

Projeto - Banco de Dados Relacional

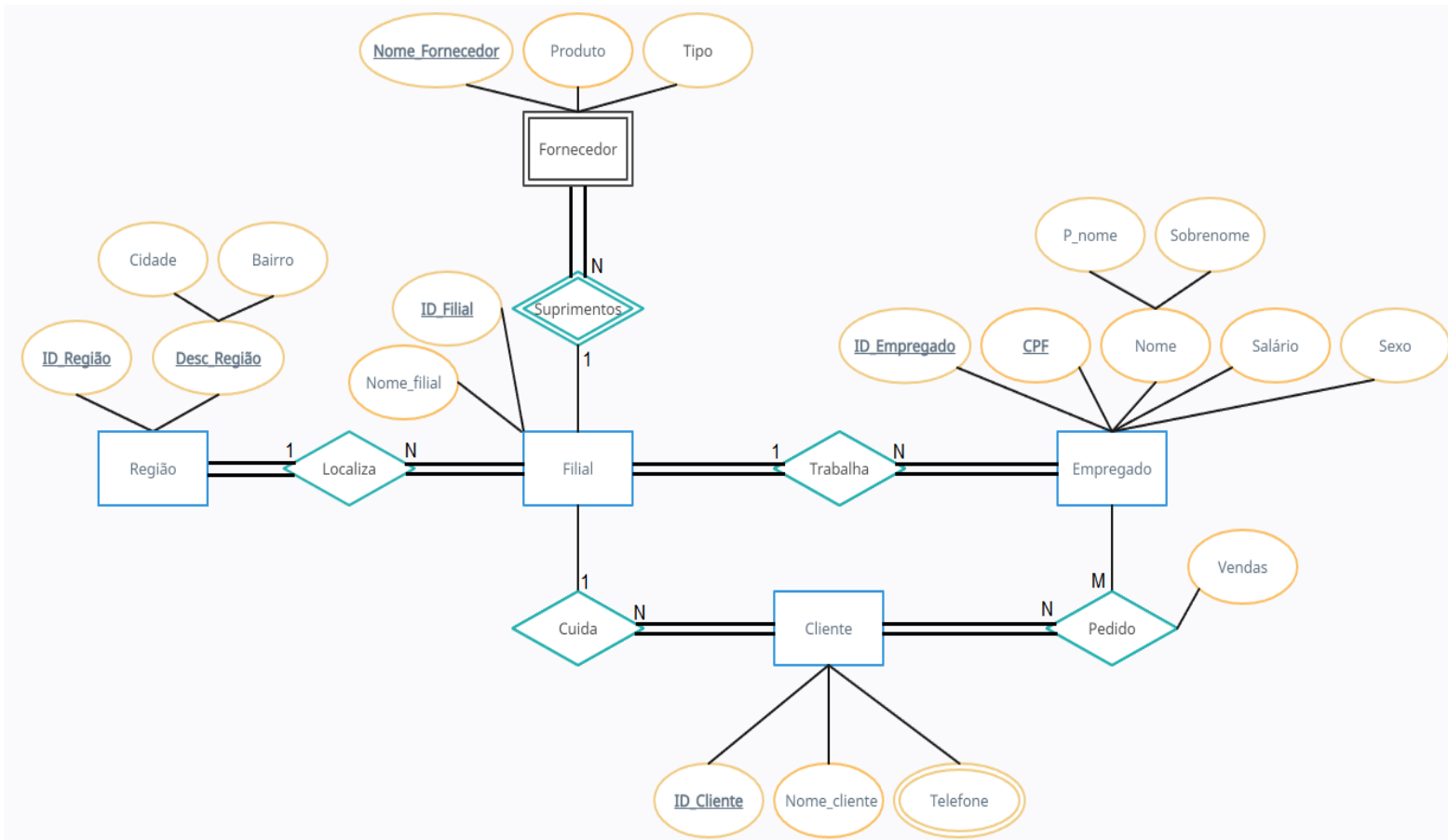
Aluno: Vitor Cunha Cavalcanti Manso

Projeto realizado