UNIDAD 2: RESTRICCION Y ORDENACION DE DATOS

INTRODUCCION

Al recuperar datos de la base de datos, es posible que se deban restringir las filas de datos que se muestran o bien especificar el orden en el que se muestran las filas. En esta unidad se estudian y ejemplifican las sentencias SQL que se deben utilizar para realizar estas acciones.

LIMITACION DE LAS FILAS MEDIANTE UNA SELECCION

En el siguiente ejemplo se solicita mostrar todos los empleados del departamento 90. Las filas con el valor 90 en la columna DEPARTMENT_ID son las únicas que se devuelven. Este método de restricción es la base de la cláusula WHERE en SQL.

LIMITACION DE LAS FILAS SELECCIONADAS

Se pueden restringir las filas devueltas por una consulta utilizando la cláusula WHERE. Esta cláusula contiene una condición que se debe cumplir y sigue directamente a la cláusula FROM. Si la condición es verdadera, se devuelve la fila que cumple la condición.

```
SELECT   *|{[DISTINCT] column|expression [alias],...}
FROM     table
[WHERE   condition(s)];
```

En la sintaxis:

WHERE	restringe la consulta a las filas que cumplen una condición
condition	está formado por nombres de columna, expresiones, constantes y un operador de comparación.

La cláusula WHERE puede comparar valores de columnas, valores literales, expresiones aritméticas o funciones. Consta de tres elementos:

- Nombre de columna
- Condición de comparación
- Nombre de columna, constante o lista de valores

USO DE LA CLAUSULA WHERE

En el siguiente ejemplo, la sentencia SELECT recupera el nombre, el identificador de cargo y el número de departamento de todos los empleados cuyos identificadores de cargo sean SA_REP.

Observar que se ha especificado el cargo SA_REP en mayúsculas para asegurarse de que coincide con la columna de identificador de cargo de la tabla EMPLOYEES. Las cadenas de caracteres son sensibles a mayúsculas/minúsculas

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

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90

CADENAS DE CARACTERES Y FECHAS

En la cláusula WHERE, las cadenas de caracteres y las fechas se deben escribir entre comillas simples (''), pero no las constantes numéricas.
Las búsquedas de caracteres son sensibles a mayúsculas/minúsculas. En el siguiente ejemplo, no se devuelve ninguna fila, pues la tabla EMPLOYEES almacena todos los apellidos con mayúsculas y minúsculas mezcladas:

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

Las bases de datos Oracle almacenan fechas en formato numérico interno, representando el siglo, el año, el mes, el día, las horas, los minutos y los segundos. El formato de fecha por defecto es DD-MON-RR.

CONDICIONES DE COMPARACION

Las condiciones de comparación se utilizan en condiciones que comparan una expresión con otro valor o expresión. Se usan en la cláusula WHERE con el siguiente formato:

Sintaxis:

WHERE expr operator value

Por Ejemplo:

WHERE hire_date='01-JAN-95'

WHERE salary>=6000

WHERE last_name='Smith'

Operador	Significado
=	Igual que
>	Mayor que
>=	Mayor o igual que
<	Menor que
<=	Menor o igual que
<>	No igual a

No se puede utilizar un alias en la cláusula WHERE.

Nota:
Los símbolos != y ^= también pueden representar la condición no igual a.

USO DE LAS CONDICIONES DE COMPARACION

En el siguiente ejemplo, la sentencia SELECT recupera el apellido y el salario de la tabla EMPLOYEES, si el salario del empleado es menor o igual que 3000. Observar que existe un valor explícito para la cláusula WHERE. El valor explícito 3000 se compara con el valor de salario de la columna SALARY de la tabla EMPLOYEES.

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

LAST_NAME	SALARY
Matos	2600
Vargas	2500

OTRAS CONDICIONES DE COMPARACION

Operador	Significado
BETWEEN ...AND...	Entre dos valores (ambos inclusive),
IN(set)	Coincide con cualquiera de una lista de valores
LIKE	Coincide con un patrón de caracteres
IS NULL	Es un valor nulo

USO DE LA CONDICION BETWEEN

Se pueden mostrar filas incluidas en un rango de valores utilizando la condición de rango BETWEEN. El rango que especifique contiene un límite inferior y uno superior.

La siguiente sentencia SELECT devuelve las filas de la tabla EMPLOYEES para cualquier empleado cuyo salario esté entre $2.500 y $3.500.

