6. 그룹 함수

컴퓨터공학부

김은경

[학습 목표]

1. 그룹 함수가 무엇인지 정의할 수 있다.
2. SELECT 문을 구성하는 각 절(Clause)이 해석되는 순서를 설명할 수 있다.
3. AVG, SUM, COUNT, MAX, MIN과 같은 그룹 함수를 적절히 사용할 수 있다.
4. GROUP BY 절을 사용하여 특정 행들의 데이터 집합을 그룹핑할 수 있다.
5. HAVING 절을 사용하여 그룹을 제한하는 조건을 기술할 수 있다.

1. 그룹 함수

그룹 함수란?

- 행의 집합에 함수를 적용하여, 행의 그룹 당 하나의 결과를 반환하는 함수
- SELECT 절과 HAVING 절, ORDER BY 절에서 사용할 수 있음

그룹 함수의 종류

함수 종류	기능
SUM	Null 값을 무시하고 합계를 계산하여 반환함
AVG	평균을 계산하여 반환함
MIN	Null 값을 무시한 최소값을 반환함
MAX	최대값을 반환함
COUNT	Null 값을 갖는 행을 제외한 행의 수를 계산하여 반환함 (단, 인수로 '*'를 지정하면, 중복되거나 Null인 행도 포함하여 계산함)

1. 그룹 함수

그룹 함수 인수의 특징

① 인수 앞에 DISTINCT 옵션을 지정한 경우, 인수 값 가운데 중복되지 않는 값만 고려하여 함수를 적용함
- DISTINCT(인수) 형태로 표현할 수 있음

② 인수 앞에 ALL 옵션을 지정한 경우, 중복 여부를 판단하지 않고 모든 인수 값에 함수를 적용함
- 디폴트로 ALL이 적용되므로, 생략해도 됨

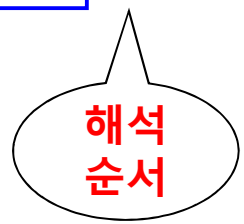
③ COUNT(*)를 제외하고, 모두 Null 값은 무시함

④ 인수에서 Null 값을 고려하려면, NVL 함수를 사용해야 함

1. 그룹 함수

그룹 함수를 포함하는 SELECT문의 각 절 해석 순서

```
SELECT      [열 이름1, 열 이름2, ...]  그룹함수(열 이름3), ...   ⑤
FROM        테이블 이름                                          ①
[WHERE      조건]                                                ②
[GROUP BY   열 이름1, 열 이름2, ...]                             ③
[HAVING     그룹 조건]                                           ④
[ORDER BY   열 이름4, 열 이름5, ...];                            ⑥
```



2. SUM 함수

SUM 함수란?

- 인수로 전달된 열 값의 합계를 계산하는 함수

SUM 함수의 특징

- 인수로 기술된 열 집합의 합계 값을 계산하여 반환함
- 인수로 NUMBER 형만 사용 가능함
- 인수가 Null 값을 갖는 경우에는 합계 계산에서 제외됨

2. SUM 함수

✓ SUM 함수 사용 사례

- 담당 업무가 세일즈인 사원들의 급여 합계 출력하기

```
SELECT    SUM(sal)  AS  "세일즈 담당자의 총급여"
FROM      emp
WHERE     job LIKE '세%';
```

세일즈 담당자의 총급여
11720

3. AVG 함수

AVG 함수란?

- 인수로 전달된 열 집합의 평균을 계산하는 함수

AVG 함수의 특징

- 인수로 NUMBER 형만 사용 가능함
- 인수가 널 값을 갖는 경우 무시됨
  - 즉, 널 값을 갖는 행은 평균을 계산하기 위해 합계를 나누는 행 수에서 제외됨
- 인수가 널 값을 갖는 행도 포함시키려면, NVL 함수를 사용해야 함

3. AVG 함수

✓ AVG 함수 사용 사례

- 업무상 보너스가 해당되는 사원들의 평균 보너스 출력하기

```
SELECT    AVG(bonus)
FROM      emp;
```

AVG(BONUS)
250

비교) 모든 사원의 평균 보너스 출력하기

```
SELECT    AVG(NVL(bonus, 0))
FROM      emp;
```

AVG(NVL(BONUS,0))
71.4285714

4. MIN과 MAX 함수

MIN과 MAX 함수란?

