7-14 부서 번호가 30인 직원들의 평균 급여 출력하기

```
01 SELECT AVG(SAL)
02    FROM EMP
03    WHERE DEPTNO = 30;
```

:: 결과 화면

AVG(SAL)
1566.66666666667

# 07-2 결과 값을 원하는 열로 묶어 출력하는 GROUP BY 절

## 실습 7-16 집합 연산자를 사용하여 각 부서별 평균 급여 출력하기

```
01 SELECT AVG(SAL), '10' AS DEPTNO FROM EMP WHERE DEPTNO = 10
02 UNION ALL
03 SELECT AVG(SAL), '20' AS DEPTNO FROM EMP WHERE DEPTNO = 20
04 UNION ALL
05 SELECT AVG(SAL), '30' AS DEPTNO FROM EMP WHERE DEPTNO = 30;
```

:: 결과 화면

AVG(SAL)	DEPTNO
2916.66666666667	10
2175	20
1566.66666666667	30

# 07-2 결과 값을 원하는 열로 묶어 출력하는 GROUP BY 절

## ▶ GROUP BY 절의 기본 사용법

```
SELECT  [조회할 열1 이름], [열2 이름], ..., [열N 이름]
FROM    [조회할 테이블 이름]
WHERE   [조회할 행을 선별하는 조건식]
GROUP BY [그룹화할 열을 지정(여러 개 지정 가능)]
ORDER BY [정렬하려는 열 지정]
```

번호	키워드	필수 요소	선택 요소	설명
1	GROUP BY	그룹화할 열 또는 데이터 지정	-	특정 열 또는 데이터를 기준으로 데이터를 그룹으로 묶습니다.

## ▶ GROUP BY절을 사용할 때 유의점

 GROUP BY절에는 별칭이 인식되지 않습니다. 즉 열 이름이나 연산식을 그대로 지정해 주어야 합니다.

# 07-2 결과 값을 원하는 열로 묶어 출력하는 GROUP BY 절

## 실습 7-17 GROUP BY를 사용하여 부서별 평균 급여 출력하기

```
01 SELECT AVG(SAL), DEPTNO
02    FROM EMP
03 GROUP BY DEPTNO;
```

:: 결과 화면

AVG(SAL)	DEPTNO
1566.66666666667	30
2175	20
2916.66666666667	10

## 실습 7-18 부서 번호 및 직책별 평균 급여로 정렬하기

```
01 SELECT DEPTNO, JOB, AVG(SAL)
02    FROM EMP
03 GROUP BY DEPTNO, JOB
04 ORDER BY DEPTNO, JOB;
```

:: 결과 화면

DEPTNO	JOB	AVG(SAL)
10	CLERK	1300
10	MANAGER	2450
10	PRESIDENT	5000
20	ANALYST	3000
20	CLERK	950
20	MANAGER	2975
30	CLERK	950
30	MANAGER	2850
30	PRESIDENT	1400

## 실습 7-19 GROUP BY절에 없는 열을 SELECT절에 포함했을 경우

```
01 SELECT ENAME, DEPTNO, AVG(SAL)
02    FROM EMP
03 GROUP BY DEPTNO;
```



GROUP BY절을 사용할 때에는 SELECT절의 열도 유심히 살펴보세요.

:: 결과 화면

Errors

[Error] Execution (1: 8): ORA-00979: GROUP BY 표현식이 아닙니다.

## 07-3 GROUP BY절에 조건을 줄 때 사용하는 HAVING절

### ▶ HAVING 절의 기본 사용법

SELECT     [조회할 열1 이름], [열2 이름], ..., [열N 이름]  
FROM       [조회할 테이블 이름]  
WHERE       [조회할 행을 선별하는 조건식]  
GROUP BY   [그룹화할 열 지정(여러 개 지정 가능)]  
**HAVING**    [출력 그룹을 제한하는 조건식]  
ORDER BY   [정렬하려는 열 지정];

번호	키워드	필수 요소	선택 요소	설명
①	HAVING	조건식	-	GROUP BY절을 사용해 그룹화된 결과 중 출력 그룹을 선별하는 조건식을 지정합니다.

