



Restringindo e Ordenando Dados



Curso de Introdução a Oracle 10g:
SQL

Prof.: Marlon Mendes Minussi
marlonminussi@gmail.br

- Limitar as linhas recuperadas por uma consulta
- Ordenar as linhas recuperadas por uma consulta
- Quando recuperar dados do banco de dados, você pode necessitar restringir os registros que serão retornados ou especificar a ordem na qual as linhas serão exibidas.



Limitando as Linhas Selecionadas

- Sintaxe:

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

- Você pode restringir as linhas recuperadas pela consulta utilizando a cláusula WHERE.
- A cláusula WHERE contém uma condição que deve ser satisfeita, devendo estar imediatamente após a cláusula FROM.



Limitando as Linhas

Selecionadas

- Na sintaxe:
- WHERE: restringe a consulta para as linhas que satisfazem a condição.
- condition: é composta de nomes de coluna, expressões, constantes, e operadores de comparação.
- A cláusula WHERE pode comparar valores em colunas, valores literais, expressões aritméticas ou funções. A cláusula WHERE consiste de três elementos:
 - Nome da coluna;
 - Operador de comparação;
 - Nome da coluna, constante ou lista de valores.

Utilizando a Cláusula WHERE

- Ex.:

```
SELECT cod_pac, nome, dt_nasc  
FROM   paciente  
WHERE  estado = 'RS';
```

- No exemplo, o comando SELECT recupera o código, nome e a data de nascimento dos pacientes cujo estado é RS.
- Observe que o estado RS foi especificado em maiúsculas para assegurar que a comparação feita com a coluna estado da tabela PACIENTE esteja de acordo com os dados nela armazenados. Strings de caractere fazem distinção entre maiúsculas e minúsculas.

[Strings de Caractere e Datas]

- Ex.:

```
SELECT cod_pac, nome  
FROM   paciente  
WHERE  estado = 'RS';
```
- Strings de caractere e datas na clausula WHERE devem ser inseridas entre aspas simples (').
- Constantes numéricas, entretanto, não devem estar entre aspas.



[Strings de Caractere e Datas]

- Todas as pesquisas tipo caractere fazem distinção entre maiúsculas e minúsculas. No próximo exemplo, nenhuma linha é retornada porque a tabela PACIENTE armazena todos os dados de estado em maiúsculas.
- Ex.:

```
SELECT cod_pac, nome
FROM   paciente
WHERE  estado = 'rs';
```
- O Oracle armazena datas em um formato numérico interno, representando o século, ano, mês, dia, horas, minutos e segundos. A exibição default é DD-MON-YY.



Operadores de Comparação

Operador	Significado
=	Igual a
>	Maior que
>=	Maior que ou igual a
<	Menor que
<=	Menor que ou igual a
<>	Diferente de

**MV**

[Operadores de Comparação]

- Operadores de comparação são utilizados em condições que comparam uma expressão com outra.
- Eles são usados na cláusula WHERE no seguinte formato:
- Sintaxe:
... WHERE expr operator value
- Exemplos:
... WHERE dt_atendimento = '01-ABR-11'
... WHERE cod_atendimento >= 8
... WHERE nome = 'Renato Gaucho'

Utilizando os Operadores de Comparação

- Ex.:

```
SELECT cod_atendimento, dt_atendimento, valor, desconto + 100  
FROM      atendimento  
WHERE     valor <= desconto + 100;
```

- No exemplo, o comando SELECT recupera o código, data de atendimento, valor e desconto mais 100 a partir da tabela ATENDIMENTO, quando o valor for menor ou igual que desconto + 100.
- Os dois valores são comparados são obtidos a partir das colunas VALOR e DESCONTO da tabela ATENDIMENTO.
-

Outros Operadores de Comparação

Operador	Significado
BETWEEN ...AND...	Entre os dois valores (inclusive)
IN(list)	Igual a um dos valores da lista
LIKE	Igual a um padrão de caracteres
IS NULL	Possui um valor nulo



Operador BETWEEN

- Ex.:

```
SELECT nome  
FROM paciente  
WHERE dt_nasc BETWEEN '01-01-60' AND '01-01-70';
```
- Você pode exibir linhas baseadas em um intervalo de valores utilizando o operador BETWEEN. O intervalo que você especifica é composto por um limite inferior e um limite superior.
- O comando SELECT acima recupera linhas da tabela PACIENTE para qualquer paciente cujo dt_nasc esta entre '01-01-60' e '01-01-70.
- Os valores especificados no operador BETWEEN fazem parte do intervalo, sendo também recuperados. Você deve especificar o limite inferior por primeiro.

Operador IN

- Ex.:

```
SELECT cod_pac, nome, cidade  
FROM paciente  
WHERE estado IN ('RS', 'RJ');
```

- Utilize o operador IN para executar a comparação com os valores de uma lista.
- No exemplo exibe cod_pac, nome, cidade dos pacientes pertencentes aos estados 'RS' e 'RJ'.
- O operador IN pode ser utilizado com qualquer tipo de dado. No próximo exemplo recupera uma linha a partir da tabela ATENDIMENTO para cada atendimento com seu código listado na cláusula WHERE.

