
ORACLE
ACADEMY

Programación de Bases de Datos con SQL

9-1
Uso de las Cláusulas Group By y Having

ORACLE ACADEMY

Objetivos

En esta lección se abordan los siguientes objetivos:

- Crear y ejecutar una consulta SQL utilizando GROUP BY
- Crear y ejecutar una consulta SQL utilizando GROUP BY … HAVING
- Crear y ejecutar GROUP BY en más de una columna
- Anidar funciones de grupo

Objetivo

- ¿Si deseara saber la altura media de todos los alumnos?
- Podría escribir una consulta similar a la siguiente:

```
SELECT AVG(height) FROM students;
```

La tabla students no está disponible en APEX.

Objetivo

- Y ¿qué ocurriría si deseara saber la altura media de los alumnos en función de su año en el centro educativo?
- Con lo que usted conoce ahora mismo, tendría que escribir una serie de diferentes sentencias SQL para realizar este proceso:

```
SELECT AVG(height) FROM students WHERE year_in_school = 10;
SELECT AVG(height) FROM students WHERE year_in_school = 11;
SELECT AVG(height) FROM students WHERE year_in_school = 12;
```

- Y así sucesivamente.
- Para simplificar problemas como este con solo una sentencia, utiliza las cláusulas GROUP BY y HAVING.

Uso de GROUP BY

- Utilice la cláusula GROUP BY para dividir las filas de una tabla en grupos más pequeños.
- A continuación puede utilizar las funciones de grupo para devolver información de resumen de cada grupo.

```
SELECT department_id, AVG(salary)
FROM employees
GROUP BY department_id
ORDER BY department_id;
```

DEPARTMENT_ID	AVG(SALARY)
10	4400
20	9500
50	3500
60	6400
80	10033,333333333333333
90	19333,333333333333333
110	10150
-	7000

Uso de GROUP BY

- En la sentencia SELECT mostrada, las filas se están agrupando por department_id.
- A continuación, se aplica la función AVG a cada grupo.

```
SELECT department_id, AVG(salary)
FROM employees
GROUP BY department_id
ORDER BY department_id;
```

DEPARTMENT_ID	AVG(SALARY)
10	4400
20	9500
50	3500
60	6400
80	10033,333333333333333
90	19333,333333333333333
110	10150
-	7000

Ejemplo de GROUP BY

- ¿Qué sucedería si deseara saber el salario máximo de los empleados de cada departamento?
- Utilizamos una cláusula GROUP BY que indica qué columna utilizar para agrupar las filas.

```
SELECT MAX(salary)
FROM employees
GROUP BY department_id;
```

DEPT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4200
50	5800
50	3500
50	3100
50	2600
50	2500
…	…

MAX(SALARY)
7000
24000
13000
…

Ejemplo de GROUP BY

- Pero, ¿cómo saber qué salario máximo corresponde a cada departamento?

DEPT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4200
50	5800
50	3500
50	3100
50	2600
50	2500
...	...

MAX(SALARY)
7000
24000
13000
...

GROUP BY en SELECT

- Normalmente deseamos incluir la columna GROUP BY en la lista SELECT.

```
SELECT department_id, MAX(salary)
FROM employees
GROUP BY department_id;
```

DEPT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4200
...	...

DEPT_ID	MAX(SALARY)
-	7000
90	24000
20	13000
...	...

Cláusula GROUP BY

- Las funciones de grupo requieren que cualquier columna mostrada en la cláusula SELECT que no forme parte de una función de grupo deba aparecer en una cláusula GROUP BY.
- ¿Qué es incorrecto en este ejemplo?

```
SELECT job_id, last_name, AVG(salary)
FROM employees
GROUP BY job_id;
```

ORA-00979: not a GROUP BY expression

Job_id es aceptable en la lista SELECT, pero last_name no, ya que cada grupo único de job_id solo produce una fila de salida (ya que es la columna GROUP BY). Sin embargo, puede haber muchos empleados distintos que tengan ese mismo job_id, por ejemplo, hay tres empleados con un valor de job_id SA_REP.

COUNT

- En este ejemplo se muestra cuántos países hay en cada región.
- Recuerde que las funciones de grupo ignoran los valores nulos, por lo que si algún país no tiene un nombre de país, no se incluirá en el valor COUNT.

```
SELECT COUNT(country_name), region_id
FROM wf_countries
GROUP BY region_id
ORDER BY region_id;
```

COUNT(COUNTRY_NAME)	REGION_ID
15	5
28	9
21	11
8	13
7	14
8	15
5	17
17	18

COUNT

- Por supuesto, esto es poco probable, pero, al crear sentencias SQL, tenemos que pensar en todas las posibilidades.
- Sería mejor escribir la consulta utilizando COUNT(*):

```
SELECT COUNT(*), region_id
FROM wf_countries
GROUP BY region_id
ORDER BY region_id;
```

- Esto contaría todas las filas de cada grupo de regiones, sin tener que comprobar las columnas que contuviesen valores NULL.

Cláusula WHERE

- También podemos utilizar una cláusula WHERE para excluir las filas antes de que las filas restantes formen grupos.

```
SELECT department_id, MAX(salary)
FROM employees
WHERE last_name != 'King'
GROUP BY department_id;
```

LAST_NAME	DEPT_ID	SALARY
King	90	24000
Kochhar	90	17000
De Haan	90	17000
Hunold	60	9000
Ernst	60	6000
Lorentz	60	4200
…	…	…

DEPT_ID	MAX(SALARY)
-	7000
90	17000
20	13000
…	…

Como empleado, King está excluido por la cláusula WHERE, el valor MAX(salary) para el departamento 90 se devuelve como 17000.