# 07-3 GROUP BY절에 조건을 줄 때 사용하는 HAVING절

## 실습 7-20 GROUP BY절과 HAVING절을 사용하여 출력하기

```
01 SELECT DEPTNO, JOB, AVG(SAL)
02   FROM EMP
03  GROUP BY DEPTNO, JOB
04   HAVING AVG(SAL) >= 2000
05  ORDER BY DEPTNO, JOB;
```

:: 결과 화면

DEPTNO	JOB	AVG(SAL)
10	CLERK	1300
10	MANAGER	2450
10	PRESIDENT	5000
20	ANALYST	3000
20	CLERK	950
20	MANAGER	2975
30	CLERK	950
30	MANAGER	2850
30	SALESMAN	1400

HAVING절을 추가하지 않았을 때

DEPTNO	JOB	AVG(SAL)
10	MANAGER	2450
10	PRESIDENT	5000
20	ANALYST	3000
20	MANAGER	2975
30	MANAGER	2850

HAVING절을 추가했을 때

# 07-3 GROUP BY절에 조건을 줄 때 사용하는 HAVING절

## ▶ HAVING절을 사용할 때 유의점

실습 7-21 HAVING절 대신 WHERE절을 잘못 사용했을 경우

```
01 SELECT DEPTNO, JOB, AVG(SAL)
02 FROM EMP
03 WHERE AVG(SAL) >= 2000
04 GROUP BY DEPTNO, JOB
05 ORDER BY DEPTNO, JOB;
```

:: 결과 화면

Errors
[Error] Execution (17: 8): ORA-00934: 그룹 함수는 허가되지 않습니다

실습 7-22 WHERE절을 사용하지 않고 HAVING절만 사용한 경우

```
01 SELECT DEPTNO, JOB, AVG(SAL)
02 FROM EMP
03 GROUP BY DEPTNO, JOB
04 HAVING AVG(SAL) >= 2000
05 ORDER BY DEPTNO, JOB;
```

:: 결과 화면

DEPTNO	JOB	AVG(SAL)
10	MANAGER	2450
10	PRESIDENT	5000
20	ANALYST	3000
30	MANAGER	2975
30	MANAGER	2850

## ▶ WHERE절과 HAVING절의 차이점

실습 7-23 WHERE절과 HAVING절을 모두 사용한 경우

```
01 SELECT DEPTNO, JOB, AVG(SAL)
02 FROM EMP
03 WHERE SAL <= 3000
04 GROUP BY DEPTNO, JOB
05 HAVING AVG(SAL) >= 2000
06 ORDER BY DEPTNO, JOB;
```

WHERE절이 GROUP BY절, HAVING절보다 먼저 실행됩니다.

:: 결과 화면

DEPTNO	JOB	AVG(SAL)
10	MANAGER	2450
20	ANALYST	3000
20	MANAGER	2975
30	MANAGER	2850

## 07-4 그룹화와 관련된 여러 함수

---

- ▶ ROLLUP, CUBE : 그룹화 데이터의 합계
- ▶ GROUPING SETS : 지정한 각 열별 그룹화
- ▶ GROUPING : ROLLUP, CUBE와 함께 사용 (하나)
- ▶ GROUPING\_ID : ROLLUP, CUBE와 함께 사용 (여럿)





## 07-4 그룹화와 관련된 여러 함수

### ▶ ROLLUP, CUBE : 그룹화 데이터의 집계

```
SELECT [조회할 열1 이름], [열2 이름], ..., [열N 이름]
FROM   [조회할 테이블 이름]
WHERE  [조회할 행을 선별하는 조건식]
GROUP BY ROLLUP [그룹화 열 지정(여러 개 지정 가능)]; -①
```

기본 형식

```
SELECT [조회할 열1 이름], [열2 이름], ..., [열N 이름]
FROM   [조회할 테이블 이름]
WHERE  [조회할 행을 선별하는 조건식]
GROUP BY CUBE [그룹화 열 지정(여러 개 지정 가능)]; -②
```

기본 형식

번호	키워드	필수 요소	선택 요소	설명
①, ②	ROLLUP, CUBE	그룹화 열 지정	-	그룹화 데이터의 집계를 함께 출력하는 데 사용합니다.