- 인수로 전달된 열 집합에서 최소 또는 최대 값을 검색해서 반환하는 함수

MIN과 MAX 함수의 특징

- 인수로 모든 데이터 형 사용 가능함
- MIN 함수의 경우, 널 값을 무시한 최소값을 반환함

4. MIN과 MAX 함수

✓ MIN과 MAX 함수 사용 사례

- 사원 가운데 이름이 가나다 순으로 가장 빠른 사원과 가장 늦은 사원의 이름 출력하기

```
SELECT    MIN(emp_name), MAX(emp_name)
FROM      emp;
```

MIN(EMP_NAME)	MAX(EMP_NAME)
김대정	진영진

5. COUNT 함수

COUNT 함수란?

- 인수로 전달된 열의 값이 널이 아닌 모든 행의 수를 반환하는 함수

COUNT 함수의 특징

- 열의 값이 중복되는 행도 포함해서 카운트함
- 인수가 '*'인 경우, 널인 행도 포함해서 모든 행의 수를 반환함

5. COUNT 함수

✓ COUNT 함수 사용 사례(1)

- 100번 부서에서 근무하는 모든 사원의 수 출력하기

```
SELECT    COUNT(*)  "100번 부서의 총 사원 수 "
FROM      emp
WHERE     dept_id = 100;
```

100번 부서의 총 사원 수
2

5. COUNT 함수

✓ COUNT 함수 사용 사례(2)

- 근무하는 사원이 있는 모든 부서의 수 출력하기

```
SELECT    COUNT(DISTINCT(dept_id))
FROM      emp;
```

COUNT(DISTINCT(DEPT_ID))
4

비교1) 사원 가운데 비서가 정해진 사원의 수 출력하기

```
SELECT    COUNT(dept_id)
FROM      emp;
```

COUNT(DEPT_ID)
13

비교2) 모든 사원의 수 출력하기

```
SELECT    COUNT(*)
FROM      emp;
```

COUNT(*)
14

6. GROUP BY 절과 그룹 함수

GROUP BY 절이란?

- SELECT 문에서 특정 행들의 집합으로 구성된 데이터 그룹을 생성하기 위한 절

```
SELECT      [열이름1, 열이름2, ...]  그룹함수(열 이름3)
FROM        테이블 이름
[WHERE      조건]
GROUP BY    열이름1, 열이름2, ...
[HAVING     그룹 제한 조건]
[ORDER BY   열이름4, 열이름5, ...];
```

GROUP BY 절 기술 시 유의사항

- **GROUP BY 절에 명시된 열 이름만 SELECT 절에 독립적으로 기술할 수 있음** (단, 그룹 함수의 인수로 사용되는 열 이름은 예외)
- GROUP BY 절에 명시한 열이 반드시 SELECT 절에 포함될 필요는 없음
- GROUP BY 절에 다중 열을 명시해서 하나 이상의 열로 그룹화 시킬 수 있음
  - 제일 먼저 기술한 열 이름에 대해서 그룹핑한 다음, 같은 그룹 안에서 다시 두 번째 기술한 열 이름을 기준으로 그룹핑하는 방식으로 그룹핑됨

6. GROUP BY 절과 그룹 함수

✓ GROUP BY 절 사용 사례

- 부서 번호와 각 부서별 평균 급여 출력하기

```
SELECT    dept_id,  AVG(sal)
FROM      emp
WHERE     dept_id  IS NOT NULL
GROUP  BY  dept_id;
```

