



Autor: **Nicola Strappazon**
Web: <http://nicola.strappazon.me>
Revisión: 17/11/11

Lección 2

Restringiendo y Ordenando los Datos retornados por el comando SELECT

Este material se encuentra basado en el curso de Fundamentos a SQL de ORACLE, el cual es adaptado para el producto PostgreSQL, todos los ejemplos, códigos fuentes y la Base de Datos HR es propiedad de ORACLE.

Objetivos

Al completar esta lección usted podrá entender los siguientes puntos:

- Limitar el numero de registros que son retornados por la consulta.
- Ordenar los registros por columnas.

Sintaxis Básica

```
SELECT [ DISTINCT ] * | expresion [ AS alias ] [, ...]  
FROM   from item [, ...]  
WHERE conditions;
```

- Limita el numero de registros mediante el uso de la cláusula **WHERE**.
- Usted agrega las condiciones necesarias para limitar o filtrar los registros que serán retornados por la consulta.

Usando la Cláusula WHERE

```
SELECT employee_id, last_name, job_id, department_id
FROM employees
WHERE department_id = 90;
```

employee_id	last_name	job_id	department_id
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90
100	King	AD_PRES	90

(3 rows)

Texto y Fechas

- Todos los valores de tipo texto y fecha deben ser encerrados dentro de comillas simples.
- El texto es de tipo case-sensitive, y la fecha es sensible al formato.
- El formado de entrada por defecto es YYYY-MM-DD.

```
SELECT employee_id, last_name, job_id, department_id  
FROM employees  
WHERE last_name = 'Whalen';
```

Condiciones de Comparación

Operador	Significado
=	Igual
>	Mayor que
<	Menor que
>=	Mayor e igual que
<=	Menor e igual que
BETWEEN	Rango, valor mayor e igual entre menor e igual.
IN	Coincidencia con una lista de valores
LIKE	Comparación de texto case-sensitive
ILIKE	Comparación de texto
IS NULL	Comparación de un valor tipo NULL

Usando la Condición de Comparación

```
SELECT last_name, salary
FROM employees
WHERE salary <= 3000;
```

last_name	salary
Baida	2900
Tobias	2800
Himuro	2600
Colmenares	2500
Mikkilineni	2700
Landry	2400
Markle	2200
Atkinson	2800
Marlow	2500

(26 rows)

Usando la Condición BETWEEN

```
SELECT last_name, salary
FROM employees
WHERE salary BETWEEN 2500 AND 3500;
```

last_name	salary
Khoo	3100
Baida	2900
Tobias	2800
Himuro	2600
Colmenares	2500
Nayer	3200
Mikkilineni	2700
Bissot	3300
Atkinson	2800
Marlow	2500

(33 rows)

Usando la Condición IN

```
SELECT last_name, salary, manager_id
FROM employees
WHERE manager_id IN (100, 101, 201);
```

last_name	salary	manager_id
Russell	14000	100
Partners	13500	100
Errazuriz	12000	100
Cambrault	11000	100
Zlotkey	10500	100
Kochhar	17000	100
De Haan	17000	100
Greenberg	12000	101
Raphaely	11000	100
Weiss	8000	100
(20 rows)		

Patrones de Búsqueda

Operador	Significado
%	Cualquier coincidencia en adelante
—	Una sola coincidencia de carácter

Usando la Condición LIKE

```
SELECT first_name  
FROM employees  
WHERE first_name LIKE 'S%';
```

```
first_name  
-----  
Sarath  
Sundar  
Sundita  
Shelli  
Sigal  
Shanta  
Steven  
Stephen  
Sarah  
Samuel  
(13 rows)
```

Usando la Condición LIKE

```
SELECT first_name  
FROM employees  
WHERE first_name LIKE '_o%';
```

```
first_name  
-----  
John  
Louise  
Jonathon  
John  
Jose Manuel  
Mozhe  
John  
Joshua  
Donald  
Douglas  
(10 rows)
```

Usando la Condición ILIKE

```
SELECT first_name  
FROM employees  
WHERE first_name LIKE 'susan';
```

```
first_name  
-----  
(0 rows)
```

```
SELECT first_name  
FROM employees  
WHERE first_name ILIKE 'susan';
```

```
first_name  
-----  
Susan  
(1 row)
```

Usando la Condición IS NULL

```
SELECT first_name, manager_id  
FROM employees  
WHERE manager_id IS NULL;
```

first_name	manager_id
Steven	

(1 row)

Condiciones Lógicas

Operador	Significado
AND	Retorna TRUE si ambas condiciones se cumplen.
OR	Retorno TRUE so si una de las condiciones se cumplen.
NOT	Retorna TRUE si ambas condiciones no se cumplen.

Usando el Operador AND

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary >= 10000
      AND job_id LIKE '%MAN%';
```

employee_id	last_name	job_id	salary
145	Russell	SA_MAN	14000
146	Partners	SA_MAN	13500
147	Errazuriz	SA_MAN	12000
148	Cambrault	SA_MAN	11000
149	Zlotkey	SA_MAN	10500
114	Raphaely	PU_MAN	11000
201	Hartstein	MK_MAN	13000

(7 rows)

Usando el Operador OR

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  salary >= 10000
      OR job_id LIKE '%MAN%';
```

employee_id	last_name	job_id	salary
145	Russell	SA_MAN	14000
146	Partners	SA_MAN	13500
147	Errazuriz	SA_MAN	12000
148	Cambrault	SA_MAN	11000
149	Zlotkey	SA_MAN	10500
150	Tucker	SA_REP	10000
156	King	SA_REP	10000
162	Vishney	SA_REP	10500
168	Ozer	SA_REP	11500
169	Bloom	SA_REP	10000

(24 rows)

Usando el Operador NOT

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  job_id NOT IN ('IT_PROG', 'ST_CLERK', 'SA_REP');
```

employee_id	last_name	job_id	salary
145	Russell	SA_MAN	14000
146	Partners	SA_MAN	13500
147	Errazuriz	SA_MAN	12000
148	Cambrault	SA_MAN	11000
149	Zlotkey	SA_MAN	10500
101	Kochhar	AD_VP	17000
102	De Haan	AD_VP	17000
108	Greenberg	FI_MGR	12000
109	Faviet	FI_ACCOUNT	9000
110	Chen	FI_ACCOUNT	8200

(52 rows)

Reglas de Precedencia

Orden	Significado
1	Operaciones Aritméticas
2	Concatenación
3	Comparación
4	IS [NOT] NULL, LIKE, NOT [IN]
5	[NOT] BETWEEN
6	No es igual
7	Operador lógico NOT
8	Operador lógico AND
9	Operador lógico OR

Reglas de Precedencia

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  job_id = 'SA_REP'
       OR job_id = 'AD_PRES'
       AND salary > 15000;
```

employee_id	last_name	job_id	salary
179	Johnson	SA_REP	6200
100	King	AD_PRES	24000
167	Banda	SA_REP	6200
168	Ozer	SA_REP	11500
169	Bloom	SA_REP	10000
170	Fox	SA_REP	9600
171	Smith	SA_REP	7400
172	Bates	SA_REP	7300

(31 rows)

Reglas de Precedencia

```
SELECT employee_id, last_name, job_id, salary
FROM   employees
WHERE  (job_id = 'SA_REP'
        OR  job_id = 'AD_PRES')
        AND salary > 15000;
```

employee_id	last_name	job_id	salary
100	King	AD_PRES	24000

(1 row)

Usando la Cláusula **ORDER BY**

- Ordena los registros que son recibidos mediante la cláusula **ORDER BY**:
 - **ASC**: Ordena de forma Ascendente (Por defecto)
 - **DESC**: Ordena de forma Decendente
- La cláusula **ORDER BY** se agrega al final de la sentencia **SELECT**.

Ordenando de Forma Ascendente

```
SELECT first_name, job_id, department_id, hire_date
FROM employees
ORDER BY hire_date;
```

first_name	job_id	department_id	hire_date
Steven	AD_PRES	90	1987-06-17
Jennifer	AD_ASST	10	1987-09-17
Neena	AD_VP	90	1989-09-21
Alexander	IT_PROG	60	1990-01-03
Bruce	IT_PROG	60	1991-05-21
Lex	AD_VP	90	1993-01-13
Susan	HR_REP	40	1994-06-07
Hermann	PR_REP	70	1994-06-07
Shelley	AC_MGR	110	1994-06-07
William	AC_ACCOUNT	110	1994-06-07

(107 row)

Ordenando de Forma Descendente

```
SELECT first_name, job_id, department_id, hire_date
FROM employees
ORDER BY hire_date DESC;
```

first_name	job_id	department_id	hire_date
Sundita	SA_REP	80	2000-04-21
Amit	SA_REP	80	2000-04-21
Sundar	SA_REP	80	2000-03-24
Steven	ST_CLERK	50	2000-03-08
David	SA_REP	80	2000-02-23
Hazel	ST_CLERK	50	2000-02-06
Girard	SH_CLERK	50	2000-02-03
Eleni	SA_MAN	80	2000-01-29
Mattea	SA_REP	80	2000-01-24
Douglas	SH_CLERK	50	2000-01-13
Charles	SA_REP	80	2000-01-04
(107 row)			

Ordenando por Alias

```
SELECT first_name, department_id, salary * 12 AS annsal  
FROM employees  
ORDER BY annsal;
```

first_name	department_id	annsal
TJ	50	25200
Hazel	50	26400
Steven	50	26400
James	50	28800
Ki	50	28800
Peter	50	30000
Karen	30	30000
Randall	50	30000
James	50	30000
Martha	50	30000
Joshua	50	30000

(107 row)

Ordenando por Varias Columnas

```
SELECT first_name, department_id, salary * 12 AS annsal  
FROM employees  
ORDER BY first_name, annsal;
```

first_name	department_id	annsal
Adam	50	98400
Alana	50	37200
Alberto	80	144000
Alexander	30	37200
Alexander	60	108000
Alexis	50	49200
Allan	80	108000
Alyssa	80	105600
Amit	80	74400
Anthony	50	36000
Britney	50	46800

(107 row)

Resumen

En esta lección, usted debió entender como:

- Usar la cláusula **WHERE** de SQL.
 - Comparar un valor a una columna.
 - Comparar un rango de valores a un columna.
 - Comparar Texto y Fecha a una columna.
 - Buscar coincidencia de caracteres en una columna.
 - Ordenar de forma Ascendente y Descendente el resultado.