# 07-4 그룹화와 관련된 여러 함수

## ▶ ROLLUP

실습 7-24 기존 GROUP BY절만 사용한 그룹화

```
01 SELECT DEPTNO, JOB, COUNT(*), MAX(SAL), SUM(SAL), AVG(SAL)
02    FROM EMP
03    GROUP BY DEPTNO, JOB
04    ORDER BY DEPTNO, JOB;
```

:: 결과 화면

DEPTNO	JOB	COUNT(*)	MAX(SAL)	SUM(SAL)	AVG(SAL)
10	CLERK	1	1300	1300	1300
10	MANAGER	1	2450	2450	2450
10	PRESIDENT	1	5000	5000	5000
20	ANALYST	2	3000	6000	3000
20	CLERK	2	1100	1900	950
20	MANAGER	1	2975	2975	2975
30	CLERK	1	950	950	950
30	MANAGER	1	2850	2850	2850
30	SALESMAN	4	1600	5600	1400

실습 7-25 ROLLUP 함수를 적용한 그룹화

```
01 SELECT DEPTNO, JOB, COUNT(*), MAX(SAL), SUM(SAL), AVG(SAL)
02    FROM EMP
03    GROUP BY ROLLUP(DEPTNO, JOB);
```

:: 결과 화면

DEPTNO	JOB	COUNT(*)	MAX(SAL)	SUM(SAL)	AVG(SAL)
10	CLERK	1	1300	1300	1300
10	MANAGER	1	2450	2450	2450
10	PRESIDENT	1	5000	5000	5000
10		3	5000	8750	2916.66666666667
20	CLERK	2	1100	1900	950
20	ANALYST	2	3000	6000	3000
20	MANAGER	1	2975	2975	2975
20		5	3000	10875	2175
30	CLERK	1	950	950	950
30	MANAGER	1	2850	2850	2850
30	SALESMAN	4	1600	5600	1400
30		6	2850	9400	1566.66666666667
		14	5000	29025	2073.21428571429

# 07-4 그룹화와 관련된 여러 함수

## ▶ CUBE

실습 7-26 CUBE 함수를 적용한 그룹화

```
01 SELECT DEPTNO, JOB, COUNT(*), MAX(SAL), SUM(SAL), AVG(SAL)
02     FROM EMP
03     GROUP BY CUBE(DEPTNO, JOB)
04     ORDER BY DEPTNO, JOB;
```

:: 결과 화면

DEPTNO	JOB	COUNT(*)	MAX(SAL)	SUM(SAL)	AVG(SAL)
10	CLERK	1	1300	1300	1300
10	MANAGER	1	2450	2450	2450
10	PRESIDENT	1	5000	5000	5000
10		3	5000	8750	2916.66666666667
20	ANALYST	2	3000	6000	3000
20	CLERK	2	1100	1900	950
20	MANAGER	1	2975	2975	2975
20		5	3000	10875	2175
30	CLERK	1	950	950	950
30	MANAGER	1	2850	2850	2850
30	SALESMAN	4	1600	5600	1400
30		6	2850	9400	1566.66666666667
	ANALYST	2	3000	6000	3000
	CLERK	4	1300	4150	1037.5
	MANAGER	3	2975	8275	2758.33333333333
	PRESIDENT	1	5000	5000	5000
	SALESMAN	4	1600	5600	1400
		14	5000	29025	2073.21428571429

# 07-4 그룹화와 관련된 여러 함수

## ▶ ROLLUP VS CUBE

ROLLUP(A, B, C)

1. A 그룹별 B 그룹별 C 그룹에 해당하는 결과 출력
2. A 그룹별 B 그룹에 해당하는 결과 출력
3. A 그룹에 해당하는 결과 출력
4. 전체 데이터 결과 출력

CUBE(A, B, C)

1. A 그룹별 B 그룹별 C 그룹에 해당하는 결과 출력
2. A 그룹별 B 그룹의 결과 출력
3. B 그룹별 C 그룹의 결과 출력
4. A 그룹별 C 그룹의 결과 출력
5. A 그룹 결과
6. B 그룹 결과
7. C 그룹 결과
8. 전체 데이터 결과

# 07-4 그룹화와 관련된 여러 함수

## ▶ ROLLUP VS CUBE

실습 7-27 DEPTNO를 먼저 그룹화한 후 ROLLUP 함수에 JOB 지정하기

```
01 SELECT DEPTNO, JOB, COUNT(*)
02    FROM EMP
03    GROUP BY DEPTNO, ROLLUP(JOB);
```