DEPT_ID	AVG(SAL)
100	4350
400	4680
300	3900
200	5750

비교) WHERE절을 생략한 경우

```
SELECT    dept_id,  AVG(sal)
FROM      emp
GROUP  BY  dept_id;
```

DEPT_ID	AVG(SAL)
100	4350
400	4680
	4500
300	3900
200	5750

6. GROUP BY 절과 그룹 함수

✓ ORDER BY 절에서 열 별칭 사용 사례

- ORDER BY절이 SELECT절 다음에 해석되므로 열 별칭 사용 가능함

```
SELECT    dept_id 부서,  AVG(sal) 평균급여
FROM      emp
WHERE     dept_id  IS NOT NULL
GROUP  BY  dept_id
ORDER  BY  평균급여;
```

부서	평균급여
300	3900
100	4350
400	4680
200	5750

6. GROUP BY 절과 그룹 함수

✓ **ORDER BY 절에 하나 이상의 열 이름을 지정한 사례**

- 부서 번호와 각 부서별 담당 업무 및 부서 단위의 업무별 총 급여 출력하기

```
SELECT    dept_id,  job,  SUM(sal)
FROM      emp
WHERE     dept_id  IS NOT NULL
GROUP  BY  dept_id,  job;
```

DEPT_ID	JOB	SUM(SAL)
400	부장	7000
100	부장	6200
300	부장	5850
200	대표이사	9000
400	세일즈	11720
300	경리	3200
200	연구직	5000
100	사무직	2500
200	사무직	2500
200	부장	6500
300	사무직	2650

11 rows selected.

6. GROUP BY 절과 그룹 함수

✓ **출력을 위해 ORDER BY 절에 열 이름을 지정한 사례**

- 각 부서의 이름과 위치, 부서별 사원의 수를 출력하되, 사원이 없는 부서는 출력에서 제외시키기

```
SELECT   dept_name  부서명,  dept_loc  부서위치,  COUNT(*) "사원 수"
FROM     emp  e,  dept  d
WHERE    e.dept_id  =  d.dept_id
GROUP  BY  dept_name,  dept_loc;
```

부서명	부서위치	사원 수
총무부	서울	3
관리부	서울	2
연구부	대전	4
영업부	인천	4

7. 그룹 함수의 오류

오류 원인

- WHERE 절에 그룹을 제한하는 조건을 기술한 경우, 각 절이 해석되는 순서에 따라 오류가 발생함

← GROUP BY 절보다 WHERE 절이 먼저 해석되므로, WHERE 절에서 어떻게 그룹핑 되었는지 알 수 없는 상태에서 그룹을 제한하는 조건을 바르게 해석할 수 없음

예)
```
SELECT  dept_id,  AVG(sal)   ④
FROM    emp                  ①
WHERE   AVG(sal) > 5000      ②
GROUP   BY    dept_id;       ③
```

오류 발생!

```
WHERE  AVG(sal) > 5000
       *

ERROR at line 3:
ORA-00934: group function is not allowed here
```

7. 그룹 함수의 오류

방지책

- 그룹을 제한하는 조건을 WHERE 절 대신 **HAVING 절**에 기술함

예)
```
SELECT   dept_id,   AVG(sal)      ④
FROM     emp                      ①
GROUP   BY    dept_id            ②
HAVING   AVG(sal) > 5000;        ③
```

; HAVING절을 해석할 때 dept_id에 대해 그룹핑되었는지 알고 있는 상태이므로, AVG(sal)이 어떤 그룹에 대한 평균 값인지 알 수 있고, 따라서 오류가 발생하지 않음

DEPT_ID	AVG(SAL)
200	5750

8. HAVING 절과 그룹 함수

HAVING 절이란?