Operador IN

- Ex.:

```
SELECT cod_atendimento, dt_atendimento, valor, desconto  
FROM atendimento  
WHERE cod_atendimento IN (5, 10,12);
```

- * Se forem utilizadas strings de caractere ou datas na lista, estas devem estar entra aspas simples (').



Operador LIKE

- Ex.:

```
SELECT nome  
FROM paciente  
WHERE nome LIKE 'R%';
```

- Às vezes você pode não saber o valor exato a pesquisar.
- Você pode então selecionar linhas que combinem com um padrão de caracteres utilizando o operador LIKE.



Operador LIKE

- Podem ser utilizados dois símbolos para construir a string de procura.

Símbolo	Descrição
%	Representa qualquer seqüência de zero ou mais caracteres
—	Representa um único caractere qualquer



Operador LIKE

- O comando SELECT acima recupera o nome dos pacientes da tabela PACIENTE para qualquer paciente que o nome começa com um “R”. Observe que os nomes que começam com um “r” minúsculo não serão recuperados.
- O operador LIKE pode ser utilizado como um atalho para algumas comparações – normalmente executadas com o operador BETWEEN. O exemplo exhibe os códigos e nomes de todos os pacientes que nasceram entre Janeiro de 1973 e dezembro de 1973.
- Ex.:

```
SELECT cod_pac, nome  
FROM paciente  
WHERE dt_nasc LIKE '%73';
```



Combinando Padrões de Caractere

- Os símbolos % e _ podem ser utilizados em qualquer combinação com literais de caracteres. O exemplo mostra os nomes de todos os pacientes cujo nome possui a letra “A” como o segundo caractere.
- Ex.:

```
SELECT cod_pac, nome, cidade
FROM   paciente
WHERE  nome LIKE ‘_A%’;
```



Operador IS NULL

- Ex.:

```
SELECT cod_atendimento, dt_atendimento, valor, desconto  
FROM      atendimento  
WHERE     desconto IS NULL;
```

- O operador IS NULL testa os valores que são nulos.
- Um valor nulo é um valor que é indisponível, não atribuído, desconhecido ou inaplicável.
- Portanto, você não pode testá-los com (=) porque um valor nulo não pode ser igual ou diferente de qualquer valor.
- O exemplo recupera os atendimentos que não possuem desconto.



Operadores Lógicos

Operador	Significado
AND	Retorna TRUE se ambas as condições resultarem em TRUE
OR	Retorna TRUE se qualquer das condições retornarem TRUE
NOT	Retorna TRUE se a condição seguinte retornar FALSE



Operadores Lógicos

- Um operador lógico combina o resultado de duas condições para produzir um único resultado ou para inverter o resultado de uma única condição.
- Três operadores lógicos estão disponíveis em SQL:
 - AND
 - OR
 - NOT
- Você pode usar varias condições em uma cláusula WHERE utilizando os operadores AND e OR.

Operador AND

- Ex.:

```
SELECT cod_atendimento, dt_atendimento, valor, desconto  
FROM atendimento  
WHERE valor >=100  
AND desconto IS NOT NULL;
```
- No exemplo, ambas as condições devem ser verdadeiras para qualquer registro a ser selecionado.
- Portanto, um atendimento que tem o valor ≥ 100 e possua desconto será selecionado.
- Todas as pesquisas tipo caractere fazem distinção entre maiúsculas e minúsculas.

Combinções de Resultados com o Operador AND

- Resultado da combinação de duas expressões com o operador AND:

AND	TRUE	FALSE	UNKNOW
TRUE	TRUE	FALSE	UNKNOW
FALSE	FALSE	FALSE	FALSE
UNKNOW	UNKNOW	UNKNOW	UNKNOW

Operador OR

- Ex.:

```
SELECT cod_atendimento, dt_atendimento, valor, desconto  
FROM atendimento  
WHERE valor >=100  
OR desconto IS NOT NULL;
```
- No exemplo, para um registro ser selecionado basta que uma das duas condições seja verdadeira. Portanto, um atendimento possui desconto ou tem valor maior ou igual a R\$100 será selecionado.

Combinações de Resultados com o Operador OR

- Resultado da combinação de duas expressões com o operador OR:

OR	TRUE	FALSE	UNKNOW
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	UNKNOW
UNKNOW	TRUE	UNKNOW	UNKNOW



Operador NOT

- Ex.:

```
SELECT nome, dt_nasc, cidade  
FROM paciente  
WHERE cidade NOT IN ('Porto Alegre', 'Passo Fundo');
```

- O exemplo exhibe o nome, a data de nascimento e a cidade de todos os pacientes cuja a cidade não seja 'Porto Alegre' ou 'Passo Fundo' .