:: 결과 화면

DEPTNO	JOB	COUNT(*)
10	CLERK	1
10	MANAGER	1
10	PRESIDENT	1
10		3
20	CLERK	2
20	ANALYST	2
20	MANAGER	1
20		5
30	CLERK	1
30	MANAGER	1
30	SALESMAN	4
30		6

실습 7-28 JOB을 먼저 그룹화한 후 ROLLUP 함수에 DEPTNO 지정하기

```
01 SELECT DEPTNO, JOB, COUNT(*)
02    FROM EMP
03    GROUP BY JOB, ROLLUP(DEPTNO);
```

:: 결과 화면

DEPTNO	JOB	COUNT(*)
10	CLERK	1
20	CLERK	2
30	CLERK	1
	CLERK	4
20	ANALYST	2
	ANALYST	2
10	MANAGER	1
20	MANAGER	1
30	MANAGER	1
	MANAGER	3
30	SALESMAN	4
	SALESMAN	4
10	PRESIDENT	1
	PRESIDENT	1

## 07-4 그룹화와 관련된 여러 함수

### ▶ GROUPING SETS : 지정한 각 열별 그룹화

```
SELECT [조회할 열1 이름], [열2 이름], ..., [열N 이름]
FROM   [조회할 테이블 이름]
WHERE  [조회할 행을 선별하는 조건식]
GROUP BY GROUPING SETS [그룹화 열 지정(여러 개 지정 가능)]; -①
```

기본 형식

번호	키워드	필수 요소	선택 요소	설명
①	GROUPING SETS	그룹화 열	-	여러 그룹화 대상 열의 결과 값을 각각 같은 수준으로 출력합니다.

실습 7-29 GROUPING SETS 함수를 사용하여 열별로 그룹으로 묶어 출력하기

```
01 SELECT DEPTNO, JOB, COUNT(*)
02    FROM EMP
03    GROUP BY GROUPING SETS(DEPTNO, JOB)
04    ORDER BY DEPTNO, JOB;
```

:: 결과 화면

DEPTNO	JOB	COUNT(*)
10		3
20		5
30		6
	ANALYST	2
	CLERK	4
	MANAGER	3
	PRESIDENT	1
	SALESMAN	4

## 07-4 그룹화와 관련된 여러 함수

### ▶ GROUPING : ROLLUP, CUBE와 함께 사용 (하나)

SELECT [조회할 열1 이름], [열2 이름], ..., [열N 이름]

기본 형식

**GROUPING** [GROUP BY절에 ROLLUP 또는 CUBE에 명시한 그룹화 할 열 이름]

FROM [조회할 테이블 이름]

WHERE [조회할 행을 선별하는 조건식]

GROUP BY ROLLUP 또는 CUBE [그룹화할 열]; - ①

번호	키워드	필수 요소	선택 요소	설명
①	GROUPING	그룹화 여부를 확인할 열	-	현재 결과가 그룹화 대상 열의 그룹화가 이루어진 상태의 집계인지 여부를 출력합니다.

# 07-4 그룹화와 관련된 여러 함수

## ▶ GROUPING : ROLLUP, CUBE와 함께 사용 (하나)

실습 7-30 DEPTNO, JOB열의 그룹화 결과 여부를 GROUPING 함수로 확인하기

```
01 SELECT DEPTNO, JOB, COUNT(*), MAX(SAL), SUM(SAL), AVG(SAL),  
02        GROUPING(DEPTNO),  
03        GROUPING(JOB)  
04 FROM EMP  
05 GROUP BY CUBE(DEPTNO, JOB)  
06 ORDER BY DEPTNO, JOB;
```

