



Estrutura e Banco de Dados

**Inserindo, alterando, excluindo registros
e recuperando informações no
banco de dados**

Prof. Claudiney Sanches

Agenda

Inserir linhas em tabelas

INSERT

UPDATE

DELETE

RENAME

CREATE TABLE ... AS SELECT

Recuperar Dados de uma Tabela

SELECT

Distinct

Expressões Aritméticas

Cláusula Where

Alias

Exercícios

Bibliografia

Pré requisito para este material

Se logar com o usuário HR e Selecionar SQL Commands

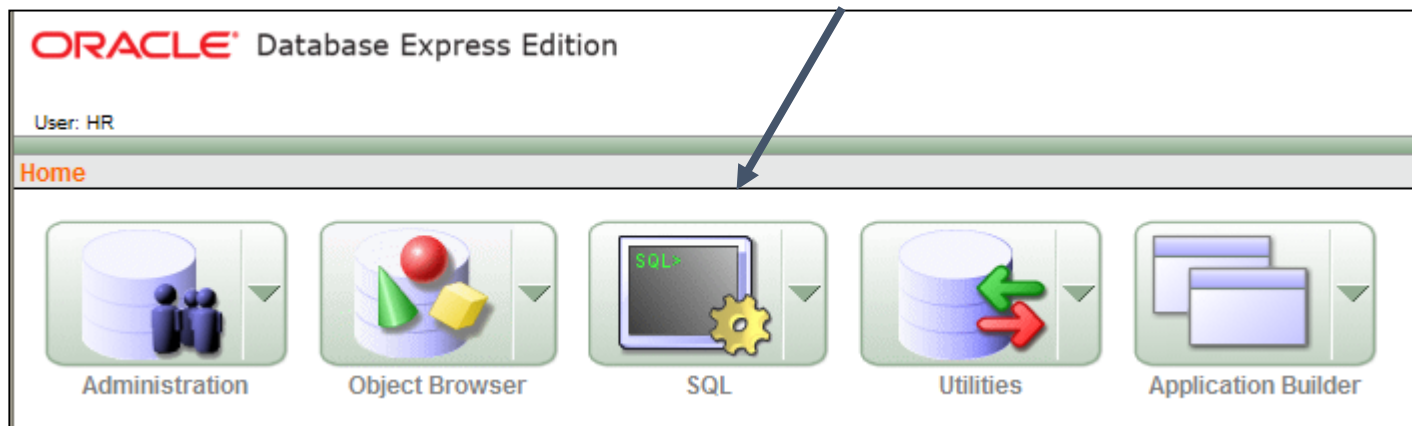
ORACLE® Database Express Edition

Database Login
Enter your database username and password.
Username
Password

[Click here to learn how to get started](#)

Links

- [License Agreement](#)
- [Documentation](#)
- [Forum Registration](#)
- [Discussion Forum](#)
- [Product Page](#)



Pré requisito para este material

Criar a tabela "pais" conforme script abaixo

```
CREATE TABLE pais
(Cd_pais number (2) ,
Nm_pais char (45) ,
Ds_nacionalidade char(35) ,
constraint pais_Cd_pais_pk primary key (Cd_pais) ,
constraint pais_Nm_pais_uk unique (Nm_pais) )
```

Inserindo registros nas tabelas

Sintaxe:

```
INSERT INTO nome_da_tabela  
    [(nome_da_coluna [,nome_da_coluna])]  
VALUES  
    (valor_da_coluna [,valor_da_coluna])
```

Valores alfanuméricos e
datas deverão estar
entre aspas simples (' ')

Ex 1:

```
INSERT INTO pais (cd_pais, nm_pais)  
VALUES (1, 'Brasil')
```

Ex 2:

```
INSERT INTO pais (cd_pais,  
                  nm_pais, ds_nacionalidade)  
VALUES (2, 'Argentina', null);
```

Ex 3:

```
INSERT INTO pais  
VALUES (4, 'Chile', 'Chilena');
```

Ex 4:

```
INSERT INTO pais (nm_pais,  
                  ds_nacionalidade, cd_pais)  
VALUES ('EUA', 'Americana', 3);
```

Inserindo registros nas tabelas

Para incluir dados em uma tabela pode-se ou não incluir valores em todas as suas colunas (caso não sejam obrigatórias).

```
CREATE TABLE pais
(Cd_pais number (2),
Nm_pais char (45),
Ds_nacionalidade char(35),
constraint pais_Cd_pais_pk
    primary key (Cd_pais),
constraint pais_Nm_pais_uk
    unique(Nm_pais))
```

```
Ex 1:
INSERT INTO pais (cd_pais, nm_pais)
VALUES (1, 'Brasil')
```

```
Ex 2:
INSERT INTO pais
VALUES (3, 'Chile', 'Chilena');
```

A relação de colunas da tabela poderá ser omitida quando forem inclusos valores para todas as colunas dessa tabela.

