

Objetivos da Aula:

- SQL – Select

27-01-2014

Roteiro

- Preparação do ambiente;
- SQL - Structured Query Language
- Recursos da Instrução Select;
- Instruções SQL Básicas;
- Expressões Aritméticas;
- Utilização da Clausula where;
- Exibindo dados de várias tabelas;
- Produto Cartesiano;
- Junções;
- Comparação da sintaxe Oracle com a sintaxe ANSI;

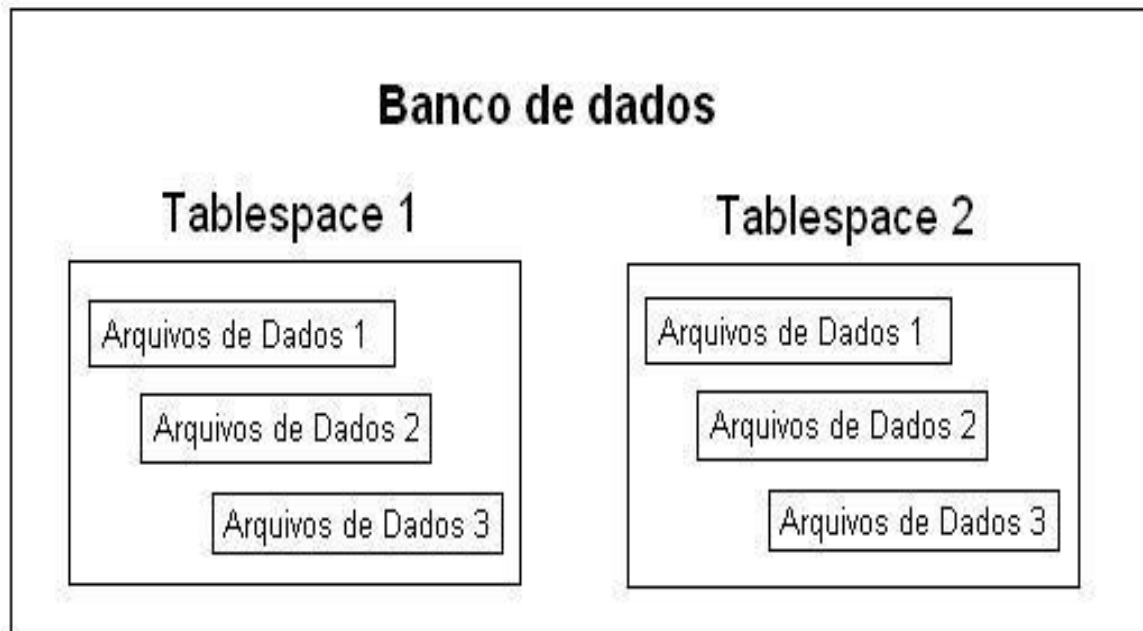



PREPARAÇÃO DO AMBIENTE

Introdução ao SGBD Oracle

Introdução: Estrutura de armazenamento do Oracle

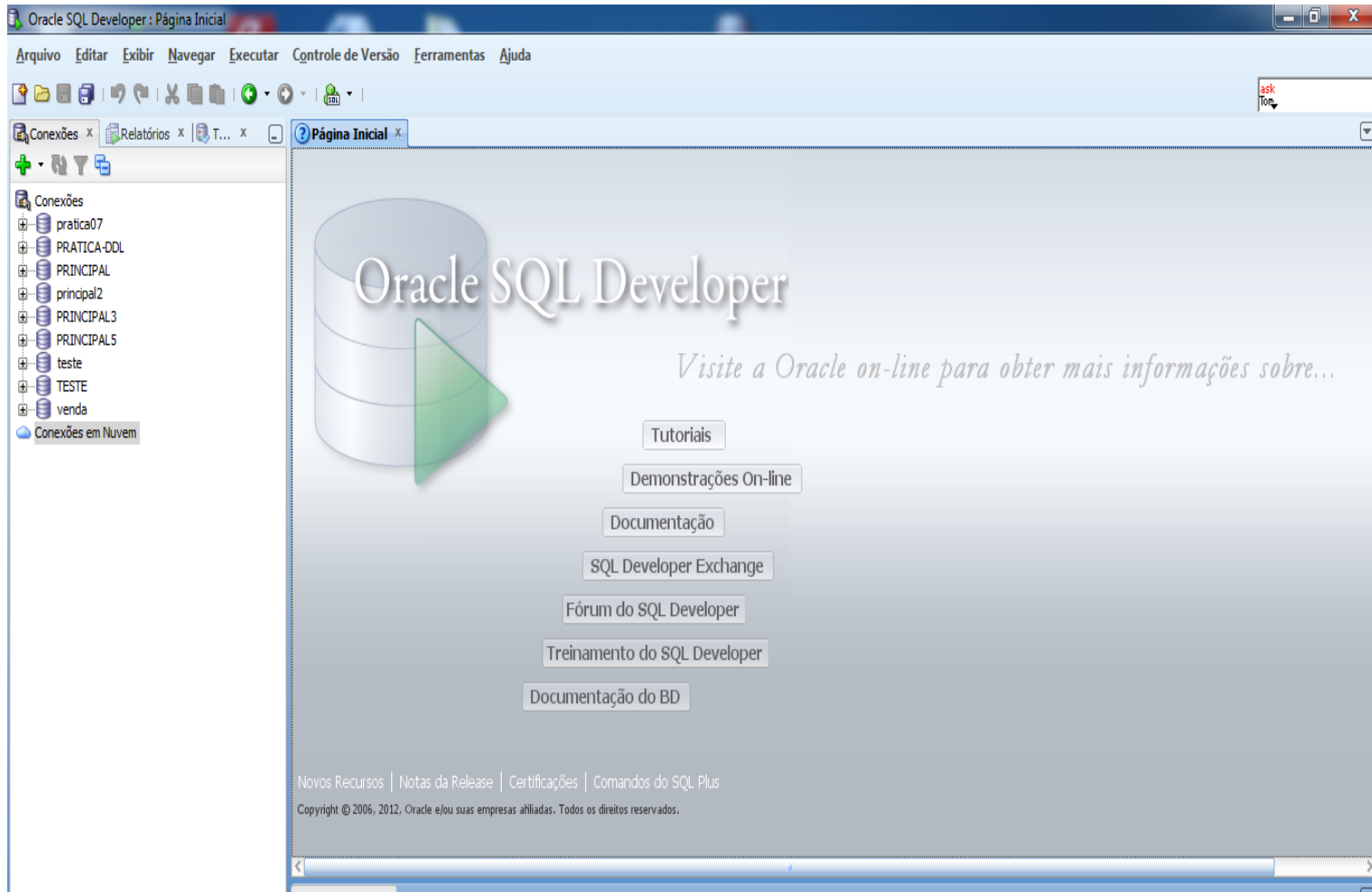
- O Oracle aloca espaço físico para o Banco de Dados, utilizando as seguintes definições, hierarquicamente organizadas:
 - ▣ **Banco de Dados:** Uma coleção lógica de dados compartilhados armazenados em *tablespaces*.
 - ▣ **Tablespace:** repositório lógico para dados fisicamente agrupados.
 - ▣ **Arquivo de Dados:** arquivo de dados físico pertencendo a uma única *tablespace*.