utilizando o DB Schema e integrado com o Postgres

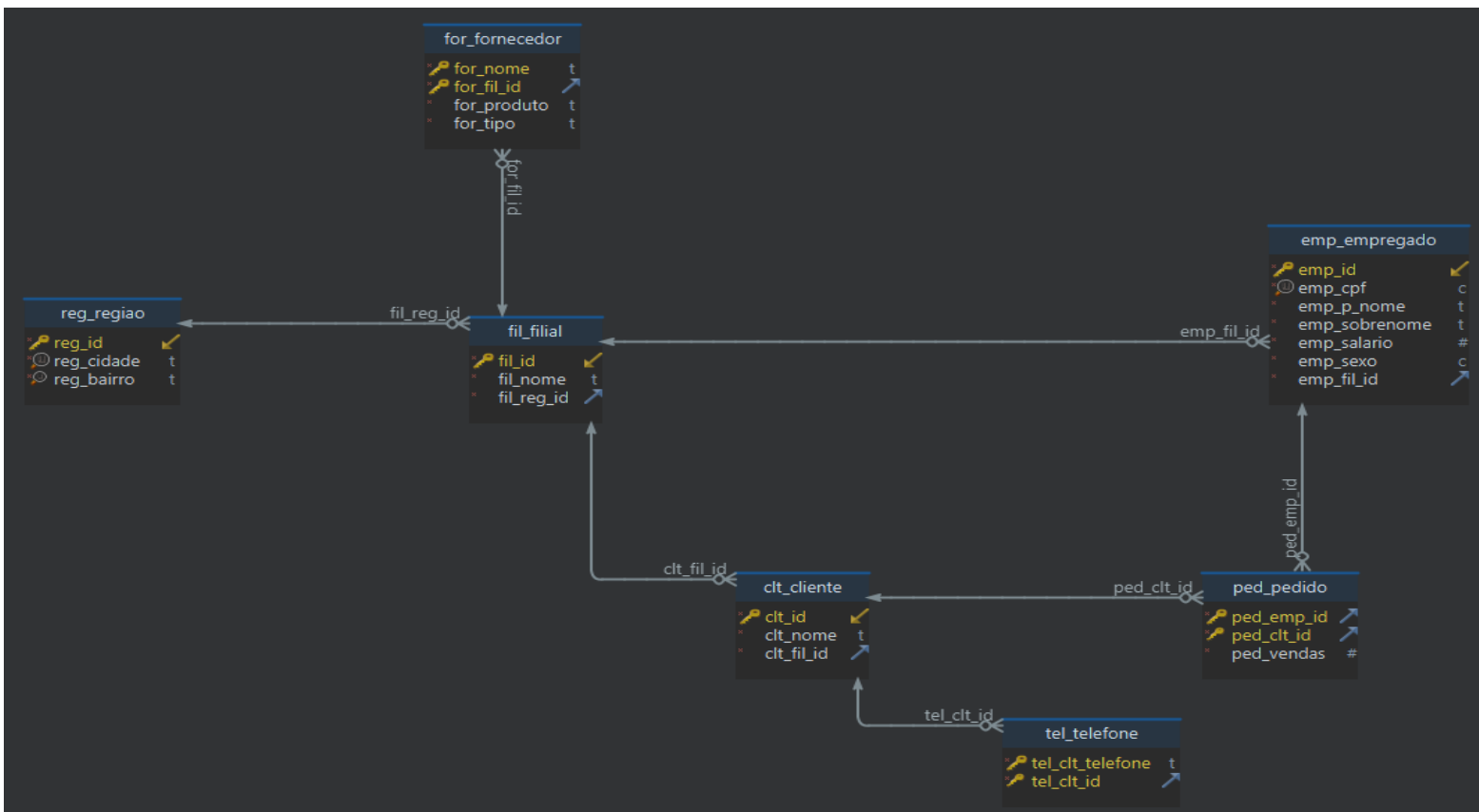
1. Requisitos para o banco de dados da empresa:

- A empresa é composta de filiais. Para cada filial, deve-se salvar o id e o nome da filial.
- A empresa ganha dinheiro vendendo para clientes. Cada cliente deve ter seu próprio id, além de nome e telefone. No caso do telefone, um cliente pode ter múltiplos números para contato.
- Para a empresa funcionar, são necessários funcionários. Cada funcionário tem o seu id e o cpf é único para cada funcionário. Além disso, deve-se salvar o nome (primeiro nome e sobrenome separadamente), salário e sexo de cada funcionário.
- Para a empresa existir, são necessárias localizações. Cada localização deve possuir um id e uma descrição única. A descrição da região é composta pelo nome da cidade e nome do bairro.
- Um funcionário pode trabalhar para apenas uma filial, e uma filial é composta de vários funcionários.
- Uma filial pode cuidar de vários clientes, mas cada cliente só pode ser cuidado por uma filial.
- Os clientes realizam seus pedidos através dos funcionários. Múltiplos funcionários podem trabalhar com vários clientes e vários clientes podem realizar pedidos com vários funcionários. Para cada pedido realizado, deve-se salvar a quantia em dinheiro que cada empregado vendeu para cada cliente, ou seja, o total da venda por pedido.
- Cada filial só pode estar localizada em apenas uma região, mas cada região pode localizar várias filiais.
- Várias filiais precisam trabalhar com fornecedores para comprar estoque de produtos. Para cada fornecedor, deve-se salvar o seu nome, além do produto e tipo de produto que ele está vendendo para a filial.
- A empresa trabalha com contratos de exclusividade com os fornecedores para cada filial, portanto, uma filial pode conseguir suprimentos de vários fornecedores, mas cada fornecedor está contratado exclusivamente com uma filial.

2. Diagrama entidade-relacionamento:



3. Esquema lógico:



4. Esquema físico – Script da criação do banco de dados:

```
CREATE SCHEMA projeto;
```

```
CREATE TABLE projeto.reg_regiao (  
    reg_id int NOT NULL,  
    reg_cidade varchar(20) NOT NULL,  
    reg_bairro varchar(20) NOT NULL,  
    CONSTRAINT pk_reg_regiao PRIMARY KEY (reg_id),  
    CONSTRAINT unq_reg_regiao UNIQUE (reg_cidade, reg_bairro)  
);
```

```
CREATE TABLE projeto.fil_filial (  
    fil_id int NOT NULL,  
    fil_nome varchar(40) NOT NULL,  
    fil_reg_id int NOT NULL,  
    CONSTRAINT pk_fil_filial PRIMARY KEY (fil_id)  
);
```

```
CREATE TABLE projeto.for_fornecedor (  
    for_nome varchar(40) NOT NULL,  
    for_fil_id int NOT NULL,  
    for_produto varchar(20) NOT NULL,  
    for_tipo varchar(20) NOT NULL,  
    CONSTRAINT pk_for_fornecedor PRIMARY KEY (for_nome, for_fil_id),  
    CONSTRAINT unq_for_fornecedor UNIQUE (for_nome)  
);
```

```
CREATE TABLE projeto.clt_cliente (  
    clt_id int NOT NULL,  
    clt_nome varchar(40) NOT NULL,  
    clt_fil_id int NOT NULL,  
    CONSTRAINT pk_clt_cliente PRIMARY KEY (clt_id)  
);
```

```
CREATE TABLE projeto.emp_empregado (  
    emp_id int NOT NULL,  
    emp_cpf char(11) NOT NULL,  
    emp_p_nome varchar(20) NOT NULL,  
    emp_sobrenome varchar(40) NOT NULL,  
    emp_salario numeric(8,2) NOT NULL,  
    emp_sexo char(1) NOT NULL,  
    emp_fil_id int NOT NULL,  
    CONSTRAINT unq_emp_empregado UNIQUE (emp_cpf) ,  
    CONSTRAINT pk_emp_empregado PRIMARY KEY (emp_id),  
    CONSTRAINT chk_emp_salario CHECK (emp_salario > 0),  
    CONSTRAINT chk_emp_sexo CHECK (emp_sexo IN ('F', 'M', 'O'))  
);
```

```
CREATE TABLE projeto.ped_pedido (  
    ped_emp_id int NOT NULL,  
    ped_clt_id int NOT NULL,  
    ped_vendas int NOT NULL,  
    CONSTRAINT pk PRIMARY KEY (ped_emp_id, ped_clt_id)  
);
```

```
CREATE TABLE projeto.tel_telefone (  
    tel_clt_telefone varchar(11) NOT NULL,  
    tel_clt_id int NOT NULL,  
    CONSTRAINT pk_tel_telefone PRIMARY KEY (tel_clt_telefone,  
    tel_clt_id),  
    CONSTRAINT unq_tel_telefone UNIQUE (tel_clt_telefone)  
);
```

```
ALTER TABLE projeto.clt_cliente  
ADD CONSTRAINT fk_clt_cliente_fil_filial FOREIGN KEY (clt_fil_id)  
REFERENCES projeto.fil_filial(fil_id);
```

```
ALTER TABLE projeto.emp_empregado  
ADD CONSTRAINT fk_emp_empregado_fil_filial FOREIGN KEY (emp_fil_id)  
REFERENCES projeto.fil_filial(fil_id);
```

```
ALTER TABLE projeto.fil_filial  
ADD CONSTRAINT fk_fil_filial_reg_regiao FOREIGN KEY (fil_reg_id)  
REFERENCES projeto.reg_regiao(reg_id);
```

```
ALTER TABLE projeto.for_fornecedor  
ADD CONSTRAINT fk_for_fornecedor_fil_filial FOREIGN KEY (for_fil_id)  
REFERENCES projeto.fil_filial(fil_id);
```

```
ALTER TABLE projeto.ped_pedido  
ADD CONSTRAINT fk_ped_pedido_emp_empregado FOREIGN KEY  
(ped_emp_id) REFERENCES projeto.emp_empregado(emp_id);
```

```
ALTER TABLE projeto.ped_pedido  
ADD CONSTRAINT fk_ped_pedido_clt_cliente FOREIGN KEY (ped_clt_id)  
REFERENCES projeto.clt_cliente(clt_id);
```

```
ALTER TABLE projeto.tel_telefone  
ADD CONSTRAINT fk_tel_telefone_clt_cliente FOREIGN KEY (tel_clt_id)  
REFERENCES projeto.clt_cliente(clt_id);
```

