

# Objetivos da Aula:

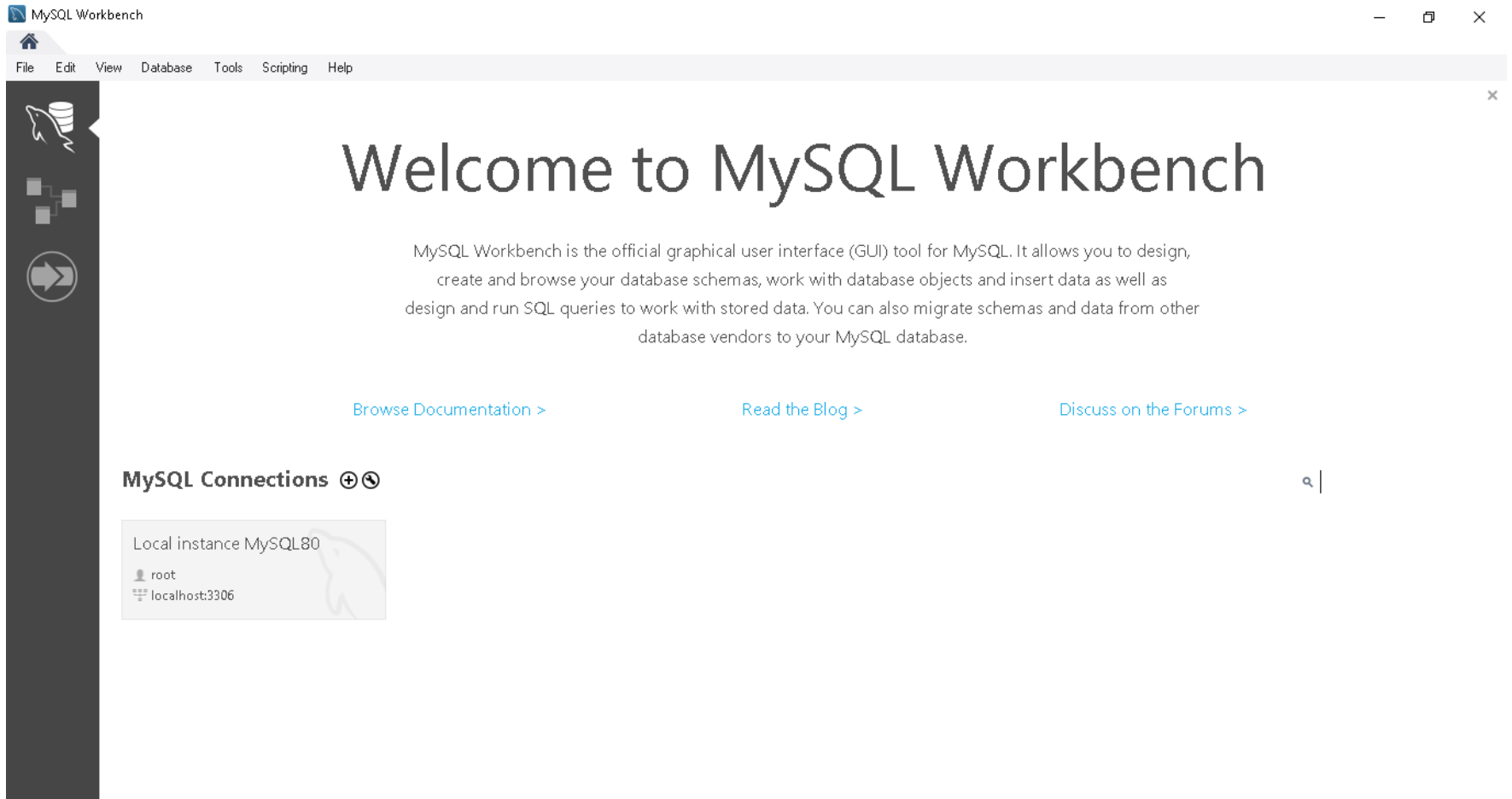
- SQL – Select

**Junções**  
**Prof. Juliana**  
30/8/2024