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

Límite inferior Límite superior

LAST_NAME	SALARY
Rajs	3500
Davies	3100
Matos	2600
Vargas	2500

Los valores especificados en la condición BETWEEN están incluidos. Debe especificar en primer lugar el límite inferior.

USO DE LA CONDICION IN

Para comprobar si hay valores en un juego especificado de valores, se utiliza la condición IN. Esta condición también se conoce como condición de pertenencia.

En el siguiente ejemplo muestra los números de empleado, apellidos, salarios y números de empleado del director para todos los empleados cuyos números de empleado del director sean 100, 101 o 201.

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

EMPLOYEE_ID	LAST_NAME	SALARY	MANAGER_ID
202	Fay	6000	201
200	Whalen	4400	101
205	Higgins	12000	101
101	Kochhar	17000	100
102	De Haan	17000	100
124	Mourgos	5800	100
149	Zlotkey	10500	100
201	Hartstein	13000	100

La condición IN se puede utilizar con cualquier tipo de dato. El siguiente ejemplo devuelve una fila de la tabla EMPLOYEES para cualquier empleado cuyo apellido esté incluido en la lista de nombres en la cláusula WHERE:

```
SELECT employee_id, manager_id, department_id
FROM   employees
WHERE  last_name IN ('Hartstein', 'Vargas');
```

Si se utilizan caracteres o fechas en la lista, deben escribirse entre comillas simples ('').

USO DE LA CONDICION LIKE

No siempre conocerá el valor exacto para buscar. Puede seleccionar filas que coincidan con un patrón de caracteres utilizando la condición LIKE. La operación de coincidencia del patrón de caracteres se denomina búsqueda con comodines. Se pueden utilizar dos símbolos para construir la cadena de búsqueda

Símbolo	Descripción
%	Representa una secuencia de cero a más caracteres
_	Representa un único caracter

La siguiente sentencia SELECT devuelve el nombre del empleado de la tabla EMPLOYEES para cualquier empleado cuyo nombre empiece por S. Observar que la S está en mayúsculas. No se devolverán nombres que empiecen con s.

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

La condición LIKE se puede utilizar como método abreviado de algunas comparaciones BETWEEN. El siguiente ejemplo muestra los apellidos y las fechas de contratación de todos los empleados contratados entre enero y diciembre de 1995:

```
SELECT last_name, hire_date
FROM   employees
WHERE  hire_date LIKE '%95';
```

Los símbolos % y _ se pueden utilizar en cualquier combinación con caracteres literales. El siguiente ejemplo muestra los nombres de todos los empleados cuyos apellidos tienen una o (o minúscula) como segundo carácter.

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

LAST_NAME
Kochhar
Lorentz
Mourgos

Si necesita una coincidencia exacta de los caracteres % y/o _ reales, se debe utilizar la opción ESCAPE. Esta opción especifica cuál es el carácter de escape. Si desea buscar cadenas que contengan 'SA_', se utiliza la siguiente sentencia SQL:

```
SELECT employee_id, last_name, job_id
FROM   employees
WHERE  job_id LIKE '%SA\_%' ESCAPE '\';
```

La opción ESCAPE identifica la barra invertida (\) como el carácter de escape. En el patrón, el carácter de escape precede al carácter de subrayado (_). Esto hace que Oracle Server interprete al carácter de subrayado literalmente.

USO DE LAS CONDICONES NULL

Las condiciones NULL incluyen la condición IS NULL y la condición IS NOT NULL. La condición IS NULL comprueba si hay valores nulos. Un valor nulo significa que el valor no está disponible, no está asignado, es desconocido o no es aplicable. Por lo tanto, no puede se comprobar con = (igual), debido a que un valor nulo no puede ser igual o distinto de ningún valor. El siguiente ejemplo recupera los apellidos y los directores de todos los empleados que no tienen director:

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

LAST_NAME	MANAGER_ID
King	

El siguiente ejemplo muestra los apellidos, identificador de cargo y comisiones para todos los empleados que NO tienen derecho a recibir una comisión:

```
SELECT last_name, job_id, commission_pct
FROM   employees
WHERE  commission_pct IS NULL;
```

CONDICIONES LOGICAS

Las condiciones lógicas combinan el resultado de dos condiciones componentes para producir un resultado único basado en ellas o invierten el resultado de una única condición. Se devuelve una fila sólo si el resultado global de la condición es verdadero. En SQL están disponibles tres operadores lógicos:

Operador	Significado
AND	Devuelve TRUE si las *dos* condiciones componentes son verdaderas
OR	Devuelve TRUE si *alguna* de las condiciones componentes es verdadera
NOT	Devuelve TRUE si la siguiente condición es falsa

Todos los ejemplos mostrados hasta ahora han especificado solamente una condición en la cláusula WHERE. Puede utilizar varias condiciones en una cláusula WHERE utilizando los operadores AND y OR.