5. Script para carga de dados no banco:

-- REGIAO

```
INSERT INTO projeto.reg_regiao VALUES(101, 'Cidade 1', 'Bairro 1_c1');
INSERT INTO projeto.reg_regiao VALUES(102, 'Cidade 1', 'Bairro 2_c1');
INSERT INTO projeto.reg_regiao VALUES(103, 'Cidade 1', 'Bairro 3_c1');
INSERT INTO projeto.reg_regiao VALUES(201, 'Cidade 2', 'Bairro 1_c2');
INSERT INTO projeto.reg_regiao VALUES(202, 'Cidade 2', 'Bairro 2_c2');
INSERT INTO projeto.reg_regiao VALUES(203, 'Cidade 2', 'Bairro 3_c2');
```

-- FILIAL

```
INSERT INTO projeto.fil_filial VALUES(1, 'Filial 1', 101);
INSERT INTO projeto.fil_filial VALUES(2, 'Filial 2', 102);
INSERT INTO projeto.fil_filial VALUES(3, 'Filial 3', 103);
INSERT INTO projeto.fil_filial VALUES(4, 'Filial 4', 201);
INSERT INTO projeto.fil_filial VALUES(5, 'Filial 5', 202);
INSERT INTO projeto.fil_filial VALUES(6, 'Filial 6', 203);
```

-- FORNECEDOR

```
INSERT INTO projeto.for_fornecedor VALUES('Fornecedor 1', 1, 'Produto 1',
'Tipo a');
INSERT INTO projeto.for_fornecedor VALUES('Fornecedor 2', 2, 'Produto 2',
'Tipo b');
INSERT INTO projeto.for_fornecedor VALUES('Fornecedor 3', 1, 'Produto 1',
'Tipo a');
INSERT INTO projeto.for_fornecedor VALUES('Fornecedor 4', 5, 'Produto 5',
'Tipo e');
INSERT INTO projeto.for_fornecedor VALUES('Fornecedor 5', 6, 'Produto 6',
'Tipo f');
INSERT INTO projeto.for_fornecedor VALUES('Fornecedor 6', 3, 'Produto 3',
'Tipo c');
INSERT INTO projeto.for_fornecedor VALUES('Fornecedor 7', 5, 'Produto 5',
'Tipo e');
INSERT INTO projeto.for_fornecedor VALUES('Fornecedor 8', 4, 'Produto 4',
'Tipo d');
```

```
INSERT INTO projeto.for_fornecedor VALUES('Fornecedor 9', 1, 'Produto 1',  
'Tipo a');
```

-- CLIENTE

```
INSERT INTO projeto.clt_cliente VALUES(01, 'Cliente 1', 1);  
INSERT INTO projeto.clt_cliente VALUES(02, 'Cliente 2', 3);  
INSERT INTO projeto.clt_cliente VALUES(03, 'Cliente 3', 6);  
INSERT INTO projeto.clt_cliente VALUES(04, 'Cliente 4', 4);  
INSERT INTO projeto.clt_cliente VALUES(05, 'Cliente 5', 1);  
INSERT INTO projeto.clt_cliente VALUES(06, 'Cliente 6', 5);  
INSERT INTO projeto.clt_cliente VALUES(07, 'Cliente 7', 5);  
INSERT INTO projeto.clt_cliente VALUES(08, 'Cliente 8', 2);  
INSERT INTO projeto.clt_cliente VALUES(09, 'Cliente 9', 1);
```