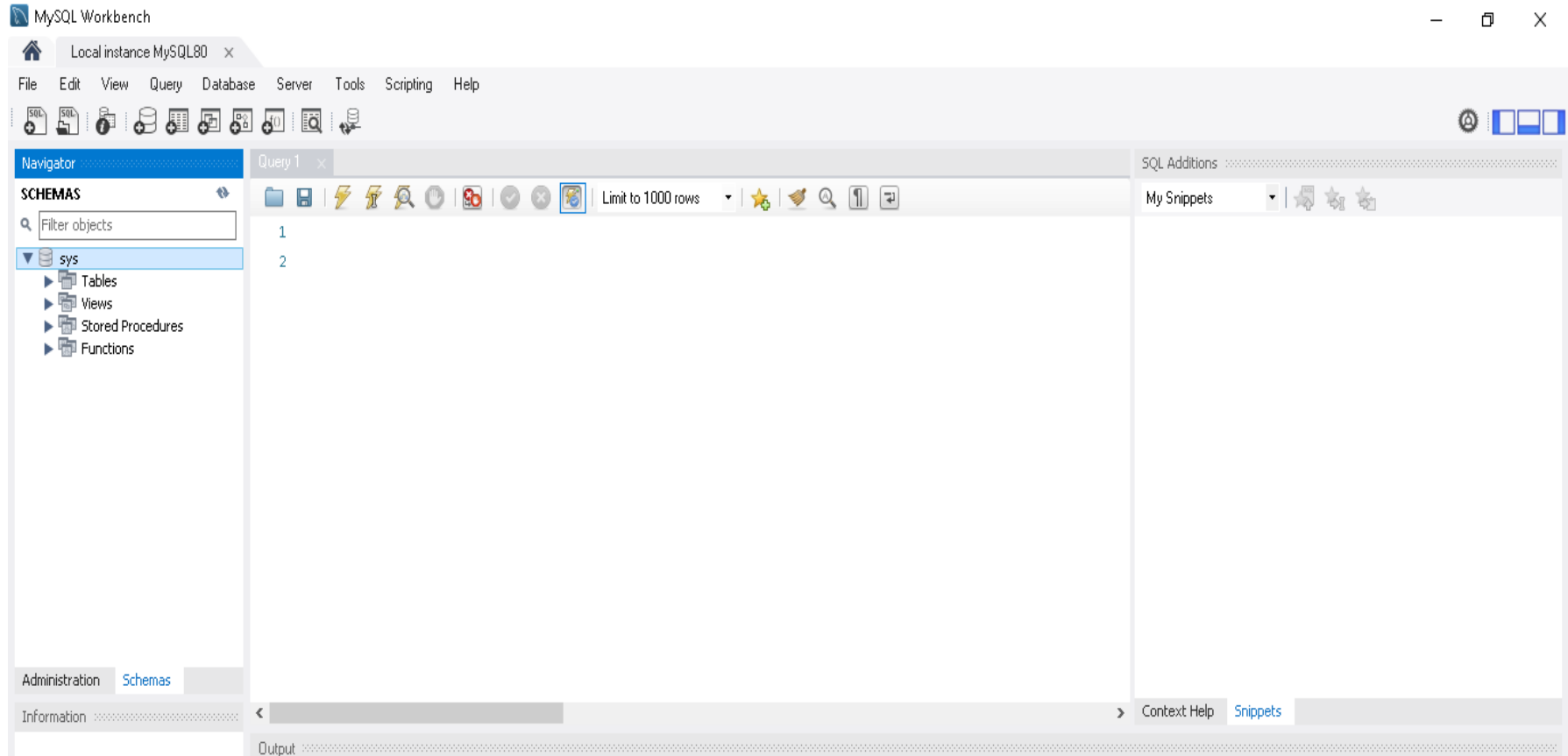
# Roteiro

- 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;
- Funções de Grupo.