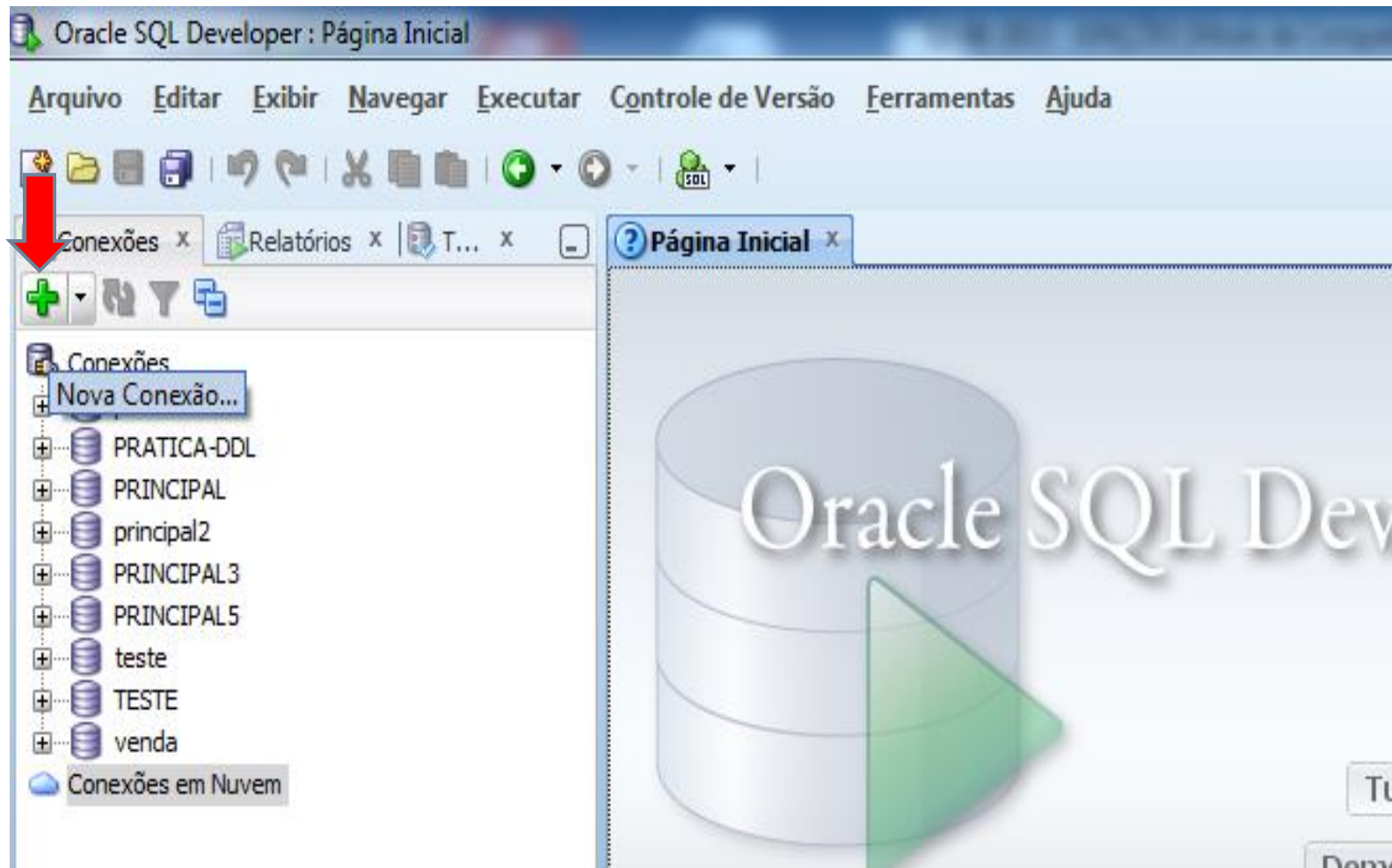


Criando uma conexão no Oracle SQLDeveloper

Abrir o Oracle SQLDeveloper



Criar nova Conexão



Conectar

Novo / Selecionar Conexão do Banco de Dados

Nome da Conexão	Detalhes da Cone...
PRATICA-DDL	USER_PRATICAD...
pratica07	USER_PRATICA0...
PRINCIPAL	SYSTEM@//localh...
principal2	system@//localho...
PRINCIPAL3	system@//localho...
PRINCIPAL5	system@//localho...
TESTE	SYSTEM@//localh...
teste	system@//localho...
venda	user_venda@//lo...

Nome da Conexão: PRINCIPAL

Nome do Usuário: SYSTEM

Senha:

☐ Salvar Senha

Oracle Access

Tipo de Conexão: Básico Atribuição: default

Nome do Host: localhost

Porta: 1521

☒ SID: xe

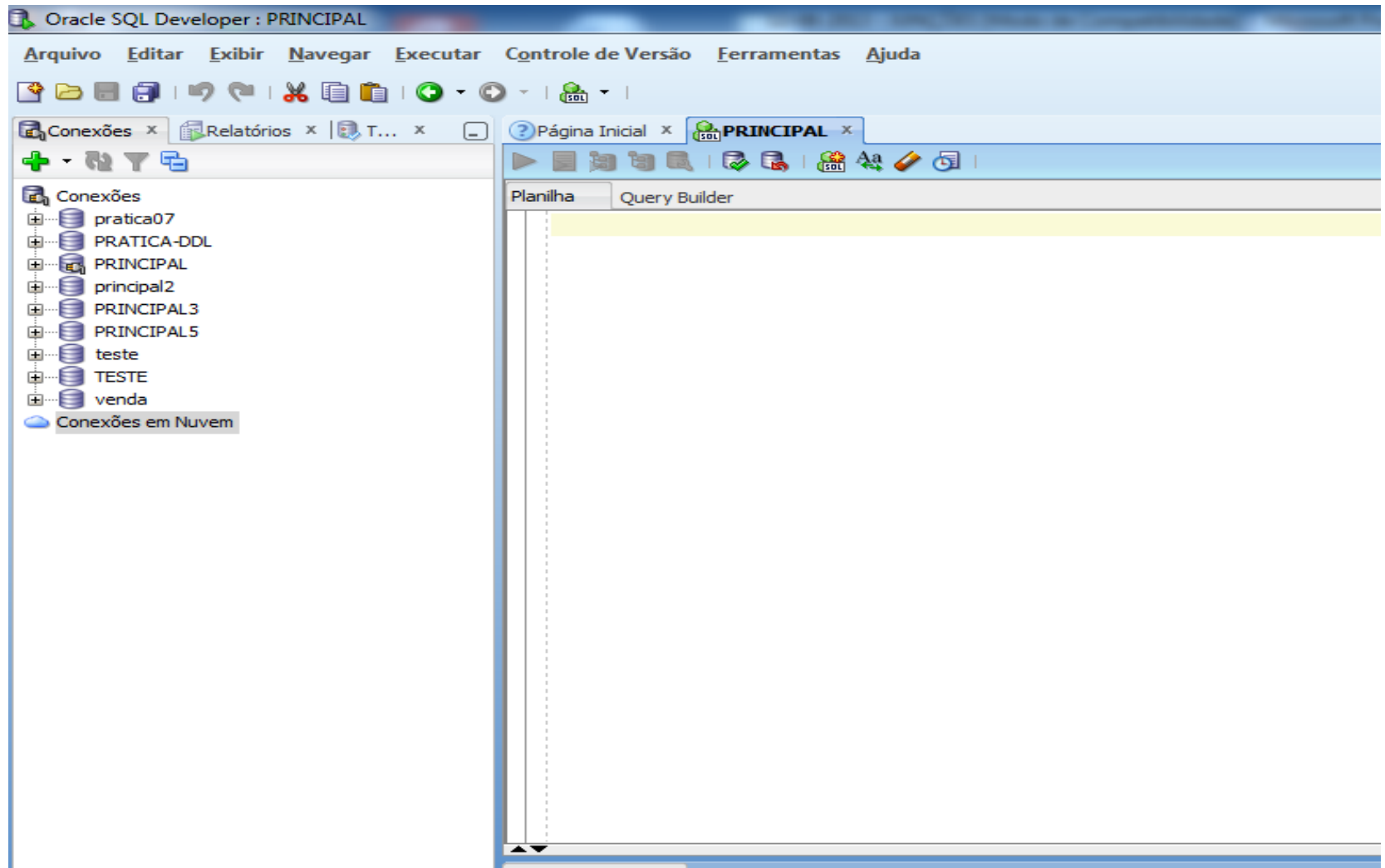
☐ Nome do Serviço

☐ Autenticação de SO ☐ Autenticação Kerberos ☐ Conexão com Proxy

Status : Com Sucesso

Ajuda Salvar Limpar Testar Conectar Cancelar

Conexão Criada



Criando uma tablespace no Oracle

- Abrir no developer o arquivo
- 1. Cria_tablespace_usuario

OBS: CRIAR A PASTA BD NO DIRETÓRIO C

Criando uma tablespace no Oracle

```
CREATE TABLESPACE TS_DB_2014
DATAFILE 'C:\BD\ts_db_2014.dbf' SIZE 1M
AUTOEXTEND ON;
```

```
-----
-- CRIA USUÁRIO --
-----
```

```
CREATE USER USER_VENDA
IDENTIFIED BY aluno
DEFAULT TABLESPACE ts_db_2014
TEMPORARY TABLESPACE TEMP
QUOTA UNLIMITED ON ts_db_2014;
```

```
-----
-- PRIVILÉGIOS --
-----
```

```
GRANT DBA TO USER_VENDA WITH ADMIN OPTION;
```