-- TELEFONE CLIENTE

```
INSERT INTO projeto.tel_telefone VALUES('31988725028', 01);  
INSERT INTO projeto.tel_telefone VALUES('3133578010', 01);  
INSERT INTO projeto.tel_telefone VALUES('11994570812', 02);  
INSERT INTO projeto.tel_telefone VALUES('1120568193', 03);  
INSERT INTO projeto.tel_telefone VALUES('21989631278', 04);  
INSERT INTO projeto.tel_telefone VALUES('3170635894', 05);  
INSERT INTO projeto.tel_telefone VALUES('21988508762', 06);  
INSERT INTO projeto.tel_telefone VALUES('1120587364', 07);  
INSERT INTO projeto.tel_telefone VALUES('1125890167', 08);  
INSERT INTO projeto.tel_telefone VALUES('31987690020', 09);
```

-- EMPREGADO

```
INSERT INTO projeto.emp_empregado VALUES(001, 33124455928, 'João',  
'Alencar', 7000, 'M', 1);  
INSERT INTO projeto.emp_empregado VALUES(002, 59003081195, 'Matilde',  
'Oliveira', 9000, 'F', 1);  
INSERT INTO projeto.emp_empregado VALUES(003, 82059675069, 'Leonor',  
'Pereira', 12000, 'M', 1);
```

```
INSERT INTO projeto.emp_empregado VALUES(004, 28915755473, 'Lucas',  
'Cavalcanti', 5000, 'M', 2);  
INSERT INTO projeto.emp_empregado VALUES(005, 87335240787, 'Marisa',  
'Barros', 8000, 'F', 3);  
INSERT INTO projeto.emp_empregado VALUES(006, 16203881796, 'Davi',  
'Ferreira', 6000, 'M', 4);  
INSERT INTO projeto.emp_empregado VALUES(007, 24306192060, 'Arthur',  
'Ribeiro', 7000, 'M', 5);  
INSERT INTO projeto.emp_empregado VALUES(008, 06955969251, 'Tânia',  
'Barros', 4000, 'F', 6);  
INSERT INTO projeto.emp_empregado VALUES(009, 12541343998, 'Erick',  
'Melo', 4000, 'M', 2);  
INSERT INTO projeto.emp_empregado VALUES(010, 17505361688, 'Nicole',  
'Dias', 8000, 'F', 5);
```

-- PEDIDO

```
INSERT INTO projeto.ped_pedido VALUES(003, 01, 80000);  
INSERT INTO projeto.ped_pedido VALUES(001, 05, 50000);  
INSERT INTO projeto.ped_pedido VALUES(002, 09, 70000);  
INSERT INTO projeto.ped_pedido VALUES(004, 08, 20000);  
INSERT INTO projeto.ped_pedido VALUES(005, 02, 30000);  
INSERT INTO projeto.ped_pedido VALUES(006, 04, 25000);  
INSERT INTO projeto.ped_pedido VALUES(007, 06, 40000);  
INSERT INTO projeto.ped_pedido VALUES(008, 03, 10000);  
INSERT INTO projeto.ped_pedido VALUES(009, 08, 15000);  
INSERT INTO projeto.ped_pedido VALUES(010, 07, 60000);
```


6. Consultas SQL e suas respostas:

Query

Query History

1

-- 1: Mostrar todos os empregados

2

SELECT * FROM projeto.emp_empregado

Data Output

Messages

Notifications

≡

+

📄

▼

📋

▼

🗑️

📦

⬇️

📈

	emp_id [PK] integer	emp_cpf character	emp_p_nome character varying (20)	emp_sobrenome character varying (40)	emp_salario numeric (8,2)	emp_sexo character	emp_fil_id integer
1	1	33124455928	João	Alencar	7000.00	M	1
2	2	59003081195	Matilde	Oliveira	9000.00	F	1
3	3	82059675069	Leonor	Pereira	12000.00	M	1
4	4	28915755473	Lucas	Cavalcanti	5000.00	M	2
5	5	87335240787	Marisa	Barros	8000.00	F	3
6	6	16203881796	Davi	Ferreira	6000.00	M	4
7	7	24306192060	Arthur	Ribeiro	7000.00	M	5
8	8	6955969251	Tânia	Barros	4000.00	F	6
9	9	12541343998	Erick	Melo	4000.00	M	2
10	10	17505361688	Nicole	Dias	8000.00	F	5