# Preparação do ambiente MySQL Workbench



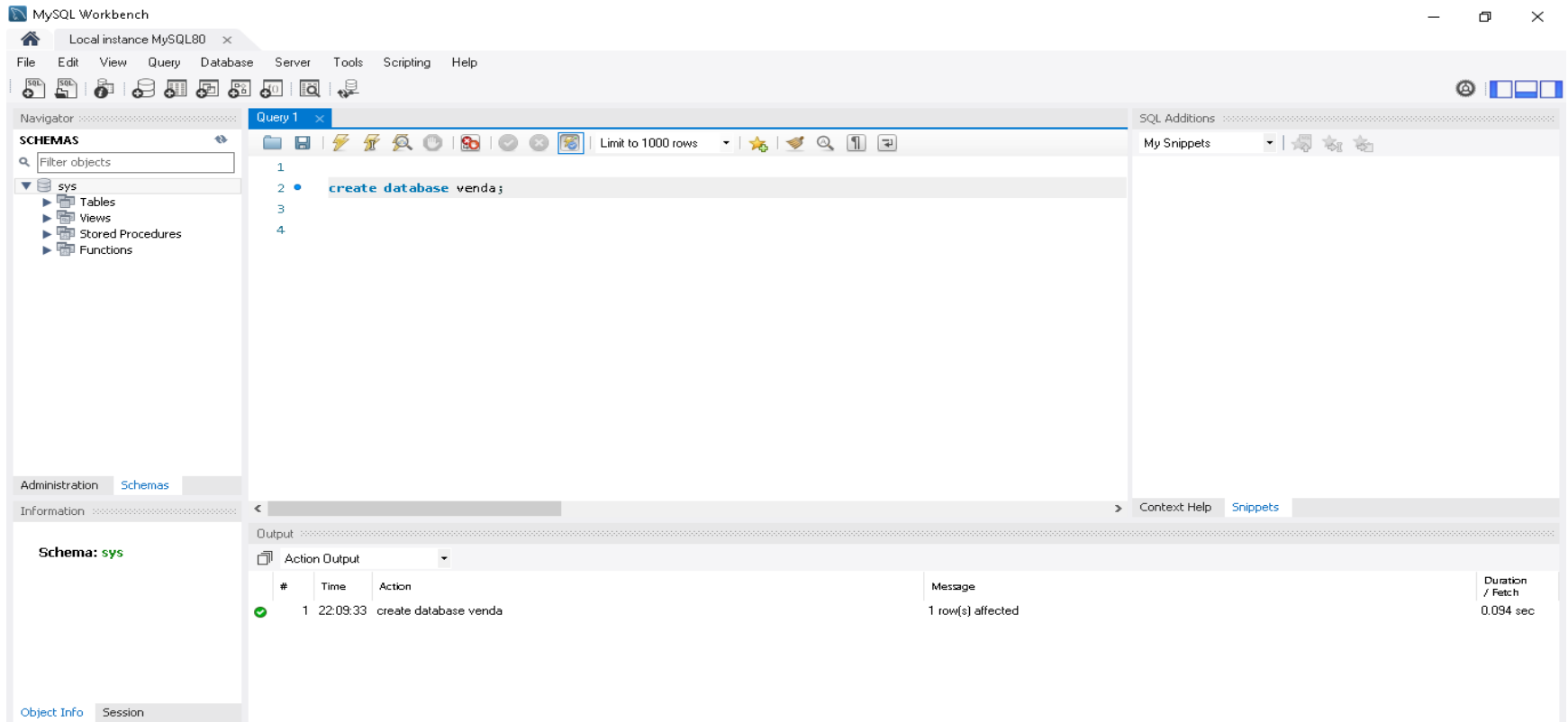
# Preparação do ambiente MySQL Workbench (2)