Modificar a(s) linha(s) existentes com o comando UPDATE.

```
UPDATE Tabela  
SET COLUNA = Valor  
WHERE Condição
```

```
Update Pais  
set Nm_pais='Portugal',  
    Ds_nacionalidade='Portug'  
Where Cd_pais = 3;
```

1 linha atualizada.

```
Update Pais  
set Ds_nacionalidade='Portuguesa'  
Where Cd_pais = 3;
```

1 linha atualizada.

Remover a(s) linha(s) existente(s) com o comando DELETE.

```
DELETE FROM Tabela  
WHERE Condição
```

Observações:

- . Verifique os nomes das colunas com o comando DESCRIBE.
- . Confirme a operação de atualização pela visualização das linhas a serem atualizadas com o comando SELECT.
- . Nunca omita a cláusula WHERE. No caso da omissão, todos os registros da tabela serão eliminados.

```
DELETE FROM Pais WHERE Cd_pais = 3;
```

1 linha deletada.

Este comando é utilizado nos casos de alteração de nome das tabelas.

```
RENAME nome_antigo_tabela TO nome_novo_tabela
```

```
RENAME Pais TO Tb_Pais
```

CREATE TABLE AS SELECT

Este comando cria uma tabela (estrutura e dados) baseado em um comando **SELECT**.

```
create table espelhoteste  
as select * from employees;
```

SELECT * FROM employees;

Results Explain Describe Saved SQL History

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER
100	Steven	King	SKING	515.123.4567
101	Neena	Kochhar	NKOCHHAR	515.123.4568
102	Lex	De Haan	LDEHAAN	515.123.4569
103	Alexander	Hunold	AHUNOLD	590.423.4567
104	Bruce	Ernst	BERNST	590.423.4568
105	David	Austin	DAUSTIN	590.423.4569
106	Valli	Pataballa	VPATABAL	590.423.4560
107	Diana	Lorentz	DLORENTZ	590.423.5567
108	Nancy	Greenberg	NGREENBE	515.124.4569
109	Daniel	Faviet	DFAVIET	515.124.4169

More than 10 rows available. Increase rows selector to view more rows.

SELECT * FROM espelhoteste;

Results Explain Describe Saved SQL History

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER
100	Steven	King	SKING	515.123.4567
101	Neena	Kochhar	NKOCHHAR	515.123.4568
102	Lex	De Haan	LDEHAAN	515.123.4569
103	Alexander	Hunold	AHUNOLD	590.423.4567
104	Bruce	Ernst	BERNST	590.423.4568
105	David	Austin	DAUSTIN	590.423.4569
106	Valli	Pataballa	VPATABAL	590.423.4560
107	Diana	Lorentz	DLORENTZ	590.423.5567
108	Nancy	Greenberg	NGREENBE	515.124.4569
109	Daniel	Faviet	DFAVIET	515.124.4169

More than 10 rows available. Increase rows selector to view more rows.

O comando **SELECT** permite a seleção e a manipulação, para visualização das informações armazenadas no banco de dados.

SELECT <<colunas que quero pesquisar>>
FROM <<tabelas que quero pesquisar>>






Sintaxe:

```
SELECT [distinct] col1, col2, coln  
FROM nome da tabela [alias]  
WHERE condição  
GROUP BY colunas  
HAVING condição  
ORDER BY expressão ou chave [desc]
```

EXEMPLOS:

O comando SELECT permite selecionar:

(Para selecionar essas tabelas, se logar com o usuário HR no Oracle)

- A) Todas as colunas  `SELECT * FROM departments;`
- B) Colunas específicas  `SELECT FIRST_NAME, EMAIL,
HIRE_DATE FROM employees;`
- C) Expressões Aritméticas  `SELECT FIRST_NAME, EMAIL ,
salary, salary*12 FROM employees;`
- D) Colunas com Apelidos
(alias)  `SELECT FIRST_NAME, EMAIL, salary
"Salário", salary*12 "Salario Anual"
FROM employees;`
- E) Colunas concatenadas  `SELECT FIRST_NAME, EMAIL,
'Salário: ' || salary "Salário",
salary*12 "Salario Anual" FROM
employees;`