USO DEL OPERADOR AND

En ell siguiente ejemplo las dos condiciones deben ser verdaderas para que se seleccione un registro. Por lo tanto, solamente se seleccionarán los empleados cuyo cargo contenga la cadena MAN y ganen más de $10.000:

```
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
149	Zlotkey	SA_MAN	10500
201	Hartstein	MK_MAN	13000

Todas las búsquedas de caracteres son sensibles a mayúsculas/minúsculas. No se devolverán filas si MAN no está en mayúsculas. Las cadenas de caracteres se deben escribir entre comillas.

La siguiente tabla muestra los resultados de combinar dos expresiones con AND:

Operador AND	TRUE	FALSE	NULL
TRUE	TRUE	FALSE	NULL
FALSE	FALSE	FALSE	FALSE
NULL	NULL	FALSE	NULL

USO DEL OPERADOR OR

En el siguiente ejemplo cualquiera de las dos condiciones puede ser verdadera para que se seleccione un registro. Por lo tanto, se seleccionará cualquier empleado cuyo identificador de cargo contenga MAN o gane más de $10.000.

```
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
100	King	AD_PRES	24000
101	Kochhar	AD_VP	17000
102	De Haan	AD_VP	17000
124	Mourgos	ST_MAN	5800
149	Zlotkey	SA_MAN	10500
174	Abel	SA_REP	11000
201	Hartstein	MK_MAN	13000
205	Higgins	AC_MGR	12000

La siguiente tabla muestra los resultados de combinar dos expresiones con OR:

Operador OR	TRUE	FALSE	NULL
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	NULL
NULL	TRUE	NULL	NULL

USO DEL OPERADOR NOT

El siguiente ejemplo muestra el apellido y el identificador de cargo de todos los empleados cuyos identificadores de cargo no sean IT_PROG, ST_CLERK o SA_REP:

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

LAST_NAME	JOB_ID
King	AD_PRES
Kochhar	AD_VP
De Haan	AD_VP
Mourgos	ST_MAN
Zlotkey	SA_MAN
Whalen	AD_ASST
Hartstein	MK_MAN
Fay	MK_REP
Higgins	AC_MGR
Gietz	AC_ACCOUNT

La siguiente tabla muestra el resultado de aplicar el operador NOT a una condición:

Operador NOT	TRUE	FALSE	NULL
	FALSE	TRUE	TRUE

Nota:

El operador NOT también se puede utilizar con otros operadores SQL, como BETWEEN, LIKE y NULL.

```
WHERE job_id    NOT IN ('AC_ACCOUNT', 'AD_VP')
WHERE salary    NOT BETWEEN 10000 AND 15000
WHERE last_name NOT LIKE '%A%'
WHERE commission_pct IS NOT NULL
```

REGLAS DE PRIORIDAD

Las reglas de prioridad determinan el orden en el que se evalúan y se calculan las expresiones. La tabla enumera el orden de prioridad por defecto. Puede sustituir el orden por defecto escribiendo entre paréntesis las expresiones que desee calcular en primer lugar.

Orden de Evaluación	Operador
1	Operadores aritméticos
2	Operador de concatenación
3	Condiciones de comparación
4	IS [NOT] NULL, LIKE, [NOT] IN
5	[NOT] BETWEEN
6	condición lógica NOT
7	condición lógica AND
8	condición lógica OR

En el siguiente ejemplo existen dos condiciones:

- La primera condición es que el identificador de cargo es AD_PRES y el salario mayor que 15.000.
- La segunda condición es que el identificador de cargo es SA_REP.

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

LAST_NAME	JOB_ID	SALARY
King	AD_PRES	24000
Abel	SA_REP	11000
Taylor	SA_REP	8600
Grant	SA_REP	7000

Por lo tanto, la sentencia SELECT indica lo siguiente:

"Seleccionar la fila si un empleado es presidente *y* gana más de $15.000 *o* si el empleado es representante de ventas.".

En el siguiente ejemplo existen dos condiciones:

- La primera condición es que el identificador de cargo es AD_PRES o SA_REP.
- La segunda condición es que el salario es mayor que $15.000.