# Preparação do ambiente

## Criando um Banco de Dados

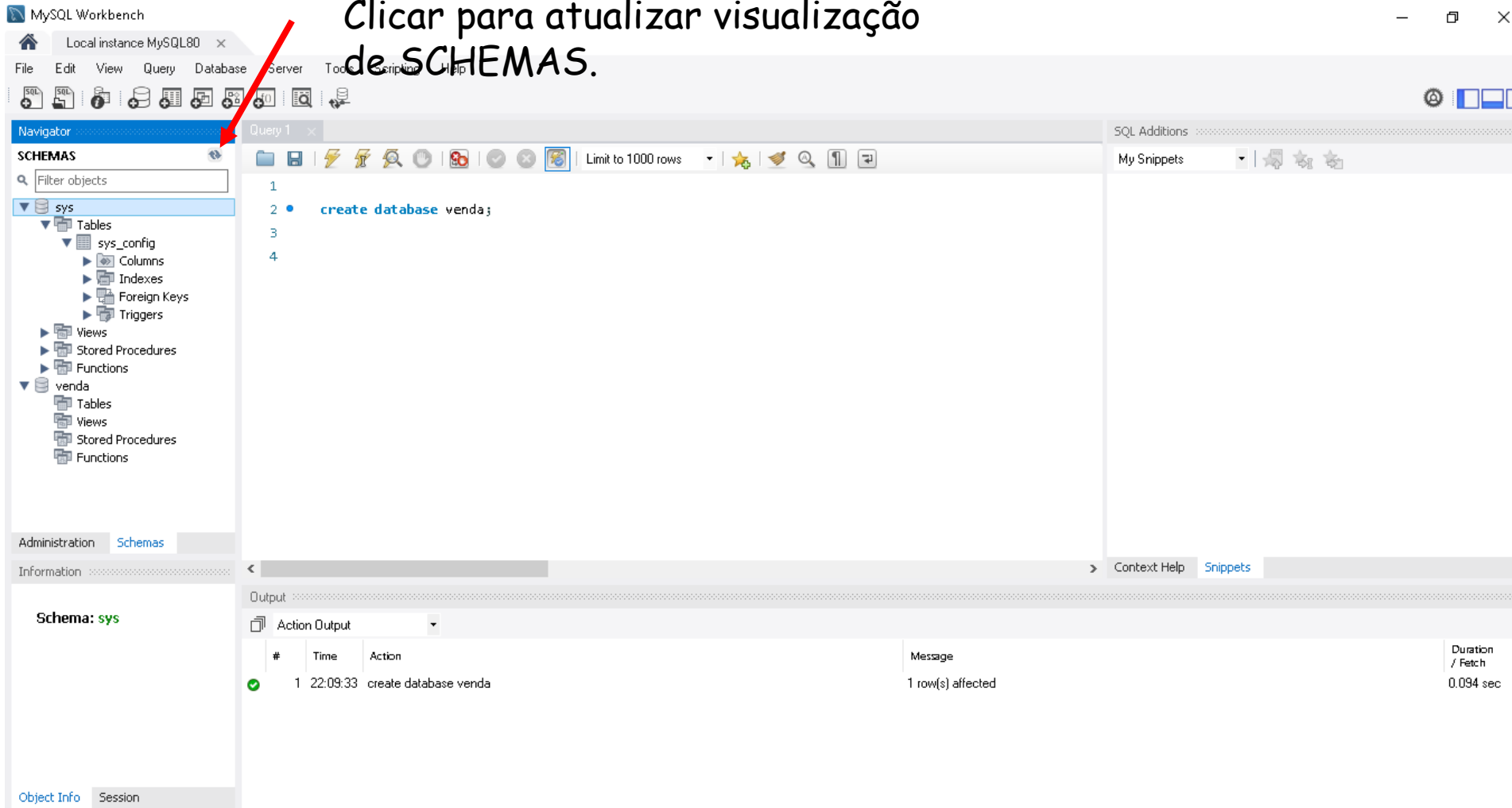
**Sintaxe:** create database <nome do Banco de dados>;



# Preparação do ambiente

## Visualizando Schemas

Clicar para atualizar visualização de SCHEMAS.



The screenshot shows the MySQL Workbench interface. The Navigator pane on the left displays the database structure. A red arrow points to the refresh icon (a circular arrow) located next to the 'sys' database entry in the Schemas section. The main query editor shows a query to create a database named 'venda'. The Output pane at the bottom shows the execution results.

**Navigator - Schemas**


- Filter objects
- sys
  - Tables
    - sys\_config
      - Columns
      - Indexes
      - Foreign Keys
      - Triggers
    - Views
    - Stored Procedures
    - Functions
  - venda
    - Tables
    - Views
    - Stored Procedures
    - Functions

**Query 1**

```
1  
2 • create database venda;  
3  
4
```

**Output**

#	Time	Action	Message	Duration / Fetch
✓ 1	22:09:33	create database venda	1 row(s) affected	0.094 sec

- 
- Executar os arquivos na schema 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