EXEMPLOS:

**SELECT FIRST_NAME , EMAIL,
HIRE_DATE FROM employees;**

FIRST_NAME	EMAIL	HIRE_DATE
Steven	SKING	17-JUN-87
Neena	NKOCHHAR	21-SEP-89
Lex	LDEHAAN	13-JAN-93
Alexander	AHUNOLD	03-JAN-90
Bruce	BERNST	21-MAY-91
David	DAUSTIN	25-JUN-97
Valli	VPATABAL	05-FEB-98
Diana	DLORENTZ	07-FEB-99
Nancy	NGREENBE	17-AUG-94
Daniel	DFAVIET	16-AUG-94
More than 10 rows available. Increase rows selector to view more rows.		

SELECT * FROM departments;

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
30	Purchasing	114	1700
40	Human Resources	203	2400
50	Shipping	121	1500
60	IT	103	1400
70	Public Relations	204	2700
80	Sales	145	2500
90	Executive	100	1700
100	Finance	108	1700
More than 10 rows available. Increase rows selector to view more rows.			

Quando mostramos o resultado de uma pesquisa, normalmente é retornado o nome das colunas selecionadas como cabeçalho.

Em alguns exemplos ele pode ser sem sentido.

Você pode modificar o cabeçalho de uma coluna usando sinônimos(alias).

Basta colocar o [alias] depois da coluna no comando select.

```
SELECT FIRST_NAME , EMAIL ,  
salary, salary*12 FROM employees;
```

FIRST_NAME	EMAIL	SALARY	SALARY*12
Steven	SKING	24000	288000
Neena	NKOCHHAR	17000	204000
Lex	LDEHAAN	17000	204000
Alexander	AHUNOLD	9000	108000
Bruce	BERNST	6000	72000
David	DAUSTIN	4800	57600
Valli	VPATABAL	4800	57600
Diana	DLORENTZ	4200	50400
Nancy	NGREENBE	12000	144000
Daniel	DFAVIET	9000	108000
More than 10 rows available. Increase rows selector to view more rows.			

```
SELECT FIRST_NAME , EMAIL , salary  
"Salário", salary*12 "Salario Anual"  
FROM employees;
```

FIRST_NAME	EMAIL	Salário	Salario Anual
Steven	SKING	24000	288000
Neena	NKOCHHAR	17000	204000
Lex	LDEHAAN	17000	204000
Alexander	AHUNOLD	9000	108000
Bruce	BERNST	6000	72000
David	DAUSTIN	4800	57600
Valli	VPATABAL	4800	57600
Diana	DLORENTZ	4200	50400
Nancy	NGREENBE	12000	144000
Daniel	DFAVIET	9000	108000
More than 10 rows available. Increase rows selector to view more rows.			

Linhas Duplicadas

A exibição default de consultas são todas as linhas, incluindo as linhas duplicadas

Para eliminar valores duplicados no resultado, incluimos o DISTINCT qualificador no comando SELECT

```
SELECT department_id  
FROM employees  
ORDER BY department_id;
```

DEPARTMENT_ID
10
20
20
30
30
30
30
30
30
40
More than 10 rows available. Increase rows selector to view more rows.

```
SELECT distinct department_id  
FROM employees  
ORDER BY department_id;
```

DEPARTMENT_ID
10
20
30
40
50
60
70
80
90
100
More than 10 rows available. Increase rows selector to view more rows.

Expressões Aritméticas podem conter nome de colunas, valores numéricos constantes e operadores aritméticos:

Operadores	Descrições
+	Adição
-	Subtração
*	Multiplicação
/	Divisão

```
SELECT FIRST_NAME , EMAIL ,  
salary, salary*12 FROM employees;
```

FIRST_NAME	EMAIL	SALARY	SALARY*12
Steven	SKING	24000	288000
Neena	NKOCHHAR	17000	204000
Lex	LDEHAAN	17000	204000
Alexander	AHUNOLD	9000	108000
Bruce	BERNST	6000	72000
David	DAUSTIN	4800	57600
Valli	VPATABAL	4800	57600
Diana	DLORENTZ	4200	50400
Nancy	NGREENBE	12000	144000
Daniel	DFAVIET	9000	108000
More than 10 rows available. Increase rows selector to view more rows.			

A Cláusula WHERE indica condição para um SELECT, no qual pode ter os seguintes operadores:

Operador	Descrição
=	Igual
<>	Diferente
>	Maior
<	Menor
>=	Maior ou igual
<=	Menor ou igual

Alfanuméricos e Datas na cláusula WHERE devem estar entre aspas simples

Operadores SQL

Existem quatro operadores SQL os quais opera, com todos tipos de dados:

Operador	Significado
Between ... and ...	Entre dois valores (inclusive)
In (lista)	Comparar uma lista de valores
Like	Compara um parâmetro alfanumérico
Is Null	É um valor nulo

O Operador BETWEEN

Compara uma faixa de valores inclusive o menor e maior valor.
Suponhamos que nós quisemos ver aqueles empregados os quais o salário está entre 1000 e 2000:

```
SELECT First_name, Last_name, Salary  
FROM Employees  
WHERE Salary BETWEEN 10000 AND 20000;
```

FIRST_NAME	LAST_NAME	SALARY
Neena	Kochhar	17000
Lex	De Haan	17000
Nancy	Greenberg	12000
Den	Raphaely	11000
John	Russell	14000
Karen	Partners	13500
Alberto	Errazuriz	12000
Gerald	Cambraut	11000
Eleni	Zlotkey	10500
Peter	Tucker	10000
More than 10 rows available. Increase rows selector to view more rows.		

Note que os valores especificados estão inclusive, e o menor precisa ser especificado primeiro.

O Operador IN

Compara os valores especificados dentro de uma lista.

Para encontrar empregados que tenham um dos três números de Manager, utilize o seguinte comando:

```
SELECT First_name, Last_name, Salary, Manager_id  
FROM Employees  
WHERE Manager_id IN (101, 145, 149);
```

Se alfanuméricos ou datas forem usados na lista precisam ser colocados entre aspas simples(' ').

FIRST_NAME	LAST_NAME	SALARY	MANAGER_ID
Nancy	Greenberg	12000	101
Jennifer	Whalen	4400	101
Susan	Mavris	6500	101
Hermann	Baer	10000	101
Shelley	Higgins	12000	101
Peter	Tucker	10000	145
David	Bernstein	9500	145
Peter	Hall	9000	145
Christopher	Olsen	8000	145
Nanette	Cambrault	7500	145
Oliver	Tuvault	7000	145
Ellen	Abel	11000	149
Alyssa	Hutton	8800	149
Jonathon	Taylor	8600	149
Jack	Livingston	8400	149
Kimberely	Grant	7000	149
Charles	Johnson	6200	149

O Operador LIKE

Algumas vezes você precisa procurar valores que você não conhece exatamente. Usando o operador LIKE é possível selecionar linhas combinando parâmetros alfanuméricos. O Caractere % é utilizado como coringa nas pesquisas. Veja exemplo abaixo:

Sintaxe	Descrição
Like '%cadeia'	Localiza valores que terminem com a cadeia de caracteres
Like 'cadeia%'	Localiza valores que comecem com a cadeia de caracteres
Like '%cadeia%'	Localiza valores que tenham a cadeia de caracteres em qualquer parte do campo

O Operador LIKE

Para listar todos os empregados que tenham o nome que comecem com a letra S, faça:

```
SELECT First_name, Last_name, Salary  
FROM Employees  
WHERE First_name like 'S%';
```

FIRST_NAME	LAST_NAME	SALARY
Steven	King	24000
Shelli	Baida	2900
Sigal	Tobias	2800
Shanta	Vollman	6500
Steven	Markle	2200
Stephen	Stiles	3200
Sarath	Sewall	7000
Sundar	Ande	6400
Sundita	Kumar	6100
Sarah	Bell	4000
More than 10 rows available. Increase rows selector to view more rows.		

O Operador IS NULL

Verifica quais campos que estão com valores nulos

Unicamente encontrar todos os empregados que não tenham gerente, você testará um valor nulo:

```
SELECT First_name, Last_name, Salary, Manager_id  
FROM Employees  
WHERE Manager_id IS Null;
```

FIRST_NAME	LAST_NAME	SALARY	MANAGER_ID
Steven	King	24000	-

Expressões Negativas

Podemos também negar todos os operadores estudados:

Sintaxe	Descrição
NOT Between	tudo que estiver fora da faixa
NOT In	tudo que não estiver na lista
NOT Like	tudo que não conter a linha de caracteres
IS NOT Null	tudo que não for nulo

Expressões Negativas

Execute os seguintes comandos e veja o Resultado:

```
SELECT First_name, Last_name, Salary  
FROM Employees  
WHERE Salary NOT BETWEEN 10000 AND 20000;
```

```
SELECT First_name, Last_name, Salary, Manager_id  
FROM Employees  
WHERE Manager_id NOT IN (101, 145, 149);
```

```
SELECT First_name, Last_name, Salary  
FROM Employees  
WHERE First_name NOT Like 'S%';
```

```
SELECT First_name, Last_name, Salary, Manager_id  
FROM Employees  
WHERE Manager_id IS NOT Null;
```

PESQUISANDO DADOS COM MÚLTIPLAS CONDIÇÕES

- Os operadores AND e OR devem ser usados para fazer composições de expressões lógicas.
- O predicado AND esperará que ambas as condições sejam verdadeiras;
- O predicado OR esperará uma das condições seja verdadeira.

Nos dois exemplos seguintes as condições são as mesmas, mas o predicado é diferente.

Veja como o resultado é dramaticamente alterado.

Para encontrar todos os escriturários do departamento estoque que ganhem entre 2000 e 3000.

```
SELECT First_name, Last_name, Salary, Job_ID  
FROM Employees  
WHERE Salary BETWEEN 2000 AND 3000  
AND Job_id = 'ST_CLERK';
```

FIRST_NAME	LAST_NAME	SALARY	JOB_ID
Irene	Mikkilineni	2700	ST_CLERK
James	Landry	2400	ST_CLERK
Steven	Markle	2200	ST_CLERK
Mozhe	Atkinson	2800	ST_CLERK
James	Marlow	2500	ST_CLERK
TJ	Olson	2100	ST_CLERK
Michael	Rogers	2900	ST_CLERK
Ki	Gee	2400	ST_CLERK
Hazel	Philtanker	2200	ST_CLERK
John	Seo	2700	ST_CLERK
Joshua	Patel	2500	ST_CLERK
Randall	Matos	2600	ST_CLERK
Peter	Vargas	2500	ST_CLERK

SELECT

Para encontrar todos os escriturários do departamento estoque e os funcionários que ganhem entre 2000 e 3000.

```
SELECT First_name, Last_name, Salary  
FROM Employees  
WHERE Salary BETWEEN 2000 AND 3000  
OR Job_id = 'ST_CLERK';
```

Perceba que a seleção foi bem maior, visto que existem funcionários que ganham entre 2000 e 3000 e não são Escriturários do departamento de Estoque.

FIRST_NAME	LAST_NAME	SALARY	JOB_ID
Shelli	Baida	2900	PU_CLERK
Sigal	Tobias	2800	PU_CLERK
Guy	Himuro	2600	PU_CLERK
Karen	Colmenares	2500	PU_CLERK
Julia	Nayer	3200	ST_CLERK
Irene	Mikkilineni	2700	ST_CLERK
James	Landry	2400	ST_CLERK
Steven	Markle	2200	ST_CLERK
Laura	Bissot	3300	ST_CLERK
Mozhe	Atkinson	2800	ST_CLERK
James	Marlow	2500	ST_CLERK
TJ	Olson	2100	ST_CLERK
Jason	Mallin	3300	ST_CLERK
Michael	Rogers	2900	ST_CLERK
Ki	Gee	2400	ST_CLERK
Hazel	Philtanker	2200	ST_CLERK
Renske	Ladwig	3600	ST_CLERK
Stephen	Stiles	3200	ST_CLERK
John	Seo	2700	ST_CLERK
Joshua	Patel	2500	ST_CLERK
Trenna	Rajs	3500	ST_CLERK
Curtis	Davies	3100	ST_CLERK
Randall	Matos	2600	ST_CLERK
Peter	Vargas	2500	ST_CLERK
Martha	Sullivan	2500	SH_CLERK
Girard	Geoni	2800	SH_CLERK
Anthony	Cabrio	3000	SH_CLERK
Timothy	Gates	2900	SH_CLERK
Randall	Perkins	2500	SH_CLERK
Vance	Jones	2800	SH_CLERK
Kevin	Feeney	3000	SH_CLERK
Donald	OConnell	2600	SH_CLERK

Você pode combinar AND e OR na mesma expressão lógica. Quando AND e OR aparecer na mesma cláusula WHERE, todos os ANDs serão feitos primeiros e posteriormente todos os "Ors" serão feitos.

```
SELECT First_name, Last_name, Salary, Job_ID
FROM   Employees
WHERE  Salary > 8000 and
       Job_id = 'ST_CLERK' OR
       Job_id = 'ST_MAN'
```

Perceba que existem funcionários ST_MAN que ganham menos que 8000, pois nesta condição somente satisfaz a cláusula OR.

FIRST_NAME	LAST_NAME	SALARY	JOB_ID
Matthew	Weiss	8000	ST_MAN
Adam	Fripp	8200	ST_MAN
Payam	Kaufling	7900	ST_MAN
Shanta	Vollman	6500	ST_MAN
Kevin	Mourgos	5800	ST_MAN

Se você quiser selecionar todos os Escriturários e Gerentes que ganhem acima de 8000, deverá fazer:

```
SELECT First_name, Last_name, Salary, Job_ID
FROM   Employees
WHERE  Salary > 8000 and
       (Job_id = 'ST_CLERK' OR
        Job_id = 'ST_MAN')
```

Perceba que existe somente o funcionário Adam que satisfaz a expressão acima.

FIRST_NAME	LAST_NAME	SALARY	JOB_ID
Adam	Frupp	8200	ST_MAN

Os parênteses especificam, a ordem na qual os operadores devem ser avaliados (prioridade).

No segundo exemplo, o operador OR é avaliado antes do AND.

Sempre que você estiver em dúvida sobre qual dos dois operadores será feito primeiro quando a expressão é avaliada, use sempre parênteses para definir a prioridade das expressões.

A cláusula ORDER BY

Normalmente a ordem das linhas retornadas de uma pesquisa é indefinida. A cláusula ORDER BY pode ser usada para ordenar as linhas.

Se usado, o ORDER BY deve ser sempre a última cláusula da declaração SELECT. Para ordenar essa consulta pelo campo Last_Name, faça:

```
SELECT First_name, Last_name, Salary
FROM Employees
WHERE Salary BETWEEN 10000 AND 20000
ORDER BY Last_name;
```

FIRST_NAME	LAST_NAME	SALARY
Ellen	Abel	11000
Hermann	Baer	10000
Harrison	Bloom	10000
Gerald	Cambraut	11000
Lex	De Haan	17000
Alberto	Errazuriz	12000
Nancy	Greenberg	12000
Michael	Hartstein	13000
Shelley	Higgins	12000
Janette	King	10000
Neena	Kochhar	17000
Lisa	Ozer	11500
Karen	Partners	13500
Den	Raphaely	11000
John	Russell	14000
Peter	Tucker	10000
Clara	Vishney	10500
Eleni	Zlotkey	10500

Padrão da Ordenação dos Dados

O padrão da ordem de ordenação é ascendente.

- valores numéricos infinitos primeiro
- valores de data primeiro
- valores alfanuméricos

Invertendo o padrão de ordenação

Para inverter essa ordem, acrescente o comando DESC (Decrescente) do depois do nome das colunas da cláusula ORDER BY.

Para inverter a ordem da coluna Last_name, faça:

```
SELECT First_name, Last_name, Salary
FROM   Employees
WHERE  Salary BETWEEN 10000 AND 20000
ORDER BY Last_name DESC;
```

FIRST_NAME	LAST_NAME	SALARY
Eleni	Zlotkey	10500
Clara	Vishney	10500
Peter	Tucker	10000
John	Russell	14000
Den	Raphaely	11000
Karen	Partners	13500
Lisa	Ozer	11500
Neena	Kochhar	17000
Janette	King	10000
Shelley	Higgins	12000
Michael	Hartstein	13000
Nancy	Greenberg	12000
Alberto	Errazuriz	12000
Lex	De Haan	17000
Gerald	Cambrault	11000
Harrison	Bloom	10000
Hermann	Baer	10000
Ellen	Abel	11000

Ordenação por várias colunas.

É possível utilizar mais de uma coluna na cláusula ORDER BY.

O limite de colunas é o número de colunas da tabela. Na cláusula ORDER BY especifica-se as colunas que serão ordenadas, separando as por vírgula. Se algumas ou todas serão invertidas especifique DESC depois de cada uma das colunas. Para ordenar por duas colunas, e mostrar ordem inversa do salário, e ordem crescente do sobrenome faça:

```
SELECT First_name, Last_name, Salary
FROM Employees
WHERE Salary BETWEEN 10000 AND 20000
ORDER BY Salary Desc, Last_name;
```

FIRST_NAME	LAST_NAME	SALARY
Lex	De Haan	17000
Neena	Kochhar	17000
John	Russell	14000
Karen	Partners	13500
Michael	Hartstein	13000
Alberto	Errazuriz	12000
Nancy	Greenberg	12000
Shelley	Higgins	12000
Lisa	Ozer	11500
Ellen	Abel	11000
Gerald	Cambrault	11000
Den	Raphaely	11000
Clara	Vishney	10500
Eleni	Zlotkey	10500
Hermann	Baer	10000
Harrison	Bloom	10000
Janette	King	10000
Peter	Tucker	10000

Ordenação por várias colunas.

Para ordenar por uma coluna, ela não precisa necessariamente estar declarada no SELECT.

A cláusula ORDER BY é usada na pesquisa quando você quer mostrar as linhas em uma ordem específica. Sem a cláusula ORDER BY as linhas são retornadas na ordem conveniente para o ORACLE.

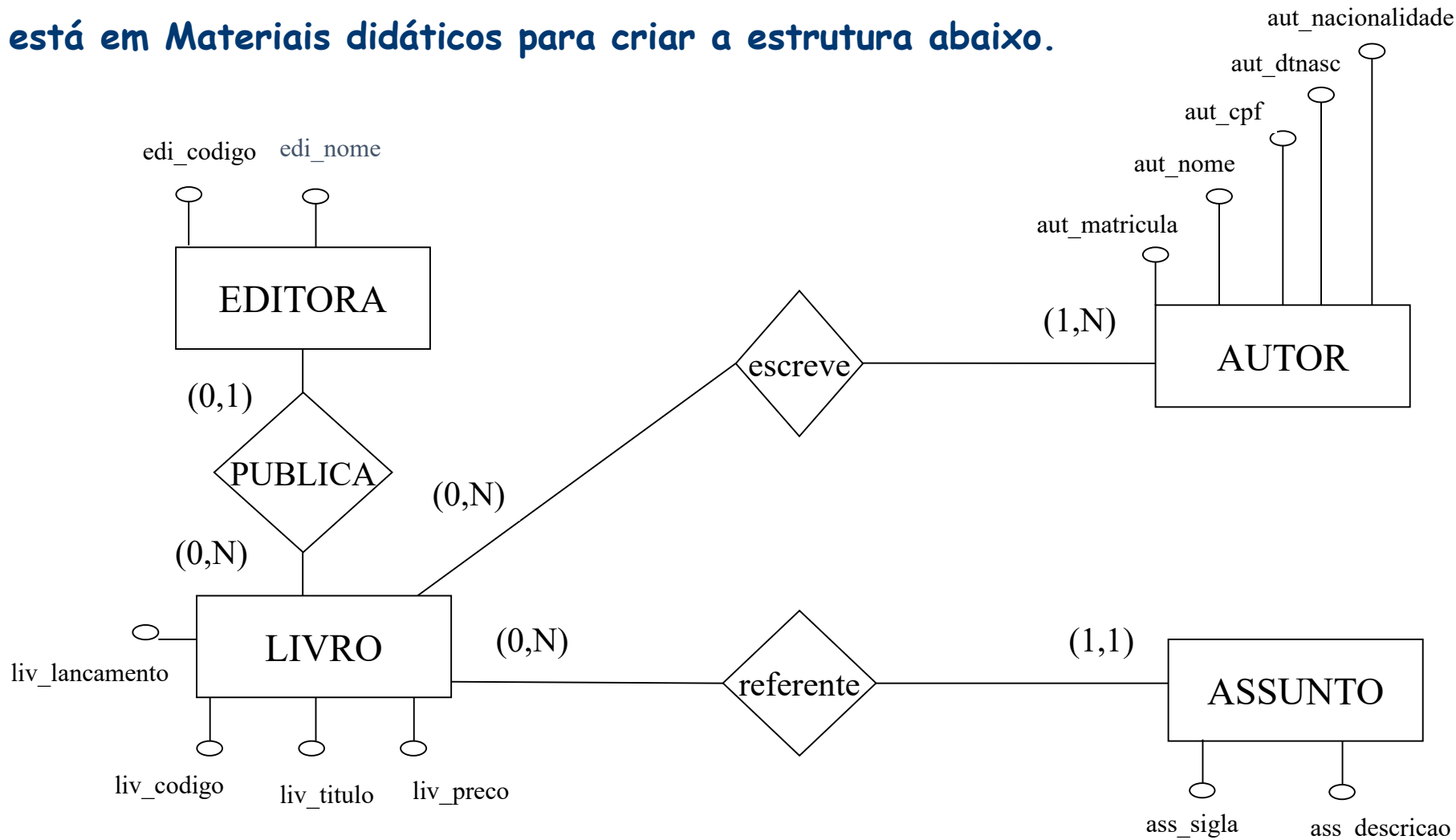
Esse comando não altera a ordem dos dados que estão armazenados no Banco de Dados.

