ch07. 다중행 함수와 데이터 그룹화

07-1. 하나의 열에 출력 결과를 담는 다중행 함수

그룹 함수 또는 복수행 함수로도 불리는 다중행 함수 *multiple-row function* 는
여러 행을 바탕으로 하나의 결과 값을 도출해내기 위해 사용하는 함수.

다중행 함수

함수	설명
SUM	지정한 데이터의 합 반환
COUNT	지정한 데이터의 개수 반환
MAX	지정한 데이터 중 최댓값 반환
MIN	지정한 데이터 중 최솟값 반환
AVG	지정한 데이터의 평균값 반환

SUM 함수

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

- SUM함수를 분석하는 용도로 사용한다면 OVER절을 사용
-

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

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

데이터 개수를 구해주는 COUNT 함수

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

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

최댓값과 최솟값을 구하는 MAX, MIN 함수

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

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

- 오라클 DB에는 날짜 및 문자 데이터 역시 MAX,MIN 사용가능

평균 값을 구하는 AVG 함수

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

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

07-2. 결과 값을 원하는 열로 묶어 출력하는 GROUP BY절

GROUP BY절의 기본 사용법

여러 데이터에서 의미 있는 하나의 결과를 특정 열 값별로 묶어서 출력할 때 데이터를 '그룹화'한다고 표현한다. SELECT문에서는 GROUP BY절을 작성하여 데이터를 그룹화할 수 있다.

```
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]
```

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

```
SELECT DEPTNO, JOB, AVG(SAL)
  FROM EMP
GROUP BY DEPTNO, JOB
ORDER BY DEPTNO, JOB;
```

GROUP BY절을 사용할 때 유의점

- 다중행 함수를 사용하지 않은 일반 열은 GROUP BY절에 명시하지 않으면 SELECT절에서 사용할 수 없다는 것.

07-3. GROUP BY절에 조건을 줄 때 사용하는 HAVING절

- HAVING절은 SELECT문에 GROUP BY절이 존재할 때만 사용 가능
- 그룹화된 결과 값의 범위를 제한하는 데 사용한다.

- 각 부서의 직책별 평균 급여를 구하되 그 평균 급여가 2000 이상인 그룹만 출력

```
SELECT DEPTNO, JOB, AVG(SAL)
  FROM EMP
GROUP BY DEPTNO, JOB
  HAVING AVG(SAL) >= 2000
ORDER BY DEPTNO, JOB;
```

HAVING절의 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)]
HAVING   [출력 그룹을 제한하는 조건식]
ORDER BY [정렬하려는 열 지정]
```

📎 HAVING절을 사용할 때 유의점
- HAVING절도 WHERE절 처럼 지정한 조건식이 참인 결과만 출력한다 는 점은 비슷하다.
-하지만 WHERE절은 출력 대상 행을 제한하고, HAVING절은 그룹화된 대상을 출력에서 제한하므로 쓰임새가 다르다.

- WHERE절을 사용하지 않고 HAVING절만 사용한 경우

```
SELECT DEPTNO, JOB, AVG(SAL)
  FROM EMP
GROUP BY DEPTNO, JOB
HAVING AVG(SAL) >= 2000
ORDER BY DEPTNO, JOB;
```

- WHERE절과 HAVING절을 모두 사용한 경우

```
SELECT DEPTNO, JOB, AVG(SAL)
  FROM EMP
 WHERE SAL <= 3000
GROUP BY DEPTNO, JOB
HAVING AVG(SAL) >= 2000
ORDER BY DEPTNO, JOB;
```

07-4. 그룹화와 관련된 여러 함수

실무에서 바로 사용할 확률은 높지않아서 간단히 훑어보거나 나중에 찾아봐도 좋다.

ROLLUP, CUBE, GROUPING SETS 함수

✅ ROLLUP

```
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)];
```

- ROLLUP함수는 명시한 열에 한하여 결과가 출력된다
- 그룹함수를 지정할 수 없다

📎 ROLLUP (A, B, C)

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

✅ CUBE