Query

Query History

1

-- 2: Mostrar todos os empregados ordenados do maior para menor salário

2

SELECT * FROM projeto.emp_empregado

3

ORDER BY emp_salario DESC

Data Output

Messages

Notifications

≡

📄

▼

📋

▼

🗑

📦

⬇

📈

	emp_id [PK] integer	emp_cpf character	emp_p_nome character varying (20)	emp_sobrenome character varying (40)	emp_salario numeric (8,2)	emp_sexo character	emp_fil_id integer
1	3	82059675069	Leonor	Pereira	12000.00	M	1
2	2	59003081195	Matilde	Oliveira	9000.00	F	1
3	5	87335240787	Marisa	Barros	8000.00	F	3
4	10	17505361688	Nicole	Dias	8000.00	F	5
5	1	33124455928	João	Alencar	7000.00	M	1
6	7	24306192060	Arthur	Ribeiro	7000.00	M	5
7	6	16203881796	Davi	Ferreira	6000.00	M	4
8	4	28915755473	Lucas	Cavalcanti	5000.00	M	2
9	8	6955969251	Tânia	Barros	4000.00	F	6
10	9	12541343998	Erick	Melo	4000.00	M	2

Query

Query History

5

-- 3: Mostrar todos os empregados ordenados por sexo e depois por nome

6

SELECT * FROM projeto.emp_empregado

7

ORDER BY emp_sexo, emp_p_nome, emp_sobrenome

Data Output

Messages

Notifications

≡+

📄

▼

📋

▼

🗑️

🗑️

📥

📈

	emp_id [PK] integer	emp_cpf character	emp_p_nome character varying (20)	emp_sobrenome character varying (40)	emp_salario numeric (8,2)	emp_sexo character	emp_fil_id integer
1	5	87335240787	Marisa	Barros	8000.00	F	3
2	2	59003081195	Matilde	Oliveira	9000.00	F	1
3	10	17505361688	Nicole	Dias	8000.00	F	5
4	8	6955969251	Tânia	Barros	4000.00	F	6
5	7	24306192060	Arthur	Ribeiro	7000.00	M	5
6	6	16203881796	Davi	Ferreira	6000.00	M	4
7	9	12541343998	Erick	Melo	4000.00	M	2
8	1	33124455928	João	Alencar	7000.00	M	1
9	3	82059675069	Leonor	Pereira	12000.00	M	1
10	4	28915755473	Lucas	Cavalcanti	5000.00	M	2

Query

Query History

9

-- 4: Mostrar o primeiro nome e sobrenome dos 5 funcionários com os maiores salários.

10

-- O nome das colunas no resultado devem ser nome e sobrenome

11

SELECT

12

emp_p_nome AS Nome,

13

emp_sobrenome AS sobrenome

14

FROM projeto.emp_empregado

15

ORDER BY emp_salario DESC

16

LIMIT 5

Data Output

Messages

Notifications

≡+

📄

▼

📋

▼

🗑️

🗑️

📥

📈

	nome character varying (20) 🔒	sobrenome character varying (40) 🔒
1	Leonor	Pereira
2	Matilde	Oliveira
3	Marisa	Barros
4	Nicole	Dias
5	João	Alencar

Query
Query History

```

18  -- 5: Mostrar a média salarial agrupando por sexo
19  SELECT
20      ROUND(AVG(emp_salario), 2) AS media_salarial,
21      emp_sexo AS sexo
22  FROM projeto.emp_empregado
23  GROUP BY sexo

```

Data Output
Messages
Notifications

≡+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

	media_salarial numeric	sexo character
1	6833.33	M
2	7250.00	F

Query
Query History

```

25  -- 6: Mostrar o nome do funcionário e o seu total de vendas, ordenado do maior para o menor
26  SELECT
27      emp_p_nome AS nome_funcionario,
28      SUM(ped_vendas) AS total
29  FROM projeto.ped_pedido AS pedidos
30  INNER JOIN projeto.emp_empregado AS empregados ON pedidos.ped_emp_id = empregados.emp_id
31  GROUP BY nome_funcionario
32  ORDER BY total DESC

