Graduação em Engenharia de Computação
Projeto de Bloco: Fundamentos de Dados

TP3

Aluno: Antônio Pedro Lima Bilby Lemos

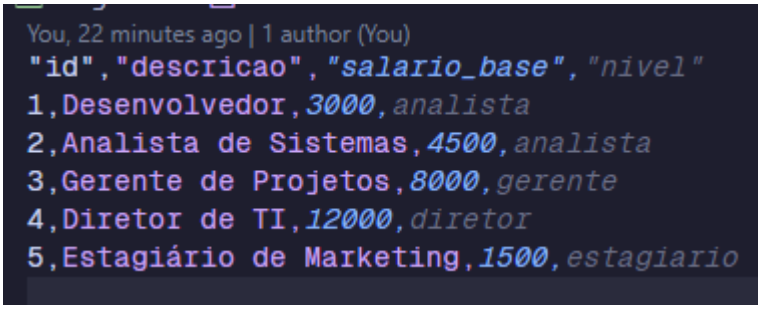
Sumário

1.Introdução

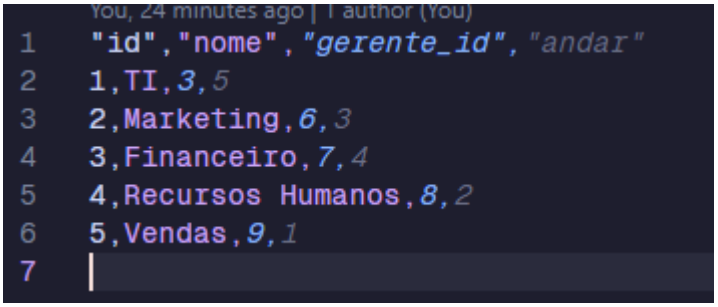
O projeto implementado consiste em um conjunto de queries desenvolvidas num banco MySQL para realizar consultas diversas, envolvendo filtragens, agrupamentos, relações entre tabelas e outros recursos que o MySQL oferece.

2.Observando os dados

2.1.Cargos



2.2.Departamentos



2.3.Dependentes

2.4.Funcionários

```csv
"id","nome","salario","cargo_id","departamento_id"
1,João Silva,3500,1,1
2,Maria Oliveira,4800,2,1
3,Carlos Santos,9000,3,2
4,Ana Costa,1500,5,2
5,José Almeida,6000,2,3
6,Luana Martins,7000,3,4
7,Felipe Souza,8500,4,3
8,Patrícia Lima,4000,2,4
9,Ricardo Pereira,5500,1,5
10,Fernanda Gomes,3000,5,5
```

2.5.Pagamentos

```csv
"id","data_pagamento","valor","funcionario_id"
1,"2024-07-01",3500,1
2,"2024-07-01",4800,2
3,"2024-07-01",9000,3
4,"2024-07-01",1500,4
5,"2024-07-01",6000,5
6,"2024-07-01",7000,6
7,"2024-07-01",8500,7
8,"2024-07-01",4000,8
9,"2024-07-01",5500,9
10,"2024-07-01",3000,10
11,"2024-08-01",3500,1
12,"2024-08-01",4800,2
13,"2024-08-01",9000,3
14,"2024-08-01",1500,4
15,"2024-08-01",6000,5
16,"2024-08-01",7000,6
17,"2024-08-01",8500,7
18,"2024-08-01",4000,8
19,"2024-08-01",5500,9
20,"2024-08-01",3000,10
21,"2024-09-01",3500,1
22,"2024-09-01",4800,2
23,"2024-09-01",9000,3
24,"2024-09-01",1500,4
25,"2024-09-01",6000,5
26,"2024-09-01",7000,6
27,"2024-09-01",8500,7
28,"2024-09-01",4000,8
29,"2024-09-01",5500,9
30,"2024-09-01",3000,10
```

```
31,"2024-10-01",3500,1
32,"2024-10-01",4800,2
33,"2024-10-01",9000,3
34,"2024-10-01",1500,4
35,"2024-10-01",6000,5
36,"2024-10-01",7000,6
37,"2024-10-01",8500,7
38,"2024-10-01",4000,8
39,"2024-10-01",5500,9
40,"2024-10-01",3000,10
41,"2024-11-01",3500,1
42,"2024-11-01",4800,2
43,"2024-11-01",9000,3
44,"2024-11-01",1500,4
45,"2024-11-01",6000,5
46,"2024-11-01",7000,6
47,"2024-11-01",8500,7
48,"2024-11-01",4000,8
49,"2024-11-01",5500,9
50,"2024-11-01",3000,10
51,"2024-12-01",4500,1
52,"2024-12-01",4800,2
53,"2024-12-01",9000,3
54,"2024-12-01",2500,4
55,"2024-12-01",6000,5
56,"2024-12-01",7000,6
57,"2024-12-01",9500,7
58,"2024-12-01",4000,8
59,"2024-12-01",5500,9
60,"2024-12-01",3000,10
```