:: 결과 화면

DEPTNO	JOB	COUNT(*)	MAX(SAL)	SUM(SAL)	AVG(SAL)	GROUPING(DEPTNO)	GROUPING(JOB)
10	CLERK	1	1300	1300	1300	0	0
10	MANAGER	1	2450	2450	2450	0	0
10	PRESIDENT	1	5000	5000	5000	0	0
10		3	5000	8750	2916.66666666667	0	1
20	ANALYST	2	3000	6000	3000	0	0
20	CLERK	2	1100	1900	950	0	0
20	MANAGER	1	2975	2975	2975	0	0
20		5	3000	10875	2175	0	1
30	CLERK	1	950	950	950	0	0
30	MANAGER	1	2850	2850	2850	0	0
30	SALESMAN	4	1600	5600	1400	0	0
30		6	2850	9400	1566.66666666667	0	1
	ANALYST	2	3000	6000	3000	1	0
	CLERK	4	1300	4150	1037.5	1	0
	MANAGER	3	2975	8275	2758.33333333333	1	0
	PRESIDENT	1	5000	5000	5000	1	0
	SALESMAN	4	1600	5600	1400	1	0
		14	5000	29025	2073.21428571429	1	1



## 07-4 그룹화와 관련된 여러 함수

### ▶ GROUPING : ROLLUP, CUBE와 함께 사용 (하나)

실습 7-31 DECODE문으로 GROUPING 함수를 적용하여 결과 표기하기

```
01 SELECT DECODE(GROUPING(DEPTNO), 1, 'ALL_DEPT', DEPTNO) AS DEPTNO,  
02          DECODE(GROUPING(JOB), 1, 'ALL_JOB', JOB) AS JOB,  
03          COUNT(*), MAX(SAL), SUM(SAL), AVG(SAL)  
04 FROM EMP  
05 GROUP BY CUBE(DEPTNO, JOB)  
06 ORDER BY DEPTNO, JOB;
```

:: 결과 화면

DEPTNO	JOB	COUNT(*)	MAX(SAL)	SUM(SAL)	AVG(SAL)
10	ALL_JOB	3	5000	8750	2916.66666666667
10	CLERK	1	1300	1300	1300
10	MANAGER	1	2450	2450	2450
10	PRESIDENT	1	5000	5000	5000
20	ALL_JOB	5	3000	10875	2175
20	ANALYST	2	3000	6000	3000
20	CLERK	2	1100	1900	950
20	MANAGER	1	2975	2975	2975
30	ALL_JOB	6	2850	9400	1566.66666666667
30	CLERK	1	950	950	950
30	MANAGER	1	2850	2850	2850
30	SALESMAN	4	1600	5600	1400
ALL_DEPT	ALL_JOB	14	5000	29025	2073.21428571429
ALL_DEPT	ANALYST	2	3000	6000	3000
ALL_DEPT	CLERK	4	1300	4150	1037.5
ALL_DEPT	MANAGER	3	2975	8275	2758.33333333333
ALL_DEPT	PRESIDENT	1	5000	5000	5000
ALL_DEPT	SALESMAN	4	1600	5600	1400

## 07-4 그룹화와 관련된 여러 함수

### ▶ GROUPING\_ID : ROLLUP, CUBE와 함께 사용 (여럿)

```
SELECT [조회할 열1 이름], [열2 이름], ..., [열N 이름]
      GROUPING_ID [그룹화 여부를 확인할 열(여러 개 지정 가능)]
FROM   [조회할 테이블 이름] WHERE [조회할 행을 선별하는 조건식]
GROUP BY ROLLUP 또는 CUBE [그룹화 할 열]; -①
```

기본 형식

번호	키워드	필수 요소	선택 요소	설명
①	GROUPING_ID	그룹화 여부를 확인할 열	-	GROUPING 함수처럼 특정 열의 그룹화 여부를 출력할 수 있으며, 검사할 열을 여러 개 지정할 수 있습니다.

GROUPING\_ID(a, b)

그룹화 된 열	그룹화 비트 벡터	최종 결과
a, b	0 0	0
a	0 1	1
b	1 0	2
없음	1 1	3

## 07-4 그룹화와 관련된 여러 함수

### ▶ GROUPING\_ID : ROLLUP, CUBE와 함께 사용 (여럿)

실습 7-32 DEPTNO, JOB을 함께 명시한 GROUPING\_ID 함수 사용하기

```
01 SELECT DEPTNO, JOB, COUNT(*), SUM(SAL),  
02        GROUPING(DEPTNO),  
03        GROUPING(JOB),  
04        GROUPING_ID(DEPTNO, JOB)  
05 FROM EMP  
06 GROUP BY CUBE(DEPTNO, JOB)  
07 ORDER BY DEPTNO, JOB;
```