RESUMO:

Sintaxe	Descrição
SELECT	Seleciona no mínimo uma coluna
Sinônimo(Alias)	Pode ser usado para colunas unicamente na lista do SELECT
*	Indica todas as colunas
DISTINCT	Pode ser usado para eliminar duplicações.
FROM Tabela	Indica a tabela de onde as colunas originam.
Where	Restringe a pesquisa para linhas que encontram a condição. Ele pode conter colunas, e literais
And / Or	Podem ser usados na clausula WHERE para construir query's mais complexas nas condições, AND tem prioridade sobre o OR.
()	Pode ser usado para forçar prioridade.
Order by	Especifica a ordem do Select. Uma ou mais colunas podem ser especificadas.
ASC	Ordem ascendente é padrão ordem de ordenação não precisa ser especificado.
DESC	Inverte a ordem padrão de ordenação e deve ser especificada depois do nome da coluna.

Salve os comandos no arquivo **Aula2.sql** e envie via Link.

1. Montar os comandos de insert para popular o der abaixo, conforme os dados do próximo slide. Utilize o Script **CriaEstrutura.sql** que está em Materiais didáticos para criar a estrutura abaixo.



Exercícios

LIVRO					
liv_codigo	liv_titulo	liv_preco	liv_lancamento	edi_codigo	ass_sigla
1	banco de dados para web	32,20	10/01/1999	1	BAN
2	programando em linguagem c	30,00	01/10/1997	1	PRO
3	progrmando em linguagem c++	115,50	01/11/1998	3	PRO
4	banco de dados na bioinformática	48,00		2	BAN
5	redes de computadores	42,00	01/09/1996	2	RED

ESCREVE	
liv_codigo	aut_matricula
1	1
2	1
3	2
4	3
5	4

ASSUNTO	
ass_sigla	ass_descricao
BAN	Banco de Dados
PRO	Programação
RED	Redes
SIS	Sistemas Operacionais

EDITORIA	
edi_codigo	edi_nome
1	Mirandela
2	Editora Via Norte
3	Editora Ilhas Tijucas
4	Maria José

AUTOR		
aut_matricula	aut_nome	...
1	Luiz	
2	Hugo	
3	Joaquim	
4	Regina	

2. Selecionar o preço do livro cujo preço seja maior que 50 e a sigla seja BAN
3. Selecionar o preço do livro cujo preço seja maior que 50 ou o assunto comece com a letra "P"
4. Selecionar os livros (todos os campos) cujo lançamento seja Nulo.
5. Selecionar os títulos do livro cujo título comece com Banco.
6. Selecionar os livros (todos os campos) cujo preço esteja entre 10 e 60
7. Selecionar os livros (todos os campos) cuja sigla seja BAN e PRO

8. Excluir o livro cujo título é Banco de Dados Distribuído ou Banco de Dados para WEB. Somente estas 2 opções devem ser consideradas;
9. Excluir da tabela de livros aqueles que possuem o código maior ou igual a 2, que possuem preço maior que R\$ 50,00;
10. Atualize para zero o preço de todos os livros onde a data de lançamento for nula ou onde seu preço atual for inferior a R\$ 5,00.
11. Excluir todos os livros onde o assunto for diferente de 'BAN' ou 'PRO'
12. Após popular a tabela livro, criar uma tabela de "backup".

Referências Bibliográficas

- [1] Fanderuff, Damaris. Dominando o Oracle 9i: Modelagem e desenvolvimento. São Paulo: Pearson Education do Brasil, 2003.
- [2] Costa, Rogério Luis de C., SQL : guia prático. 2. ed. Rio de Janeiro : Brasport, 2006.
- [3] SILBERSCHATZ, A. Sistema de bancos de dados. São Paulo: Pearson Education do Brasil, 2004.
- [4] Morelli, Eduardo M. Terra, 1996. Oracle 9i Fundamental: Sql, Pl/SQL e Administração. São Paulo: Érica, 2002.