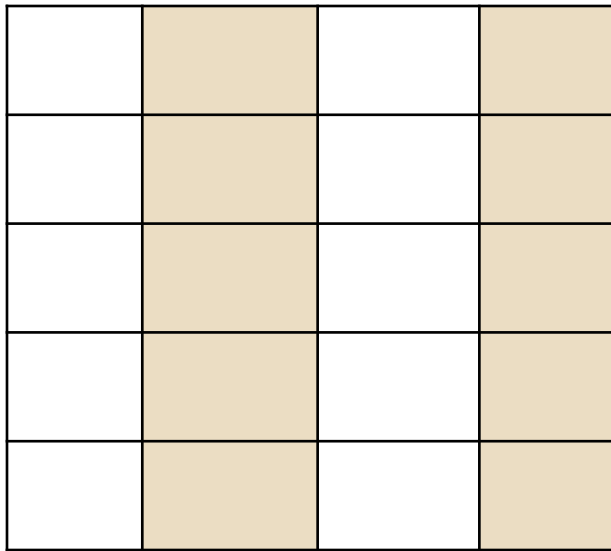
# Recursos da Instrução Select

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

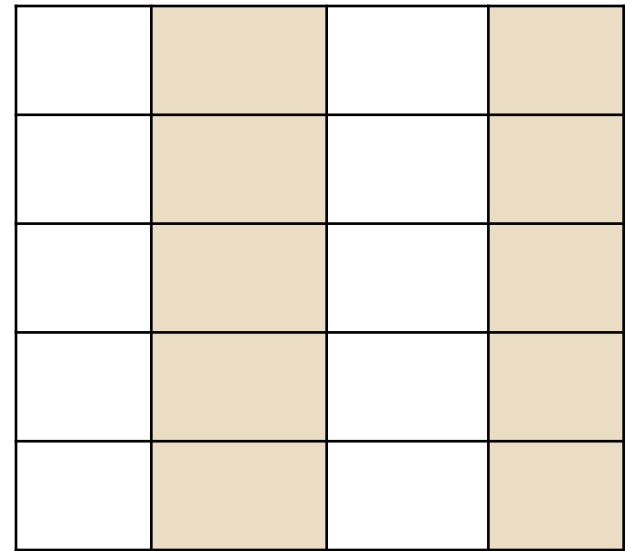
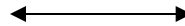
# Recursos da Instrução Select