```

Data Output
Messages
Notifications

≡+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

	nome_funcionario character varying (20)	total bigint
1	Leonor	80000
2	Matilde	70000
3	Nicole	60000
4	João	50000
5	Arthur	40000
6	Marisa	30000
7	Davi	25000
8	Lucas	20000
9	Erick	15000
10	Tânia	10000

QueryQuery History

34-- 7: Mostrar uma lista de todos os clientes e fornecedores de cada filial

35SELECT

36 clt_nome AS nome,

37 clt_fil_id AS filial

38FROM projeto.clt_cliente

39UNION

40SELECT

41 for_nome,

42 for_fil_id

43FROM projeto.for_fornecedor


44ORDER BY filial


Data Output


Messages


Notifications


≡+


▼

▼









	nome character varying (40) 🔒	filial integer 🔒
1	Cliente 9	1
2	Fornecedor 9	1
3	Fornecedor 3	1
4	Cliente 5	1
5	Fornecedor 1	1
6	Cliente 1	1
7	Cliente 8	2
8	Fornecedor 2	2
9	Cliente 2	3
10	Fornecedor 6	3
11	Cliente 4	4
12	Fornecedor 8	4
13	Cliente 7	5
14	Cliente 6	5
15	Fornecedor 7	5
16	Fornecedor 4	5
17	Fornecedor 5	6
18	Cliente 3	6

Query

Query History

46

-- 8: Mostrar a descrição de cada região e os nomes das filiais de cada uma

47

SELECT

48

reg_cidade AS cidade,

49

reg_bairro AS bairro,

50

fil_nome AS nome_filial

51

FROM projeto.reg_regiao as regiao

52

INNER JOIN projeto.fil_filial AS filial ON regiao.reg_id = filial.fil_reg_id

53

Data Output

Messages

Notifications

≡+

▼

▼

	<div>cidade</div> <div>character varying (20)</div> <div>🔒</div>	<div>bairro</div> <div>character varying (20)</div> <div>🔒</div>	<div>nome_filial</div> <div>character varying (40)</div> <div>🔒</div>	
1	Cidade 1	Bairro 1_c1	Filial 1	
2	Cidade 1	Bairro 2_c1	Filial 2	
3	Cidade 1	Bairro 3_c1	Filial 3	
4	Cidade 2	Bairro 1_c2	Filial 4	
5	Cidade 2	Bairro 2_c2	Filial 5	
6	Cidade 2	Bairro 3_c2	Filial 6	

Query
Query History

```

54  -- 9: Mostrar quantos números de telefone cada cliente possui e o nome dos clientes
55  SELECT
56      COUNT(tel_clt_telefone) AS qtd_telefone,
57      clt_nome AS nome_cliente
58  FROM projeto.tel_telefone as telefone
59  INNER JOIN projeto.clt_cliente AS cliente ON telefone.tel_clt_id = cliente.clt_id
60  GROUP BY nome_cliente
61  ORDER BY qtd_telefone DESC
62  --

```

Data Output
Messages
Notifications

≡+

📄

▼

📋

▼

🗑️

🗑️

📥

📈

	qtd_telefone bigint	nome_cliente character varying (40)
1	2	Cliente 1
2	1	Cliente 4
3	1	Cliente 9
4	1	Cliente 3
5	1	Cliente 5
6	1	Cliente 2
7	1	Cliente 6
8	1	Cliente 7
9	1	Cliente 8

Query
Query History

```

63  -- 10: Mostrar todos os funcionários que venderam entre 20.000 e 50.000 em um único pedido
64  SELECT
65      emp_p_nome AS nome_funcionario,
66      SUM(ped_vendas) AS total
67  FROM projeto.ped_pedido AS pedidos
68  INNER JOIN projeto.emp_empregado AS empregados ON pedidos.ped_emp_id = empregados.emp_id
69  GROUP BY nome_funcionario
70  HAVING SUM(ped_vendas) >= 20000 AND SUM(ped_vendas) <= 50000
71  ORDER BY total DESC
72  --

```

Data Output
Messages
Notifications

≡+

📄

▼

📋

▼

🗑️

🗑️

📥

📈

	nome_funcionario character varying (20)	total bigint
1	João	50000
2	Arthur	40000
3	Marisa	30000
4	Davi	25000
5	Lucas	20000