3.Leitura e inserção dos dados

3.1.CSV

```
You, 20 minutes ago | 1 author (You)
import csv      You, 26 minutes ago • s
import sqlite3
import datetime


def ler_csv(nome_arquivo):
    with open(nome_arquivo, "r") as arquivo:
        leitor = csv.DictReader(arquivo)
        dados = [linha for linha in leitor]
    return dados


cargos = ler_csv("cargos.csv")
departamentos = ler_csv("departamentos.csv")
dependentes = ler_csv("dependentes.csv")
funcionarios = ler_csv("funcionarios.csv")
pagamentos_salarios = ler_csv("pagamentos_salarios.csv")
```

3.2.SQLite3

3.2.1.Criação das tabelas

```
def criar_tabelas():
    conexao = sqlite3.connect("empresa.db")
    cursor = conexao.cursor()
    cursor.execute(
        """
        CREATE TABLE IF NOT EXISTS `funcionarios` (
            `id` INTEGER PRIMARY KEY,
            `nome` TEXT NOT NULL,
            `salario` REAL NOT NULL,
            `cargo_id` INTEGER NOT NULL,
            `departamento_id` INTEGER NOT NULL
        );
        """
    )
    cursor.execute(
        """
        CREATE TABLE IF NOT EXISTS `cargos` (
            `id` INTEGER PRIMARY KEY,
            `descricao` TEXT NOT NULL,
            `salario_base` REAL NOT NULL,
            `nivel` TEXT CHECK(nivel IN ('estagiario', 'tecnico', 'analista', 'gerente', 'diretor')) NOT NULL
        );
        """
    )
```

```
    cursor.execute(
        """
        CREATE TABLE IF NOT EXISTS `departamentos` (
            `id` INTEGER PRIMARY KEY,
            `nome` TEXT NOT NULL,
            `gerente_id` INTEGER,
            `andar` INTEGER NOT NULL,
            FOREIGN KEY (`gerente_id`) REFERENCES `funcionarios` (`id`)
        );
        """
    )
    cursor.execute(
        """
        CREATE TABLE IF NOT EXISTS `pagamentos_salario` (
            `id` INTEGER PRIMARY KEY,
            `data_pagamento` DATE NOT NULL,
            `valor` REAL NOT NULL,
            `funcionario_id` INTEGER NOT NULL,
            FOREIGN KEY (`funcionario_id`) REFERENCES `funcionarios` (`id`)
        );
    """
    )
    cursor.execute(
        """
    CREATE TABLE IF NOT EXISTS `dependentes` (
        `id` INTEGER PRIMARY KEY,
        `nome` TEXT NOT NULL,
        `relacao` TEXT CHECK(relacao IN ('filhos', 'netos', 'conjugues', 'pais', 'avos', 'tios', 'outros')) NOT NULL,
        `funcionario_id` INTEGER NOT NULL,
        `genero` TEXT CHECK(genero IN ('masculino', 'feminino')) NOT NULL,
        `idade` INTEGER NOT NULL,
        FOREIGN KEY (`funcionario_id`) REFERENCES `funcionarios` (`id`)
    );
    """
    )
    conexao.commit()
    conexao.close()
```

3.2.2.Popular tabelas

Para facilitar a integridade do banco, os dados do csv foram inseridos no banco de dados.

```python
def popular_tabelas():
    conexao = sqlite3.connect("empresa.db")
    cursor = conexao.cursor()
    for cargo in cargos:
        cursor.execute(
            """
            INSERT INTO `cargos` (`id`, `descricao`, `salario_base`, `nivel`)
            VALUES (?, ?, ?, ?);
            """,
            (cargo["id"], cargo["descricao"], cargo["salario_base"], cargo["nivel"]),
        )
    for departamento in departamentos:
        cursor.execute(
            """
            INSERT INTO `departamentos` (`id`, `nome`, `gerente_id`, `andar`)
            VALUES (?, ?, ?, ?);
            """,
            (
                departamento["id"],
                departamento["nome"],
                departamento["gerente_id"],
                departamento["andar"],
            ),
        )
    for dependente in dependentes:
        cursor.execute(
            """
            INSERT INTO `dependentes` (`id`, `nome`, `relacao`, `funcionario_id`, `genero`, `idade`)
            VALUES (?, ?, ?, ?, ?, ?);
            """,
            (
                dependente["id"],
                dependente["nome"],
                dependente["relacao"],
                dependente["funcionario_id"],
                dependente["genero"],
                dependente["idade"],
            ),
        )
```

```python
    for funcionario in funcionarios:
        cursor.execute(
            """
            INSERT INTO `funcionarios` (`id`, `nome`, `salario`, `cargo_id`, `departamento_id`)
            VALUES (?, ?, ?, ?, ?);
            """,
            (
                funcionario["id"],
                funcionario["nome"],
                funcionario["salario"],
                funcionario["cargo_id"],
                funcionario["departamento_id"],
            ),
        )
    for pagamento_salario in pagamentos_salarios:
        cursor.execute(
            """
            INSERT INTO `pagamentos_salario` (`id`, `data_pagamento`, `valor`, `funcionario_id`)
            VALUES (?, ?, ?, ?);
            """,
            (
                pagamento_salario["id"],
                pagamento_salario["data_pagamento"],
                pagamento_salario["valor"],
                pagamento_salario["funcionario_id"],
            ),
        )
    conexao.commit()
    conexao.close()
```