Projeção

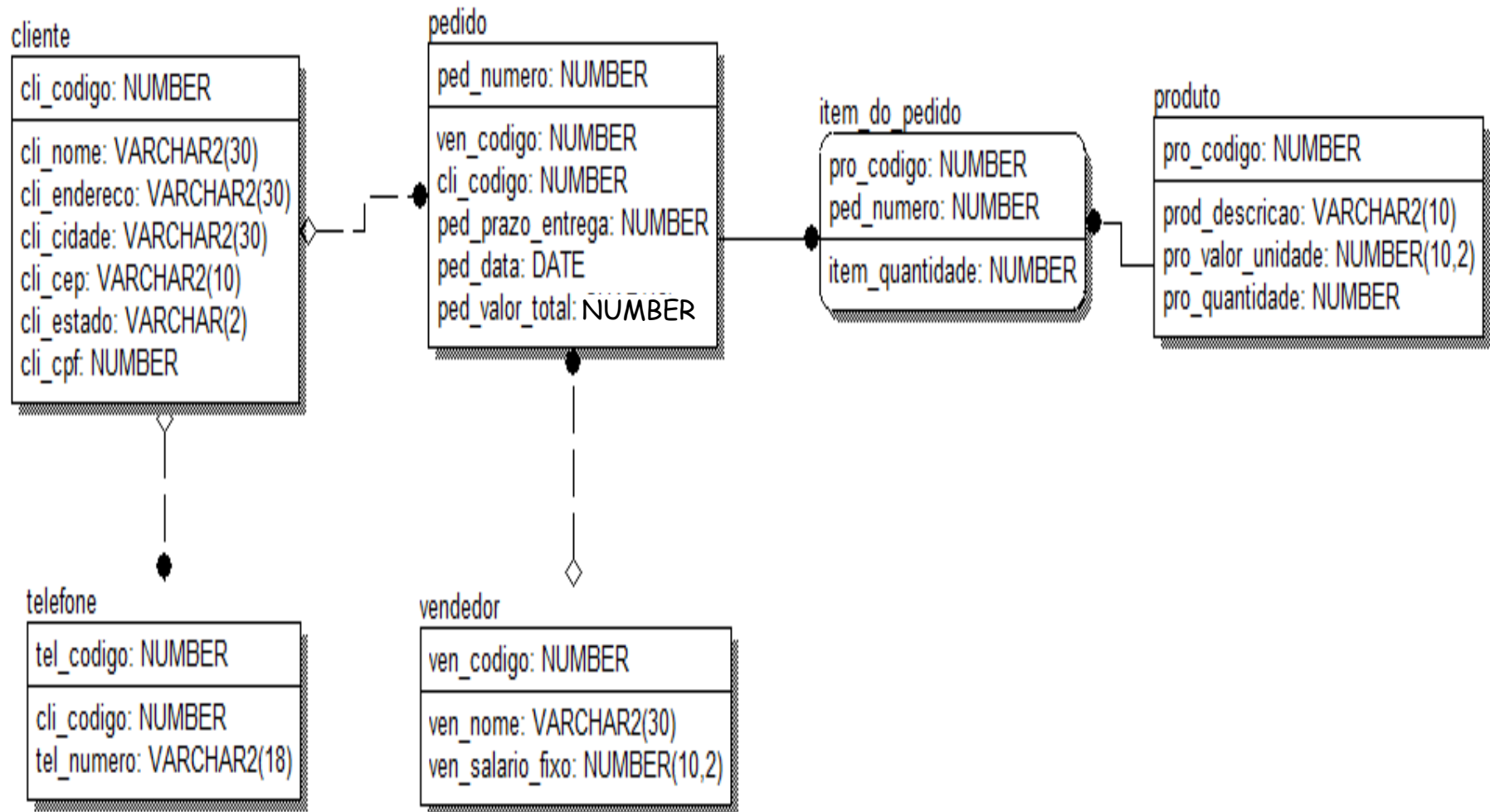

Seleção

# 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.

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

# Listar os produtos.

Select \* from produto;

	pro_codigo	prod_descricao	pro_quantidade	pro_valor_unidade
▶	1	CANETA	1	100.00
	2	APONTADOR	4	400.00
	3	REGUA	5	600.00
⊞	NULL	NULL	NULL	NULL



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

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

## 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
▶	CANETA	100.00	110.0000
	APONTADOR	400.00	440.0000
	REGUA	600.00	660.0000

# **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 '2000-10-07'

```
select * from pedido
```

```
where ped_data= '2000-10-07'
```

Result Grid		Filter Rows:		Edit:		Export/Import:	
	ped_numero	ven_codigo	cli_codigo	ped_prazo_entrega	ped_data	ped_valor_total	
▶	1	7	1	20	2000-10-07	NULL	
•	NULL	NULL	NULL	NULL	NULL	NULL	

**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


ult Grid




Filter Rows:


Edit:








Export/Import:





Wrap Cell Conte

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
6	Rodolfo	Largo da Lapa 27 sobrado	Rio de Janeiro	30078900	RJ	1283512823
7	Beth	Av.Climério n. 45	São Paulo	25679300	SP	3248512673
8	Paulo	TV. Moraes c/3	Londrina	0	PR	3284822332
9	Lívio	Av. Beira Mar n. 1256	Florianópolis	300077500	SC	12736571
10	Susana	Rua Lopes Mendes 12	Niterói	30046500	RJ	2176357123
11	Renato	Rua Meireles n.123 sl.345	São Paulo	30225900	SP	1327657112
12	Sebastião	Rua da Igreja n.10	Uberaba	30438700	MG	3217654721
13	José	Quadra 3 bl. 3 sl. 1003	Brasília	22841650	DF	21763576123
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Total de 13 clientes

# Tabela Pedido

ped_numero	ven_codigo	cli_codigo	ped_prazo_entrega	ped_data	ped_valor_total
1	7	1	20	2000-10-07	NULL
2	1	7	20	2012-07-31	NULL
3	1	7	15	2013-06-01	NULL
4	1	5	20	2013-07-01	NULL
5	1	7	20	2013-08-01	NULL
6	2	7	15	2013-09-30	NULL
7	3	7	30	2014-06-30	NULL
8	5	9	30	2015-08-01	NULL
9	7	1	20	2019-04-12	NULL
10	8	2	30	2021-04-12	NULL
NULL	NULL	NULL	NULL	NULL	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
```

	cli_nome	ped_data
▶	Ana	2021-04-12
	Ana	2019-04-12
	Ana	2015-08-01
	Ana	2014-06-30
	Ana	2013-09-30
	Ana	2013-08-01
	Ana	2013-07-01
	Ana	2013-06-01
	Ana	2012-07-31
	Ana	2000-10-07
	Flávio	2021-04-12
	Flávio	2019-04-12
	Flávio	2015-08-01
	Flávio	2014-06-30
	Flávio	2013-09-30
	Flávio	2013-08-01
	Flávio	2013-07-01
	Flávio	2013-06-01
	Flávio	2012-07-31

$10 \times 13 = 130$   
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. Nome 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
```