Criar a Conexão Venda

Clicar em nova conexão. Usuário User_Venda e senha aluno

Novo / Selecionar Conexão do Banco de Dados

Nome da Conexão	Detalhes da Cone...
PRATICA-DDL	USER_PRATICAD...
pratica07	USER_PRATICA0...
PRINCIPAL	SYSTEM@//localh...
principal2	system@//localho...
PRINCIPAL3	system@//localho...
PRINCIPAL5	system@//localho...
TESTE	SYSTEM@//localh...
teste	system@//localho...

Nome da Conexão: VENDA

Nome do Usuário: USER_VENDA

Senha:

☐ Salvar Senha

Oracle Access

Tipo de Conexão: Básico Atribuição: default

Nome do Host: localhost

Porta: 1521

☒ SID: xe

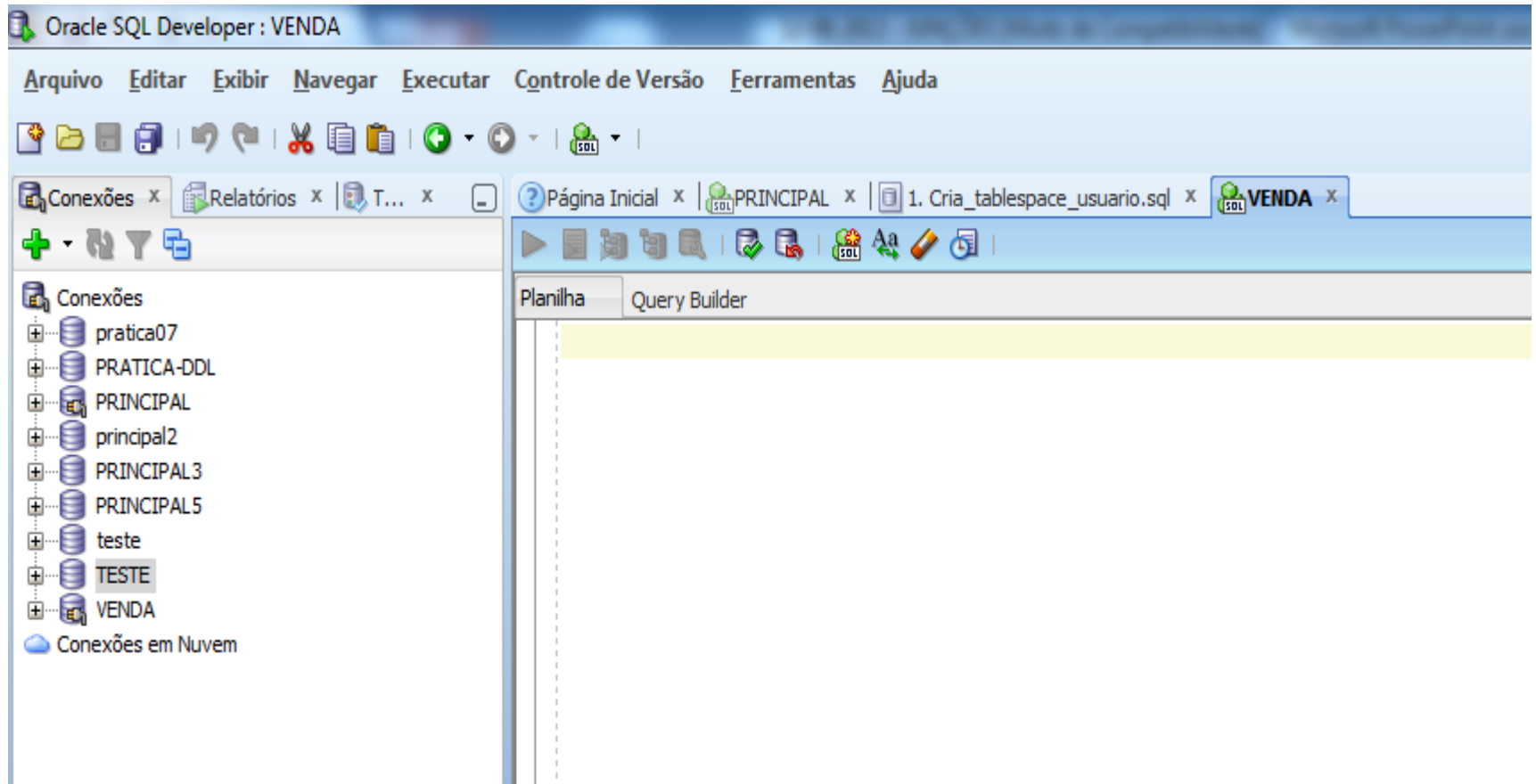
☐ Nome do Serviço


☐ Autenticação de SO ☐ Autenticação Kerberos ☐ Conexão com Proxy

Status : Com Sucesso

Ajuda Salvar Limpar Testar Conectar Cancelar

Conexão Venda Criada



- 
- Executar os arquivos na conexão Venda:
 - VENDA-DDL;
 - VENDA-DML.