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

LAST_NAME	JOB_ID	SALARY
King	AD_PRES	24000

Por lo tanto, la sentencia SELECT indica lo siguiente:

"Seleccionar la fila si un empleado es presidente *o* representante de ventas *y* gana más de $15.000."

CLAUSULA ORDER BY

El orden de filas devuelto por una consulta no está definido. Se puede utilizar la cláusula ORDER BY para ordenar las filas. Si lo hace, debe ser la última de la sentencia SQL. Puede especificar una expresión o alias o posición de columna como condición de orden.

Sintaxis

```
SELECT      expr
FROM        table
[WHERE      condition(s)]
[ORDER BY   {column, expr} [ASC|DESC]];
```

En la sintaxis:

ORDER BY	especifica el orden en el que se muestran las filas recuperadas
ASC	ordena las filas en orden ascendente (es el orden por defecto)
DESC	ordena las filas en orden descendente

Si no se utiliza la cláusula ORDER BY, el orden no está definido y es posible que Oracle Server no recupere las filas en el mismo orden dos veces para la misma consulta. Utilice esta cláusula para mostrar las filas en un orden específico.

El siguiente ejemplo muestra los empleados ordenados por fecha de ingreso ascendente

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

LAST_NAME	JOB_ID	DEPARTMENT_ID	HIRE_DATE
King	AD_PRES	90	17-JUN-87
Whalen	AD_ASST	10	17-SEP-87
Kochhar	AD_VP	90	21-SEP-89
Hunold	IT_PROG	60	03-JAN-90
Ernst	IT_PROG	60	21-MAY-91

...

ORDENACION EN ORDEN DESCENDENTE

El orden por defecto es ascendente:

- Los valores numéricos se muestran con los valores más bajos primero; por ejemplo, 1 a 999.
- Los valores de fecha se muestran con la fecha anterior primero; por ejemplo, 01-ENE-92 antes de 01-ENE-95.
- Los valores de caracteres se muestran en orden alfabético; por ejemplo, A en primer lugar y Z en último.
- Los valores nulos se muestran los últimos para las secuencias ascendentes y los primeros para las descendentes.

Reversión al Orden por Defecto

Para revertir el orden en el que se muestran las filas, se especa la palabra clave DESC después del nombre de columna en la cláusula ORDER BY. El ejemplo de la transparencia ordena el resultado según el último empleado contratado:

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

LAST_NAME	JOB_ID	DEPARTMENT_ID	HIRE_DATE
Zlotkey	SA_MAN	80	29-JAN-00
Mourgos	ST_MAN	50	16-NOV-99
Grant	SA_REP		24-MAY-99
Lorentz	IT_PROG	60	07-FEB-99
Vargas	ST_CLERK	50	09-JUL-98
Taylor	SA_REP	80	24-MAR-98
Matos	ST_CLERK	50	15-MAR-98
Fay	MK_REP	20	17-AUG-97
Davies	ST_CLERK	50	29-JAN-97

...

ORDENACION SEGÚN ALIAS DE COLUMNA

Se puede utilizar un alias de columna en la cláusula ORDER BY. En el siguiente ejemplo se ordenan los datos según el salario anual:

```
SELECT employee_id, last_name, salary*12 annsal
FROM   employees
ORDER BY annsal;
```

EMPLOYEE_ID	LAST_NAME	ANNSAL
144	Vargas	30000
143	Matos	31200
142	Davies	37200
141	Rajs	42000
107	Lorentz	50400
200	Whalen	52800
124	Mourgos	69600
104	Ernst	72000
202	Fay	72000
178	Grant	84000

...

ORDENACION SEGÚN MULTIPLES COLUMNAS

Se puede ordenar el resultado de la consulta por más de una columna. El límite de ordenación es el número de columnas de la tabla dada.

En la cláusula ORDER BY, se especifican las columnas separadas mediante comas. Para revertir el orden de una columna se especica DESC después de su nombre. También se puede ordenar por columnas no incluidas en la cláusula SELECT.

En el siguiente ejemplo se muestran los apellidos y los salarios de todos los empleados, ordenados según su número de departamento y después en orden descendente según su salario.

```
SELECT last_name, department_id, salary
FROM   employees
ORDER BY department_id, salary DESC;
```

LAST_NAME	DEPARTMENT_ID	SALARY
Whalen	10	4400
Hartstein	20	13000
Fay	20	6000
Mourgos	50	5800
Rajs	50	3500
Davies	50	3100
Matos	50	2600
Vargas	50	2500

...