Más Ejemplos de GROUP BY

- Muestre la población media de todos los países de cada región.
- Redondee la media a un número entero.

```
SELECT region_id, ROUND(AVG(population)) AS population
FROM wf_countries
GROUP BY region_id
ORDER BY region_id;
```

- Cuente el número de idiomas hablados para todos los países.

```
SELECT country_id, COUNT(language_id) AS "Number of languages"
FROM wf_spoken_languages
GROUP BY country_id;
```

Directrices de GROUP BY

- Las directrices importantes que no se deben olvidar al utilizar una cláusula GROUP BY son:
  - Si incluye una función de grupo (AVG, SUM, COUNT, MAX, MIN, STDDEV, VARIANCE) en una cláusula SELECT junto con cualquier otra columna individual, cada columna individual también debe aparecer en la cláusula GROUP BY.
  - No puede utilizar un alias de columna en la cláusula GROUP BY.
  - La cláusula WHERE excluye las filas antes de que se dividan en grupos.

Grupos dentro de GRUPOS

- A veces tiene que dividir los grupos en grupos más pequeños.
- Por ejemplo, puede que desee agrupar todos los empleados por departamento y, a continuación, dentro de cada departamento, agruparlos por trabajo.

```
SELECT department_id, job_id, count(*)
FROM employees
WHERE department_id > 40
GROUP BY department_id, job_id;
```

DEPT_ID	JOB_ID	COUNT(*)
110	AC_ACCOUNT	1
50	ST_CLERK	4
80	SA_REP	2
90	AD_VP	2
50	ST_MAN	1
…	…	…

Grupos dentro de GRUPOS

- En este ejemplo se muestra el número de empleados que están realizando cada trabajo dentro de cada departamento.

```
SELECT department_id, job_id, count(*)
FROM employees
WHERE department_id > 40
GROUP BY department_id, job_id;
```

DEPT_ID	JOB_ID	COUNT(*)
110	AC_ACCOUNT	1
50	ST_CLERK	4
80	SA_REP	2
90	AD_VP	2
50	ST_MAN	1
…	…	…

Anidamiento de Funciones de Grupo

- Las funciones de grupo se pueden anidar en una profundidad de dos cuando se utilice GROUP BY.

```
SELECT max(avg(salary))
FROM employees
GROUP by department_id;
```

- ¿Cuántos valores se devuelven con esta consulta?
- La respuesta es uno: la consulta buscará el salario medio de cada departamento y, a continuación, en esa lista, seleccionará el único valor mayor.

HAVING

- Suponga que desea buscar el salario máximo en cada departamento, pero solo para aquellos departamentos que tengan más de un empleado.
- ¿Qué es incorrecto en este ejemplo?

```
SELECT department_id, MAX(salary)
FROM employees
WHERE COUNT(*) > 1
GROUP BY department_id;
```

ORA-00934: group function is not allowed here

Se puede utilizar una cláusula WHERE solo para incluir/excluir algunas filas, no grupos de filas. Por lo tanto, no podemos utilizar las funciones de grupo en una cláusula WHERE.

HAVING

- De la misma forma que ha utilizado la cláusula WHERE para restringir las filas que seleccionó, puede utilizar la cláusula HAVING para restringir los grupos.
- En una consulta con una cláusula GROUP BY y HAVING, primero se agrupan las filas, se aplican las funciones de grupo y, a continuación, solo se muestran los grupos que coincidan con la cláusula HAVING.

HAVING

- Se utiliza la cláusula WHERE para restringir las filas; se utiliza la cláusula HAVING para restringir los grupos devueltos por una cláusula GROUP BY.

```
SELECT department_id,MAX(salary)
FROM employees
GROUP BY department_id
HAVING COUNT(*)>1
ORDER BY department_id;
```

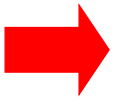
DEPARTMENT_ID	MAX(SALARY)
20	13000
50	5800
60	9000
80	11000
90	24000
110	12000

La consulta encuentra primero el salario MAX de cada departamento en la tabla employees.
A continuación, la cláusula HAVING restringe los grupos devueltos a los departamentos que tengan más de 1 empleado.

HAVING

- Esta consulta busca la población media de los países de cada región.
- A continuación, solo devuelve los grupos de regiones con una población inferior superior a trescientos mil.

```
SELECT region_id,
  ROUND(AVG(population))
FROM wf_countries
GROUP BY region_id
HAVING MIN(population)>300000
ORDER BY region_id;
```



REGION_ID	ROUND(AVG(POPULATION))
14	27037687
17	18729285
30	193332379
34	173268273
143	12023602
145	8522790
151	28343051

Las cláusulas HAVING y GROUP BY pueden utilizar diferentes columnas. En el ejemplo de la diapositiva se realizan GROUP BY en los valores region_id, pero la cláusula HAVING restringe los grupos según la población.

HAVING

- Aunque la cláusula HAVING puede preceder a la cláusula GROUP BY en una sentencia SELECT, se recomienda que coloque cada cláusula en el orden que se muestra.
- La cláusula ORDER BY (si se utiliza) es siempre la última.

```
SELECT column, group_function
FROM table
WHERE
GROUP BY
HAVING
ORDER BY
```

Terminología

Entre los términos clave utilizados en esta lección se incluyen:

- GROUP BY
- HAVING

Resumen

En esta lección, debe haber aprendido lo siguiente:

- Crear y ejecutar una consulta SQL utilizando GROUP BY
- Crear y ejecutar una consulta SQL utilizando GROUP BY … HAVING
- Crear y ejecutar GROUP BY en más de una columna
- Anidar funciones de grupo


ORACLE
ACADEMY