Nome	Data
Ana	2000-10-07
Beth	2012-07-31
Beth	2013-06-01
Maurício	2013-07-01
Beth	2013-08-01
Beth	2013-09-30
Beth	2014-06-30
Lívio	2015-08-01
Ana	2019-04-12
Flávio	2021-04-12

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



## 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_codigo=c.cli_codigo and  
       p.ven_codigo=v.ven_codigo
```



## Listar todos os telefones.

tel_codigo	cli_codigo	tel_numero
1	1	3923-1546
2	2	97858999
3	2	81267270
4	3	82567896
5	8	87589658
NULL	NULL	NULL

# 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	97858999
2	Flávio	81267270
3	Jorge	82567896
8	Paulo	87589658

A tabela cliente possui quantos clientes?

## Junções Externas (OuterJoin)

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

	cli_codigo	cli_nome	tel_numero
	1	Ana	3923-1546
	2	Flávio	97858999
	2	Flávio	81267270
	3	Jorge	82567896
	4	Lúcia	NULL
	5	Maurício	NULL
	6	Rodolfo	NULL
	7	Beth	NULL
	8	Paulo	87589658
	9	Lívio	NULL
	10	Susana	NULL
	11	Renato	NULL
	12	Sebastião	NULL
	13	José	NULL

## Junções Externas (OuterJoin)

```
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
	4	Lúcia	NULL
	5	Maurício	NULL
	6	Rodolfo	NULL
	7	Beth	NULL
	8	Paulo	87589658
	9	Lívio	NULL
	10	Susana	NULL
	11	Renato	NULL
	12	Sebastião	NULL
	13	José	NULL

# Junções Externas (OuterJoin)

37

**Funcionário**

func_cod	func_nome	func_sal	dep_id
1	Paulo	4000.00	1
2	Maria	1500.00	2
3	João	4000.00	1
4	Laura	5000.00	2
5	Ana	6000.00	3
6	Maria	NULL	NULL
NULL	NULL	NULL	NULL

**Departamento**

	dep_id	dep_descricao
▶	1	RH
	2	Vendas
	3	Informática

**Telefone2**

	tel_codigo	func_cod	tel_numero
▶	1	1	39231546
	2	2	97858999

Executar o script funcionario.txt

## Junções Externas (OuterJoin)

38

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



	fun_cod	fun_nome	dep_descricao
▶	1	Paulo	RH
	2	Maria	Vendas
	3	João	RH
	4	Laura	Vendas
	5	Ana	Informática
	6	Maria	NULL


```
Select f.fun_cod,    f.fun_nome,    d.dep_descricao  
From Departamento d right outer Join Funcionario f  
On f.dep_id = d.dep_id;
```

```
Select f.fun_cod,    f.fun_nome,    d.dep_descricao  
From Funcionario f left outer Join Departamento d  
On f.dep_id = d.dep_id;
```

# Junções Externas (OuterJoin)

- Listar os funcionários.

Result Grid				
  Filter Rows: <input type="text"/>				
	fun_cod	fun_nome	dep_descricao	tel_codigo
1		Paulo	RH	1
2		Maria	Vendas	2
3		João	RH	NULL
4		Laura	Vendas	NULL
5		Ana	Informática	NULL
6		Maria	NULL	NULL



```
Select f.fun_cod,  
        f.fun_nome,  
        d.dep_descricao,  
        t.tel_código
```



```
From Departamento D right outer Join Funcionario F  
On F.dep_id = D.dep_id  
left outer join telefone2 t  
on t.func_cod=f.fun_cod  
order by f.fun_cod;
```



# Junções Não-Idênticas (No Equijoin)

41

**FUNCIONÁRIO**

Result Grid     Filter Rows: <input type="text"/>				
	fun_cod	fun_nome	fun_sal	dep_id
	1	Paulo	4000.00	1
	2	Maria	1500.00	2
	3	João	4000.00	1
	4	Laura	5000.00	2
	5	Ana	6000.00	3
	6	Maria	NULL	NULL