```
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)];
```

- CUBE함수는 지정한 모든 열에서 가능한 조합의 결과를 모두 출력한다.

📎 CUBE(A, B, C)
1. A 그룹별 B 그룹별 C 그룹에 해당하는 결과 출력
2. A 그룹별 B 그룹에 해당하는 결과 출력
3. B 그룹별 C 그룹에 해당하는 결과 출력
4. A 그룹별 C 그룹에 해당하는 결과 출력
5. A 그룹의 결과
6. B 그룹의 결과
7. C 그룹의 결과
8. 전체 데이터 결과

✅ GROUPING SETS 함수

```
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)];
```

- GROUPING SETS함수는 지정한 모든 열을 각각 대그룹으로 처리하여 출력한다.
- 그룹화를 위해 지정한 열이 계층적으로 분류되지 않고 각각 따로 그룹화한 후 연산을 수행했음.

그룹화 함수인 GROUPING, GROUPING_ID

✅ GROUPING

```
SELECT [조회할 열1 이름], [열2 이름], ... , [열N 이름]
  GROUPING [GROUP BY절에 ROLLUP 또는 CUBE에 명시한그룹화 할 열 이름]
FROM   [조회할 테이블 이름]
WHERE  [조회할 행을 선별하는 조건식]
GROUP BY ROLLUP 또는 CUBE [그룹화 할 열];
```

- 지정한 열이 그룹화 되었음을 의미하면 0이 나오고
- 지정한 열이 그룹화되지 않은 데이터를 의미하면 1이 출력된다.

✅ GROUPING_ID

```
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];
```

- 결과가 그룹화 비트 벡터 *grouping bit vector* 값으로 나타낸다
- 한번에 여러 개 열을 지정할 수 있으므로 지정한 열의 순서에 따라 0, 1 값이 하나씩 출력된다.
- 0과 1로 구성된 그룹화 비트 벡터 값을 2진수로 보고 10진수로 바꾼 값이 최종 결과로 출력.

LISTAGG 함수

✅ LISTAGG

```
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))
FROM   [조회할 테이블 이름]
WHERE  [조회할 행을 선별하는 조건식]
```

- 각 데이터를 구분하는 구분자를 지정하지 않을 경우 NULL이 기본값이 된다.

PIVOT, UNPIVOT 함수

✅ PIVOT

- 직책별/부서별 최고 급여를 2차원 표 형태로 출력하기

```
SELECT *
 FROM(SELECT DEPTNO, JOB, SAL
        FROM EMP)
PIVOT(MAX(SAL)
        FOR DEPTNO IN (10, 20, 30)
        )
ORDER BY JOB;
```

	JOB	10	20	30
1	ANALYST	(null)	3000	(null)
2	CLERK	1300	1100	950
3	MANAGER	2450	2975	2850
4	PRESIDENT	5000	(null)	(null)
5	SALESMAN	(null)	(null)	1600

```
SELECT *
  FROM(SELECT JOB, DEPTNO, SAL
         FROM EMP)
PIVOT(MAX(SAL)
      FOR JOB IN ('CLERK' AS CLERK
                  'SALESMAN' AS SALESMAN
                  'PRESIDENT' AS PRESIDENT
                  'MANAGER' AS MANAGER
                  'ANALYST' AS ANALYST
      )
ORDER BY DEPTNO;
```

```
SELECT DEPTNO,
       MAX(DECODE(JOB, 'CLERK', SAL)) AS "CLERK",
       MAX(DECODE(JOB, 'SALESMAN', SAL)) AS "SALESMAN",
       MAX(DECODE(JOB, 'PRESIDENT', SAL)) AS "PRESIDENT",
       MAX(DECODE(JOB, 'MANAGER', SAL)) AS "MANAGER",
       MAX(DECODE(JOB, 'ANALYST', SAL)) AS "ANALYST"
  FROM EMP
GROUP BY DEPTNO
ORDER BY DEPTNO;
```

위의 두 SQL문의 출력 값이 같다.