- 그룹을 제한하기 위한 조건을 명시하기 위한 절
  → HAVING 절의 조건을 만족하는 그룹만 표시하게 됨

```
SELECT      [열이름1, 열이름2, ...]  그룹함수(열 이름3)
FROM        테이블 이름
[WHERE      조건]
GROUP BY    열이름1, 열이름2, ...
HAVING      그룹 제한 조건
[ORDER BY   열이름4, 열이름5, ...];
```

HAVING 절 기술 시 유의사항

① 그룹을 제한하는 조건은 WHERE 절 대신 HAVING 절에 기술함

② HAVING 절을 GROUP BY 절 뒤에 작성하는 것이 좋음

8. HAVING 절과 그룹 함수

✓ HAVING 절 사용 사례

- 부서의 급여 총액이 5000 이상인 부서의 부서 번호와 급여 총액 출력하기

```
SELECT    dept_id  부서,  SUM(sal)  "급여 총액"
FROM      emp
GROUP     BY dept_id
HAVING    SUM(sal) >= 15000;
```

부서	급여 총액
400	18720
200	23000

8. HAVING 절과 그룹 함수

✓ **HAVING 절과 SELECT 절에서 다른 그룹 함수를 사용한 사례**

- 소속 사원의 최대 급여가 3700 이상인 부서의 부서번호와 평균 급여 출력하기

```
SELECT    dept_id  부서,  AVG(sal)  "평균 급여"
FROM      emp
GROUP     BY  dept_id
HAVING    MAX(sal)  >= 7000;
```

부서	평균 급여
400	4680
200	5750

8. HAVING 절과 그룹 함수

✓ **SELECT 문의 6개 절을 모두 기술한 사례**

- 연구직을 제외하고 각 업무별 사원들의 전체 급여가 10,000 이상인 각 업무에 대해서, 업무명과 각 업무별 급여의 합계를 합계의 오름차순으로 출력하기

```
SELECT    job  업무,  SUM(sal)  "급여 총액"
FROM      emp
WHERE     job  NOT LIKE  '연구직'
GROUP     BY   job
HAVING        SUM(sal) >= 10000
ORDER  BY  SUM(sal);
```

업무	급여 총액
세일즈	11720
부장	25550

9. 그룹 함수의 중첩

그룹 함수 중첩의 특징

① 그룹 함수들끼리 중첩해서 사용할 수 있음

② 단일 행 함수와도 중첩 가능함

③ 가장 내부 함수가 가장 먼저 계산됨

9. 그룹 함수의 중첩

✓ 그룹 함수 중첩 사례

- 평균 급여가 최대인 부서의 평균 급여가 얼마인지 검색하기

```
SELECT    MAX(AVG(sal))   "평균 급여의 최대치"
FROM      emp
GROUP     BY    dept_id;
```

평균 급여의 최대치
5750

[비교] dept_id를 SELECT 절에 명시한 경우

```
SELECT    dept_id, MAX(AVG(sal))   "부서별 평균 급여의 최대치"
FROM      emp
GROUP     BY    dept_id;
```

```
SELECT      dept_id,
            *

1행에 오류:
ORA-00937: 단일 그룹의 그룹 함수가 아닙니다
```

그룹별 출력이 아닌 경우, GROUP BY 절에 명시한 열 이름을 SELECT 절에 명시할 수 없음

9. 그룹 함수의 중첩

✓ 단일 행 함수와 그룹 함수의 중첩 사용 사례

- 부서번호와 각 부서별 사원의 수, 부서별 평균 급여를 출력하되, 평균 급여는 소수점 이하 첫째 자리에서 반올림해서 출력하기

```
SELECT    dept_id  부서,  COUNT(*)  "사원 수",
          ROUND(AVG(sal), 1)  "급여 평균"
FROM      emp
WHERE     dept_id  IS NOT NULL
GROUP     BY  dept_id;
```

부서	사원 수	급여 평균
100	2	4350
400	4	4680
300	3	3900
200	4	5750

Q & A