Combinações de Resultados com o Operador NOT

- Resultado da aplicação do operador NOT a uma condição:

NOT	TRUE	FALSE	UNKNOWN
	FALSE	TRUE	UNKNOWN

- Nota: o operador NOT também pode ser utilizado com outros operadores SQL como BETWEEN, LIKE e NULL.
 - ... WHERE estado NOT IN ('RJ', 'RS')
 - ... WHERE valor NOT BETWEEN 75,5 AND 100
 - ... WHERE nome NOT LIKE '%A%'
 - ... WHERE desconto IS NOT NULL



Regras de Precedência

Ordem	Operador
1	Todos os operadores de comparação
2	NOT
3	AND
4	OR




Regras de Precedência

- Exemplo de Precedência do Operador AND

Ex.:

```
SELECT cod_atendimento, cod_pac, valor  
FROM atendimento  
WHERE valor < 75  
OR      valor > 100  
AND     desconto IS NULL;
```




- No exemplo, existem duas condições:
- A primeira condição é aquela onde o valor deve ser menor que R\$75;
- A segunda condição é aquela onde o valor deve ser maior que R\$100, mas não deve ter desconto.

Utilizando Parênteses para Alterar a Prioridade

Ex.:

```
SELECT cod_atendimento, cod_pac, valor  
FROM atendimento  
WHERE (valor < 75  
OR valor > 100)  
AND desconto IS NULL;
```



- No exemplo, existem duas condições:
- A primeira condição é aquela onde o valor do curso seja menor que R\$75 ou maior que R\$100.
- A segunda condição é aquela onde o desconto deve ser nulo.
- Portanto lê-se o comando SELECT como segue:
- “selecione a linha se o curso possuir o valor menor que R\$75 ou maior que R\$100 e não possua desconto.



Cláusula ORDER BY

- Ex.:

```
SELECT cod_pac, nome, dt_nasc  
FROM     paciente  
ORDER BY nome;
```

- A ordem das linhas recuperadas no resultado de uma consulta é indefinida. A cláusula ORDER BY pode ser utilizada para ordenar as linhas. Se utilizada, deve aparecer como a última cláusula do comando SELECT. Você pode ordenar por uma expressão ou por um alias de coluna.

Cláusula ORDER BY

- Sintaxe:

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

- Onde:
- ORDER BY: especifica a ordem na qual as linhas recuperadas serão exibidas;
- ASC: ordena as linhas em ordem ascendente. Esta é a ordenação padrão;
- DESC: ordena as linhas em ordem decrescente.



Cláusula ORDER BY

- Se a cláusula ORDER BY não for utilizada, a ordem em que as linhas serão recuperadas é indefinida, e o Servidor Oracle pode não recuperar as linhas duas vezes na mesma ordem para a mesma consulta.
- Utilize a cláusula ORDER BY para exibir as linhas em uma ordem específica.
- Ainda é possível realizar a ordenação começando pelos valores nulos ou terminando pelos valores nulos:
- NULLS FIRST
- NULLS LAST
- Classificando em Ordem Decrescente

```
SELECT nome, dt_nasc, cidade, estado  
FROM paciente  
ORDER BY nome DESC;
```

Ordenação Padrão dos Dados

- A ordenação padrão dos dados é ascendente:
- Valores numéricos são exibidos com o valor mais baixo primeiro, por exemplo: 1-999.
- Valores tipo data são exibidos com o valor mais antigo primeiro, por exemplo: 01-JAN-11 antes de 01-MAR-11.
- Valores caractere são exibidos em ordem alfabética, por exemplo: A primeiro e Z por último.
- Valores nulos são exibidos por último para ordenações ascendentes e por primeiro para ordenações descendentes.



Ordenação Padrão dos Dados

- Invertendo a Ordem Padrão
- Para inverter a ordem na qual as linhas são exibidas, especifique o palavra chave DESC depois do nome da coluna na cláusula ORDER BY.
- O exemplo acima ordena o resultado pelo nome do cliente em ordem decrescente.

Ordenando pelo Alias de Coluna

Ex.:

```
SELECT cod_atendimento, dt_atendimento, valor VALOR_CONSULTA  
FROM      atendimento  
ORDER BY 2 NULLS LAST, VALOR_CONSULTA;
```

- Você pode utilizar um alias de coluna na cláusula ORDER BY.
- O exemplo ordena os dados pelo valor de desconto (coluna 2 do SELECT) e o total (VALOR).

Ordenando por Múltiplas Colunas

Ex.:

```
SELECT nome, estado, dt_nasc  
FROM      paciente  
ORDER BY estado, nome DESC;
```

- Você pode ordenar os resultados das colunas por mais de uma coluna.
- Na cláusula ORDER BY, especifique as colunas separando-as por vírgulas. Se você quiser inverter a ordem de uma coluna, especifique DESC depois de seu nome. Você pode ordenar por colunas que não estão incluídas na lista da cláusula SELECT.
- No exemplo, ordenamos todos os clientes da tabela TCLIENTES pelo estado em ordem crescente e depois pelo nome em ordem decrescente.