:: 결과 화면

DEPTNO	JOB	COUNT(*)	SUM(SAL)	GROUPING(DEPTNO)	GROUPING(JOB)	GROUPING_ID(DEPTNO, JOB)
10	CLERK	1	1300	0	0	0
10	MANAGER	1	2450	0	0	0
10	PRESIDENT	1	5000	0	0	0
10		3	8750	0	1	1
20	ANALYST	2	6000	0	0	0
20	CLERK	2	1900	0	0	0
20	MANAGER	1	2975	0	0	0
20		5	10875	0	1	1
30	CLERK	1	950	0	0	0
30	MANAGER	1	2850	0	0	0
30	SALESMAN	4	5600	0	0	0
30		6	9400	0	1	1
	ANALYST	2	6000	1	0	2
	CLERK	4	4150	1	0	2
	MANAGER	3	8275	1	0	2
	PRESIDENT	1	5000	1	0	2
	SALESMAN	4	5600	1	0	2
		14	29025	1	1	3

## 07-4 그룹화와 관련된 여러 함수

---

- ▶ LISTAGG : 그룹 데이터 가로 출력
- ▶ PIVOT, UNPIVOT : 행/열 바꾸어 출력



## 07-4 그룹화와 관련된 여러 함수

### ▶ LISTAGG : 그룹 데이터 가로 출력

```
SELECT ENAME  
FROM EMP  
WHERE DEPTNO = 10;
```

:: 결과 화면

ENAME
CLARK
KING
MILLER

실습 7-33 GROUP BY절로 그룹화하여 부서 번호와 사원 이름 출력하기

```
01 SELECT DEPTNO, ENAME  
02 FROM EMP  
03 GROUP BY DEPTNO, ENAME;
```

:: 결과 화면

DEPTNO	ENAME
20	JONES
30	WARD
20	SCOTT
10	KING
30	JAMES
30	ALLEN
30	MARTIN
30	BLAKE
20	FORD
20	SMITH
20	ADAMS
10	MILLER
10	CLARK
30	TURNER

## 07-4 그룹화와 관련된 여러 함수

### ▶ LISTAGG : 그룹 데이터 가로 출력

```
SELECT [조회할 열1 이름], [열2 이름], ..., [열N 이름]
      LISTAGG([나열할 열(필수)], [각 데이터를 구분하는 구분자(선택)])
      WITHIN GROUP(ORDER BY 나열할 열의 정렬 기준 열 (선택)) —①
FROM   [조회할 테이블 이름]
WHERE  [조회할 행을 선별하는 조건식];
```

기본 형식

번호	키워드	필수 요소	선택 요소	설명
①	LISTAGG ~ WITHIN GROUP	나열할 열	각 데이터를 구분하는 구분자, 지정하지 않을 경우 NULL이 기본값이 됩니다.	그룹화 데이터를 하나의 열에 가로로 나열하여 출력하는 데 사용합니다.

## 07-4 그룹화와 관련된 여러 함수

### ▶ LISTAGG : 그룹 데이터 가로 출력

실습 7-34 부서별 직원 이름을 나란히 나열하여 출력하기

```
01 SELECT DEPTNO,  
02        LISTAGG(ENAME, ', ' )  
03        WITHIN GROUP(ORDER BY SAL DESC) AS ENAMES  
04 FROM EMP  
05 GROUP BY DEPTNO;
```

:: 결과 화면

DEPTNO	ENAMES
10	KING, CLARK, MILLER
20	FORD, SCOTT, JONES, ADAMS, SMITH
30	BLAKE, ALLEN, TURNER, MARTIN, WARD, JAMES



LISTAGG 함수 사용으로 직원 이름이 훨씬 보기 편해졌습니다.