4.Consultas

4.1.Listar individualmente as tabelas de: Funcionários, Cargos, Departamentos, Histórico de Salários e Dependentes em ordem crescente.

Leitura dos dados do csv

```python
def listar_tabelas():
    # csv
    print("Cargos")
    for cargo in cargos:
        print(cargo)
    print("Departamentos")
    for departamento in departamentos:
        print(departamento)
    print("Dependentes")
    for dependente in dependentes:
        print(dependente)
    print("Funcionarios")
    for funcionario in funcionarios:
        print(funcionario)
    print("Pagamentos Salarios")
    for pagamento_salario in pagamentos_salarios:
        print(pagamento_salario)


listar_tabelas()
```

```
python/bloco on  main [!?] via  v3.10.12 (.venv)
❯ python3 tp3.py
Cargos
{'id': '1', 'descricao': 'Desenvolvedor', 'salario_base': '3000', 'nivel': 'analista'}
{'id': '2', 'descricao': 'Analista de Sistemas', 'salario_base': '4500', 'nivel': 'analista'}
{'id': '3', 'descricao': 'Gerente de Projetos', 'salario_base': '8000', 'nivel': 'gerente'}
{'id': '4', 'descricao': 'Diretor de TI', 'salario_base': '12000', 'nivel': 'diretor'}
{'id': '5', 'descricao': 'Estagiário de Marketing', 'salario_base': '1500', 'nivel': 'estagiario'}
Departamentos
{'id': '1', 'nome': 'TI', 'gerente_id': '3', 'andar': '5'}
{'id': '2', 'nome': 'Marketing', 'gerente_id': '6', 'andar': '3'}
{'id': '3', 'nome': 'Financeiro', 'gerente_id': '7', 'andar': '4'}
{'id': '4', 'nome': 'Recursos Humanos', 'gerente_id': '8', 'andar': '2'}
{'id': '5', 'nome': 'Vendas', 'gerente_id': '9', 'andar': '1'}
Dependentes
{'id': '1', 'nome': 'Luana Silva', 'relacao': 'filhos', 'funcionario_id': '1', 'genero': 'feminino', 'idade': '10'}
{'id': '2', 'nome': 'Ana Silva', 'relacao': 'filhos', 'funcionario_id': '1', 'genero': 'feminino', 'idade': '8'}
{'id': '3', 'nome': 'Camila Oliveira', 'relacao': 'conjugues', 'funcionario_id': '2', 'genero': 'feminino', 'idade': '30'}
{'id': '4', 'nome': 'Jorge Oliveira', 'relacao': 'filhos', 'funcionario_id': '2', 'genero': 'masculino', 'idade': '5'}
{'id': '5', 'nome': 'Pedro Santos', 'relacao': 'pais', 'funcionario_id': '3', 'genero': 'masculino', 'idade': '60'}
{'id': '6', 'nome': 'Sônia Santos', 'relacao': 'conjugues', 'funcionario_id': '3', 'genero': 'feminino', 'idade': '55'}
{'id': '7', 'nome': 'Marcos Costa', 'relacao': 'netos', 'funcionario_id': '4', 'genero': 'masculino', 'idade': '3'}
{'id': '8', 'nome': 'Patricia Costa', 'relacao': 'conjugues', 'funcionario_id': '4', 'genero': 'feminino', 'idade': '30'}
{'id': '9', 'nome': 'Bruna Almeida', 'relacao': 'filhos', 'funcionario_id': '5', 'genero': 'feminino', 'idade': '12'}
{'id': '10', 'nome': 'Carlos Almeida', 'relacao': 'outros', 'funcionario_id': '5', 'genero': 'masculino', 'idade': '23'}
{'id': '11', 'nome': 'