
ORACLE
ACADEMY

Programación de Bases de Datos con SQL

3-2
Ordenación de Filas

ORACLE ACADEMY

Objetivos

En esta lección se abordan los siguientes objetivos:

- Crear una consulta para ordenar un juego de resultados en orden ascendente o descendente
- Establecer el orden en el que se evalúan y calculan las expresiones en función de las reglas de prioridad
- Crear una consulta para ordenar un juego de resultados con un alias de columna
- Crear una consulta para ordenar un juego de resultados para una o varias columnas

Objetivo

- Por naturaleza, la mayoría de nosotros necesitamos orden en nuestras vidas.
- Imagine que cada vez que cenara, tuviera que buscar en cada cajón o armario de la cocina para buscar un cuchillo y un tenedor.
- Ordenar, agrupar y clasificar facilita la búsqueda de las cosas.
- Los biólogos agrupan a los animales en filos, los astrónomos ordenan el brillo de las estrellas por su magnitud y los programadores de Java organizan el código en clases.

Objetivo

- La vida diaria está ordenada en muchas situaciones:
  - Los libros en la biblioteca
  - Las estanterías del supermercado
  - Los documentos almacenados en archivadores
- Poder ordenar resultados es una función útil en SQL y permite a los programadores mostrar información de muchas formas diferentes.
- Para el diseño de bases de datos, las funciones de negocio se ordenan por entidades y atributos; en la información de la base de datos, SQL utiliza la cláusula ORDER BY.

Cláusula ORDER BY

- La información ordenada en orden ascendente nos es conocida.
- Es lo que hace que consultar un número en una agenda, buscar una palabra en el diccionario, o localizar una casa por su dirección sea relativamente fácil.
- SQL utiliza la cláusula ORDER BY para ordenar datos.
- La cláusula ORDER BY puede especificar varias formas en las que ordenar las filas devueltas en una consulta.

Cláusula ORDER BY

- El orden por defecto es el orden ascendente.
- Se muestran los valores numéricos del más bajo al más alto.
- Los valores de fecha se muestran con el valor más reciente en primer lugar.
- Los valores de caracteres se muestran en orden alfabético.
- Los valores nulos se muestran al final para las secuencias ascendentes y al principio para las secuencias descendentes.
- NULLS FIRST especifica que los valores nulos se deben devolver antes los valores no nulos.
- NULLS LAST especifica que los valores nulos se deben devolver después los valores no nulos.

Si utiliza NULLS FIRST o NULLS LAST, deben estar al final de la cláusula ORDER BY, después de los nombres de columna. Por ejemplo:

```
SELECT last_name, hire_date, department_id
FROM employees
ORDER BY department_id NULLS LAST;
```

Cláusula ORDER BY

- El siguiente ejemplo de empleados utiliza la cláusula ORDER BY para ordenar hire_date en orden ascendente (valor por defecto).
- Nota: La cláusula ORDER BY debe ser la última cláusula de la sentencia SQL.

```
SELECT last_name, hire_date
FROM employees
ORDER BY hire_date;
```

LAST_NAME	HIRE_DATE
King	17-Jun-1987
Whalen	17-Sep-1987
Kochhar	21-Sep-1989
Hunold	03-Jan-1990
Ernst	21-May-1991
De Haan	13-Jan-1993
Gietz	07-Jun-1994
Higgins	07-Jun-1994
Rajs	17-Oct-1995
Hartstein	17-Feb-1996

Ordenación en Orden Descendente

- Puede invertir el orden por defecto en la cláusula ORDER BY para que aparezcan en orden descendente especificando la palabra clave DESC después del nombre de columna en la cláusula ORDER BY.

```
SELECT last_name, hire_date
FROM employees
ORDER BY hire_date DESC;
```

LAST_NAME	HIRE_DATE
Zlotkey	29-Jan-2000
Mourgos	16-Nov-1999
Grant	24-May-1999
Lorentz	07-Feb-1999
Vargas	09-Jul-1998
Taylor	24-Mar-1998
Matos	15-Mar-1998
Fay	17-Aug-1997
Davies	29-Jan-1997
Abel	11-May-1996

Si utiliza ASC o DESC en la cláusula ORDER BY influirá la colocación de los valores nulos: los valores nulos se muestran los últimos en orden ascendente y los primeros en orden descendente.

Además, puede utilizar NULLS FIRST para especificar que los valores nulos se deben devolver antes de los valores no nulos. NULLS LAST especifica que los valores nulos se deben devolver después los valores no nulos.

Uso de Alias de Columna

- Puede ordenar los datos mediante un alias de columna.
- El alias utilizado en la sentencia SELECT es al que se hace referencia en la cláusula ORDER BY.

```
SELECT last_name, hire_date AS "Date
  Started"
FROM employees
ORDER BY "Date Started";
```

LAST_NAME	Date Started
King	17-Jun-1987
Whalen	17-Sep-1987
Kochhar	21-Sep-1989
Hunold	03-Jan-1990
Ernst	21-May-1991
De Haan	13-Jan-1993
Gietz	07-Jun-1994
Higgins	07-Jun-1994
Rajs	17-Oct-1995
Hartstein	17-Feb-1996

Ordenación con Otras Columnas

- También es posible utilizar la cláusula ORDER BY para ordenar la salida según una columna que no aparezca en la cláusula SELECT.
- En el siguiente ejemplo, los datos se ordenan según la columna last_name, incluso aunque esta columna no aparece en la sentencia SELECT.

```
SELECT employee_id, first_name
FROM employees
WHERE employee_id < 105
ORDER BY last_name;
```

EMPLOYEE_ID	FIRST_NAME
102	Lex
104	Bruce
103	Alexander
100	Steven
101	Neena

Es difícil verificar los resultados cuando se ordenan según una columna que no está seleccionada. En el mundo real, debería ejecutar la consulta seleccionando la columna last_name hasta que estuviera seguro de que obtenía los datos correctos. Después, podría eliminar dicha columna en la sentencia SELECT.

Orden de Ejecución

- El orden de ejecución de una sentencia SELECT es el siguiente:
  - Cláusula FROM: busca la tabla que contiene los datos
  - Cláusula WHERE: restringe las filas que se van a devolver
  - Cláusula SELECT: selecciona en el conjunto de datos reducido las columnas solicitadas
  - Cláusula ORDER BY: ordena el conjunto de resultados

Ordenación con Varias Columnas

- También se puede ordenar los resultados de la consulta por más de una columna.
- De hecho, no hay ningún límite en el número de columnas que se pueden agregar a la cláusula ORDER BY.

Ordenación con Varias Columnas

- A continuación se muestra un ejemplo de ordenación con varias columnas.
- Los empleados se ordenan primero por número de departamento (del más bajo al más alto), a continuación, se muestran los apellidos en orden alfabético (A-Z).

```
SELECT department_id, last_name
FROM employees
WHERE department_id <= 50
ORDER BY department_id, last_name;
```

DEPARTMENT_ID	LAST_NAME
10	Whalen
20	Fay
20	Hartstein
50	Davies
50	Matos
50	Mourgos
50	Rajs
50	Vargas

Ordenación con Varias Columnas

- Para crear una cláusula ORDER BY a fin de ordenar por varias columnas, especifique las columnas que se van a devolver y separe los nombres de columna mediante comas.
- Si desea invertir el orden de ordenación de una columna, agregue DESC después del nombre.

```
SELECT department_id, last_name
FROM employees
WHERE department_id <= 50
ORDER BY department_id DESC, last_name;
```

DEPARTMENT_ID	LAST_NAME
50	Davies
50	Matos
50	Mourgos
50	Rajs
50	Vargas
20	Fay
20	Hartstein
10	Whalen

Se ha invertido el orden para el valor de department_id (de la diapositiva anterior) con DESC, por lo que ahora se muestran del más alto al más bajo, el orden del apellido sigue siendo alfabético, de la A a la Z.

Terminología

Entre los términos clave utilizados en esta lección se incluyen:

- Cláusula ORDER BY
- ASCENDENTE
- DESCENDENTE
- Orden de Ejecución

Resumen

En esta lección, debe haber aprendido lo siguiente:

- Crear una consulta para ordenar un juego de resultados en orden ascendente o descendente
- Crear una consulta para ordenar un juego de resultados con un alias de columna
- Crear una consulta para ordenar un juego de resultados para una o varias columnas


ORACLE
ACADEMY