# 07-4 그룹화와 관련된 여러 함수

## ▶ PIVOT : 행/열 바꾸어 출력

실습 7-35 부서별·직책별로 그룹화하여 최고 급여 데이터 출력하기

```
01 SELECT DEPTNO, JOB, MAX(SAL)
02   FROM EMP
03  GROUP BY DEPTNO, JOB
04  ORDER BY DEPTNO, JOB;
```

:: 결과 화면

DEPTNO	JOB	MAX(SAL)
10	CLERK	1300
10	MANAGER	2450
10	PRESIDENT	5000
20	ANALYST	3000
20	CLERK	1100
20	MANAGER	2975
30	CLERK	950
30	MANAGER	2850
30	SALESMAN	1600

실습 7-36 PIVOT 함수를 사용하여 직책별·부서별 최고 급여를 2차원 표 형태로 출력하기

```
01 SELECT *
02   FROM(SELECT DEPTNO, JOB, SAL
03        FROM EMP)
04  PIVOT(MAX(SAL)
05        FOR DEPTNO IN (10, 20, 30)
06        )
07  ORDER BY JOB;
```

:: 결과 화면

JOB	10	20	30
ANALYST		3000	
CLERK	1300	1100	950
MANAGER	2450	2975	2850
PRESIDENT	5000		
SALESMAN			1600

실습 7-37 PIVOT 함수를 사용하여 부서별·직책별 최고 급여를 2차원 표 형태로 출력하기

```
01 SELECT *
02   FROM(SELECT JOB, DEPTNO, SAL
03        FROM EMP)
04  PIVOT(MAX(SAL)
05        FOR JOB IN ('CLERK' AS CLERK,
06                    'SALESMAN' AS SALESMAN,
07                    'PRESIDENT' AS PRESIDENT,
08                    'MANAGER' AS MANAGER,
09                    'ANALYST' AS ANALYST)
10        )
11  ORDER BY DEPTNO;
```

:: 결과 화면

DEPTNO	CLERK	SALESMAN	PRESIDENT	MANAGER	ANALYST
10	1300		5000	2450	
20	1100			2975	3000
30	950	1600		2850	



## 07-4 그룹화와 관련된 여러 함수

### ▶ PIVOT: 행/열 바꾸어 출력

실습 7-38 DECODE문을 활용하여 PIVOT 함수와 같은 출력 구현하기

```
01 SELECT DEPTNO,  
02        MAX(DECODE(JOB, 'CLERK', SAL)) AS "CLERK",  
03        MAX(DECODE(JOB, 'SALESMAN', SAL)) AS "SALESMAN",  
04        MAX(DECODE(JOB, 'PRESIDENT', SAL)) AS "PRESIDENT",  
05        MAX(DECODE(JOB, 'MANAGER', SAL)) AS "MANAGER",  
06        MAX(DECODE(JOB, 'ANALYST', SAL)) AS "ANALYST"  
07 FROM EMP  
08 GROUP BY DEPTNO  
09 ORDER BY DEPTNO;
```

:: 결과 화면

DEPTNO	CLERK	SALESMAN	PRESIDENT	MANAGER	ANALYST
10	1300		5000	2450	
20	1100			2975	3000
30	950	1600		2850	

## 07-4 그룹화와 관련된 여러 함수

### ▶ UNPIVOT : 행/열 바꾸어 출력

실습 7-39 UNPIVOT 함수를 사용하여 열로 구분된 그룹을 행으로 출력하기

```
01 SELECT *
02     FROM(SELECT DEPTNO,
03              MAX(DECODE(JOB, 'CLERK' , SAL)) AS "CLERK",
04              MAX(DECODE(JOB, 'SALESMAN' , SAL)) AS "SALESMAN",
05              MAX(DECODE(JOB, 'PRESIDENT', SAL)) AS "PRESIDENT",
06              MAX(DECODE(JOB, 'MANAGER' , SAL)) AS "MANAGER",
07              MAX(DECODE(JOB, 'ANALYST' , SAL)) AS "ANALYST"
08             FROM EMP
09            GROUP BY DEPTNO
10            ORDER BY DEPTNO)
11     UNPIVOT(
12       SAL FOR JOB IN (CLERK, SALESMAN, PRESIDENT, MANAGER,ANALYST))
13    ORDER BY DEPTNO, JOB;
```

:: 결과 화면

DEPTNO	JOB	SAL
10	CLERK	1300
10	MANAGER	2450
10	PRESIDENT	5000
20	ANALYST	3000
20	CLERK	1100
20	MANAGER	2975
30	CLERK	950
30	MANAGER	2850
30	SALESMAN	1600