SQL

Comando	Descrição	Tipo
Select	Recupera dados de uma ou mais tabelas.	Recuperação de Dados
Insert update delete	Servem para incluir, alterar e eliminar linhas de uma tabela, respectivamente.	DML
commit rollback savepoint	Responsáveis pelo controle de transações, permitem que o usuário desfça (ROLLBACK) ou confirme alterações em tabelas.	Controle de Transações
create alter drop	Úteis para definir, alterar e remover estruturas do banco de dados	DDL
grant revoke	Permitem remover direitos de acesso dos usuários do Banco de Dados e seus componentes, ou concedê-los.	DCL

Recursos das Instrução Select

- Projeção: Listar coluna(s) de uma tabela.
- Seleção: Escolhe as linhas de uma tabela.
- Junção: Reuni dados de uma ou mais tabela.

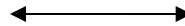
Recursos das Instrução Select

Projeção

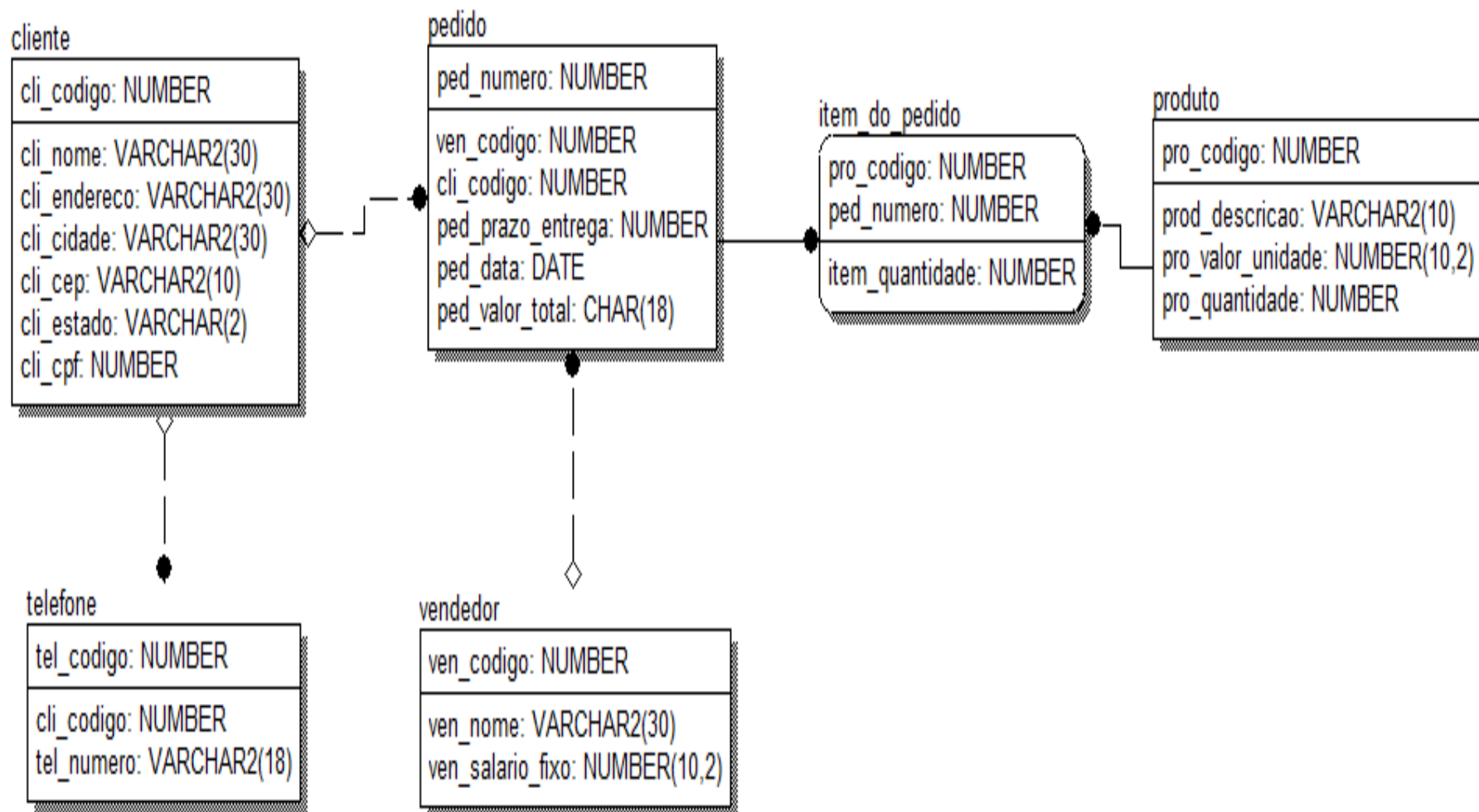
Seleção

Recursos das Instrução Select

Junção



ESTUDO DE Caso Venda



Instruções SELECT

```
SELECT * {coluna/expressão [apelido],....}  
FROM TABELA;
```

- Select identifica quais colunas
- FROM identifica qual tabela

Expressões Aritméticas

Operadores: +

-

*

/

```
Select emp_nome, 12*emp_salário + 100  
From empregado
```

Listar os produtos.