**FAIXA\_SALARIAL**

fa_cod	fa_menor	fa_maior
A	1000.00	2999.00
B	3000.00	5999.00
C	6000.00	9999.00

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

42

```
create table faixa_salarial (fa_cod varchar(1) primary key,  
                             fa_menor decimal(6,2),  
                             fa_maior decimal (6,2));
```

```
insert into faixa_salarial values ('A',1000.00,2999.00);  
insert into faixa_salarial values ('B',3000.00,5999.00);  
insert into faixa_salarial values ('c',6000.00,9999.00);
```

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




43

```
SELECT f.fun_cod,  
       f.fun_nome,  
       f.fun_sal,  
       fa.fa_cod  
FROM   funcionario f,  
       faixa_salarial fa  
where  f.fun_sal between fa.fa_menor and fa.fa_maior;
```

fun_cod	fun_nome	fun_sal	fa_cod
1	Paulo	4000.00	B
2	Maria	1500.00	A
3	João	4000.00	B
4	Laura	5000.00	B
5	Ana	6000.00	c

## AutoJunções

```
select * from funcionario
```

Result Grid					
				Filter Rows:	<input type="text"/>
		Edit:			
	fun_cod	fun_nome	fun_sal	dep_id	GERENTE_COD
▶	1	Paulo	4000.00	1	5
	2	Maria	1500.00	2	5
	3	João	4000.00	1	5
	4	Laura	5000.00	2	5
	5	Ana	6000.00	3	NULL
	6	Maria	NULL	NULL	NULL

# Alteração – Tabela Funcionário

```
ALTER TABLE FUNCIONARIO ADD GERENTE_COD int;
```

```
ALTER TABLE FUNCIONARIO ADD constraint foreign key  
(gerente_cod)REFERENCES FUNCIONARIO (FUN_COD);
```

```
update funcionario set gerente_cod=5  
where fun_cod in (1,2,3,4);
```

## AutoJunções (2)

46

```
select f.fun_nome Funcionário,  
       g.fun_nome Gerente  
from funcionario f,  
     funcionario g  
where f.gerente_cod=g.fun_cod;
```

Funcionário	Gerente
Paulo	Ana
Maria	Ana
João	Ana
Laura	Ana

# Funções de Grupo

47

- AVG
- COUNT
- MAX
- MIN
- SUM

# Sintaxe da Função de Grupo

48

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



## Funções AVG e SUM

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

fun_cod	fun_nome	fun_sal	dep_id
1	Paulo	4000.00	1
2	Maria	1500.00	2
3	João	4000.00	1
4	Laura	5000.00	2
5	Ana	6000.00	3

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

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

*Utilizadas para dados numéricos.*

## Funções MIN e MAX

```
SELECT MIN (FUN_SAL), MAX (FUN_SAL)  
FROM FUNCIONARIO;
```

	AVG(FUN_SAL)	SUM(FUN_SAL)
▶	4100.000000	20500.00

# Funções Count

51

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

fun_cod	fun_nome	fun_sal	dep_id
1	Paulo	4000.00	1
2	Maria	1500.00	2
3	João	4000.00	1
4	Laura	5000.00	2
5	Ana	6000.00	3

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

```
SELECT count(DISTINCT DEP_ID)  
FROM FUNCIONARIO;
```



## Cláusula GROUP BY

52

```
SELECT dep_id, avg(fun_sal)
FROM FUNCIONÁRIO
GROUP BY dep_id;
```

fun_cod	fun_nome	fun_sal	dep_id
1	Paulo	4000.00	1
2	Maria	1500.00	2
3	João	4000.00	1
4	Laura	5000.00	2
5	Ana	6000.00	3

Resultado:

Result Grid				 Filter Rc
	dep_id	avg(fun_sal)		
▶	1	4000.000000		
	2	3250.000000		
	3	6000.000000		

## Cláusula GROUP BY

```
SELECT dep_id,  
       round(avg(fun_sal),2)  
FROM FUNCIONARIO  
GROUP BY dep_id;
```

dep_id	round(avg(fun_sal),2)
1	4000.00
2	3250.00
3	6000.00

## HAVING

54

- Exibir apenas os departamentos e as médias salariais maiores que 3500,00.

```
SELECT dep_id,  
       round(avg(fun_sal),2)  
FROM FUNCIONARIO  
GROUP BY dep_id  
having avg(fun_sal) > 3500;
```

dep_id	round(avg(fun_sal),2)
1	4000.00
3	6000.00