Select * from produto

PRO_CODIGO	PROD_DESCRICAO	PRO_QUANTIDADE	PRO_VALOR_UNIDADE
1	CANETA	1	100
2	APONTADOR	4	400
3	REGUA	5	600

Listar a descrição do produto, o valor e o seu valor acrescido de 10%.

```
select prod_descricao,  
       pro_valor_unidade,  
       pro_valor_unidade * 1.10 Valor  
from produto
```

	PROD_DESCRICAO	PRO_VALOR_UNIDADE	VALOR
1	CANETA	100	110
2	APONTADOR	400	440
3	REGUA	600	660

Restringindo e Classificando Dados

Usando a Cláusula Where

```
Select emp_id, emp_nome  
From empregado  
Where emp_id=100;
```

Condições de Comparação

Operadores:

=

>

>=

<

<=

<>

Exemplo

```
Select emp_id, emp_nome, emp_salário  
From empregado  
Where emp_salário > 1000
```

Listar os pedidos que foram realizados em '12/04/11'

```
select * from pedido  
where ped_data='12/04/11'
```

PED_NUMERO	VEN_CODIGO	CLI_CODIGO	PED_PRAZO_ENTREGA	PED_DATA	PED_VALOR_TOTAL
9	7	1	20	12/04/11	
10	8	2	30	12/04/11	

Exibindo Dados de Várias Tabelas

Produtos Cartesianos

Gera-se um produto cartesiano quando:

- Uma condição de junção for omitida
- Uma condição de junção for inválida

Para evitar um produto cartesiano, sempre inclua uma condição de junção válida em uma **cláusula where.**

Tabela Cliente

CLI_CODIGO	CLI_NOME	CLI_ENDERECO	CLI_CIDADE	CLI_CEP	CLI_ESTADO	CLI_CPF
1	Ana	Rua 17 n.19	Niterói	24358310	RJ	11111111111
2	Flávio	Áv. Pres. Vargas 10	São Paulo	22763931	SP	2253412693
3	Jorge	Rua Caiapo 13	Curitiba	30078500	PR	14512798349
4	Lúcia	Rua Itabira 123 Loja 9	Belo Horizonte	22124391	MG	2831521393
5	Maurício	Av. Paulista 1236 sl/2345	São Paulo	3012683	SP	3281698574
7	Rodolfo	Largo da Lapa 27 sobrado	Rio de Janeiro	30078900	RJ	1283512823
8	Beth	Av.Climério n. 45	São Paulo	25679300	SP	3248512673
9	Paulo	TV. Moraes c/3	Londrina	0	PR	3284822332
10	Lívio	Av. Beira Mar n. 1256	Florianópolis	300077500	SC	12736571
11	Susana	Rua Lopes Mendes 12	Niterói	30046500	RJ	2176357123
12	Renato	Rua Meireles n.123 sl.345	São Paulo	30225900	SP	1327657112
13	Sebastião	Rua da Igreja n.10	Uberaba	30438700	MG	3217654721
14	José	Quadra 3 bl. 3 sl. 1003	Brasília	22841650	DF	21763576123

Total de 13 clientes

Tabela Pedido

PED_NUMERO	VEN_CODIGO	CLI_CODIGO	PED_PRAZO_ENTREGA	PED_DATA	PED_VALOR_TOTAL
1	7	1	20	10/06/12	(null)
2	1	7	20	31/07/12	(null)
3	1	7	15	01/06/03	(null)
4	1	5	20	01/07/03	(null)
5	1	7	20	01/08/11	(null)
6	2	7	15	31/05/03	(null)
7	3	7	30	30/06/11	(null)
8	5	9	30	01/08/11	(null)
9	7	1	20	12/04/11	(null)
10	8	2	30	12/04/11	(null)

Total de 10 pedidos

Gerando um Produto Cartesiano

Listar os pedidos realizados: Nome do cliente e Data do Pedido

```
select cli_nome,ped_data  
from pedido, cliente
```

Flávio	12/04/11
Jorge	12/04/11
Lúcia	12/04/11
Maurício	12/04/11
Rodolfo	12/04/11
Beth	12/04/11
Paulo	12/04/11
Lívio	12/04/11
Susana	12/04/11
Renato	12/04/11
Sebastião	12/04/11
José	12/04/11
Ana	12/04/11
Flávio	12/04/11
Jorge	12/04/11
Lúcia	12/04/11
Maurício	12/04/11
Rodolfo	12/04/11
Beth	12/04/11
Paulo	12/04/11
Lívio	12/04/11
Susana	12/04/11
Renato	12/04/11
Sebastião	12/04/11
José	12/04/11

130 linhas selecionadas

$10 \times 13 = 130$
Produto Cartesiano.

Junções - Join

Utilizado para obter dados de Várias tabelas.

Estabelece o critério de relacionamento entre as tabelas.

- Junções Idênticas (Equi-Join)
- Junções Externas (OuterJoin)
- Junções Não-Idênticas (No Equijoin)
 - Auto Junções

Junções Idênticas (Equi-Join) (Junções simples/ Junções internas)

Listar os pedidos realizados. Código do cliente e Data do Pedido

```
select c.cli_nome Nome,p.ped_data Data from  
pedido p , cliente c  
where c.cli_codigo=p.cli_codigo
```

CLI_NOME	PED_DATA
Ana	12/04/11
Ana	10/06/12
Flávio	12/04/11
Maurício	01/07/03
Rodolfo	30/06/11
Rodolfo	31/05/03
Rodolfo	01/08/11
Rodolfo	01/06/03
Rodolfo	31/07/12
Paulo	01/08/11

10 linhas selecionadas

Listar todos os telefones.

	AZ	TEL_CODIGO	AZ	CLI_CODIGO	AZ	TEL_NUMERO
1		1		1		3923-1546
2		2		2		97858999
3		3		2		81267270
4		4		3		82567896
5		5		8		87589658

Listar os nomes e os telefones dos clientes.

```
select c.cli_codigo,c.cli_nome,t.tel_numero
from cliente c,
     telefone t
where c.cli_codigo=t.cli_codigo
```

CLI_CODIGO	CLI_NOME	TEL_NUMERO
-----	-----	-----
1	Ana	3923-1546
2	Flávio	81267270
2	Flávio	97858999
3	Jorge	82567896
8	Beth	87589658

A tabela cliente possui quantos clientes?

Junções Externas (OuterJoin)

Listar os nomes e os telefones dos clientes.

```
select c.cli_codigo,c.cli_nome,t.tel_numero  
from cliente c,  
     telefone t
```

```
where c.cli_codigo=t.cli_codigo(+) ← Sintaxe Oracle
```

CLI_CODIGO	CLI_NOME	TEL_NUMERO
1	Ana	3923-1546
2	Flávio	97858999
2	Flávio	81267270
3	Jorge	82567896
8	Beth	87589658
5	Maurício	
12	Renato	
10	Lívio	
7	Rodolfo	
4	Lúcia	
14	José	
9	Paulo	
13	Sebastião	
11	Susana	

14 linhas selecionadas

```
select c.cli_codigo, c.cli_nome,t.tel_numero
from cliente c left outer join telefone t
on (c.cli_codigo=t.cli_codigo)
```

CLI_CODIGO	CLI_NOME	TEL_NUMERO
1	Ana	3923-1546
2	Flávio	97858999
2	Flávio	81267270
3	Jorge	82567896
8	Beth	87589658
5	Maurício	
12	Renato	
10	Lívio	
7	Rodolfo	
4	Lúcia	
14	José	
9	Paulo	
13	Sebastião	
11	Susana	

14 linhas selecionadas

```
select c.cli_codigo, c.cli_nome,t.tel_numero
from telefone t right outer join cliente c
on (c.cli_codigo=t.cli_codigo)
```

CLI_CODIGO	CLI_NOME	TEL_NUMERO
1	Ana	3923-1546
2	Flávio	97858999
2	Flávio	81267270
3	Jorge	82567896
8	Beth	87589658
5	Maurício	
12	Renato	
10	Lívio	
7	Rodolfo	
4	Lúcia	
14	José	
9	Paulo	
13	Sebastião	
11	Susana	

14 linhas selecionadas

Listar os pedidos (Número do Pedido e Nome do Cliente, nome do vendedor)

```
select p.ped_numero,  
       c.cli_nome,  
       v.ven_nome  
from pedido p,  
     cliente c,  
     vendedor v  
where p.cli_cod=c.cli_cod and  
       p.ven_codigo=v.ven_codigo and  
       p.ped_data='13/04/05'
```

Junções Externas (OuterJoin)

42

Funcionário

FUN_COD	FUN_NOME	DEP_ID
1	Maria	1
2	Helena	1
3	João	2
4	Ana	-

Departamento

DEP_ID	DEP_NOME
1	Venda
2	RH
3	Informática
4	Compra

Listar os funcionários que estão alocados em um departamento e também os departamentos que não possuem funcionários, conforme resultado abaixo:

FUN_COD	FUN_NOME	DEP_ID	DEP_NOME
1	Maria	1	Venda
2	Helena	1	Venda
3	João	2	RH
-	-	4	Compra
-	-	3	Informática

Listar os funcionários que estão alocados em um departamento e também os departamentos que não possuem funcionários.

43

Sintaxe Ansi	Sintaxe Oracle
Select f.fun_cod, f.fun_nome, d.dep_nome From Funcionario F Right outer Join Departamento D On (F.dep_id = D.dep_id)	Select f.fun_cod, f.fun_nome, d.dep_id, d.dep_nome From Funcionario F, Departamento D Where F.dep_id (+) = D.dep_id
Select f.fun_cod, f.fun_nome, d.dep_nome From Departamento D Left outer Join Funcionario F On (F.dep_id = D.dep_id)	

Liste os funcionários que estão alocados em um departamento, também os que não estão alocados e os departamentos que não possuem funcionários alocados.

44

FUN_COD	FUN_NOME	DEP_NOME
2	Helena	Venda
1	Maria	Venda
3	João	RH
4	Ana	-
-	-	Compra
-	-	Informática

Liste os funcionários que estão alocados em um departamento, também os que não estão alocados e os departamentos que não possuem funcionários alocados.

45

```
Select f.fun_cod,  
       f.fun_nome,  
       d.dep_nome  
From Funcionario F FULL outer Join  
     Departamento D  
On (F.dep_id = D.dep_id)
```

Junções Não-Idênticas (No Equijoin)

46

EMPREGADO

Emp_cod	Emp_nome	Emp_sal
1	Ana	2400
2	Margarida	1500
3	Pedro	1000
4	Lia	5000

FAIXA_SALARIAL

Fa_cód	Fa_menor	Fa_maior
A	1000	2999
B	3000	5999
C	6000	9999

Exibir o nome do funcionário, seu salário e o código correspondente a sua faixa salarial.

Obtemos este resultado utilizando um outro operador que não o igual =

As tabelas empregado e Faixa_salarial não possuem relacionamento.

Comandos Create Table

47

```
SQL> ed
```

Gravou arquivo afiedt.buf

```
1 create table empregado (emp_cod number(6)
2 constraint emp_cod_fk primary key,
3 emp_nome varchar2(30),
4* emp_sal number(6,2))
SQL> /
```

Tabela criada.

```
SQL> create table faixa_salarial (fa_cod varchar2(1)
2 constraint fs_fk primary key,
3 fa_menor number(6,2),
4 fa_maior number (6,2));
```

Tabela criada.

Exibir o nome do funcionário, seu salário e o código correspondente a sua faixa salarial.

48

```
SQL> ed
```

```
Gravou arquivo afiedt.buf
```

```
1 select e.emp_nome, e.emp_sal, f.fa_cod
```

```
2 from empregado e, faixa_salarial f
```

```
3* where e.emp_sal between f.fa_menor and f.fa_maior
```

```
SQL> /
```

EMP_NOME	EMP_SAL	F
Ana	2400	A
Margarida	1500	A
Pedro	1000	A
Lia	5000	B

AutoJunções

49

```
SQL> select * from empregado;
```

EMP_COD	EMP_NOME	EMP_SAL	DEP_COD	GERENTE_COD
1	Paulo	2400	1	
3	Lia	500	1	1
2	Ana	400	1	1

AutoJunções (2)

50

```
1 select e.emp_nome Empregado, g.emp_nome Gerente
2 from empregado e, empregado g
3* where e.gerente_cod=g.emp_cod
```

SQL> /

EMPREGADO

GERENTE

Lia

Paulo

Ana

Paulo

O que são Funções de Grupo?

51

Operam em conjuntos de linhas para fornecer um resultado por grupo.

func_cod	func_nome	func_sal	dep_id
1	Paulo	4.000,00	1
2	Maria	1.500,00	2
3	João	4.000,00	1
4	Laura	5.000,00	2
5	Ana	6.000,00	3

*Qual o maior
salário dos
funcionários?*

Funções de Grupo

52

- AVG
- COUNT
- MAX
- MIN
- SUM

Sintaxe da Função de Grupo

53

```
SELECT [coluna], funcao_de_grupo(coluna),..  
FROM   tabela  
[WHERE condição]  
[GROUP BY coluna]  
[ORDER BY coluna];
```

Funções AVG e SUM

```
SELECT AVG (FUNC_SAL)  
FROM FUNCIONÁRIO  
WHERE DEP_ID =1
```

```
SELECT SUM(FUNC_SAL)  
FROM FUNCIONÁRIO  
WHERE DEP_ID=2
```

func_cod	func_nome	func_sal	dep_id
1	Paulo	4.000,00	1
2	Maria	1.500,00	2
3	João	4.000,00	1
4	Laura	5.000,00	2
5	Ana	6.000,00	3

```
SELECT AVG(FUNC_SAL), SUM(FUNC_SAL)  
FROM FUNCIONÁRIO
```

Utilizadas para dados numéricos.

Funções MIN e MAX

```
SELECT MIN (FUNC_SAL), MAX (FUNC_SAL)  
FROM FUNCIONÁRIO
```

func_cod	func_nome	func_sal	dep_id
1	Paulo	4.000,00	1
2	Maria	1.500,00	2
3	João	4.000,00	1
4	Laura	5.000,00	2
5	Ana	6.000,00	3

Funções Count

56

```
SELECT COUNT(*)  
FROM FUNCIONÁRIO  
WHERE DEP_ID=2
```

```
SELECT COUNT (DEP_ID)  
FROM FUNCIONÁRIO
```

func_cod	func_nome	func_sal	dep_id
1	Paulo	4.000,00	1
2	Maria	1.500,00	2
3	João	4.000,00	1
4	Laura	5.000,00	2
5	Ana	6.000,00	3

```
SELECT COUNT (DISTINCT DEP_ID)  
FROM FUNCIONÁRIO
```


Funções de Grupo e Valores Nulos

57

As funções de grupo ignoram valores nulos na coluna

```
SELECT AVG(FUNC_SAL)
```

```
FROM FUNCIONÁRIO
```

Resultado=4750,00

Solução:

```
SELECT AVG(NVL(FUNC_SAL,0))
```

```
FROM FUNCIONÁRIO
```

Resultado=3800,00

func_cod	func_nome	func_sal	dep_id
1	Paulo	4.000,00	1
2	Maria		2
3	João	4.000,00	1
4	Laura	5.000,00	2
5	Ana	6.000,00	3

Cláusula GROUP BY

58

- SELECT dep_id, avg(func_sal)
- FROM FUNCIONÁRIO
- GROUP BY dep_id

func_cod	func_nome	func_sal	dep_id
1	Paulo	4.000,00	1
2	Maria	1.500,00	2
3	João	4.000,00	1
4	Laura	5.000,00	2
5	Ana	6.000,00	3

Resultado:

dep_id	func_sal
1	4.000,00
2	3.250,00
3	6000,00

HAVING

59

- Exibir apenas os salários dos departamentos maiores que 4000.

Como utilizar o having?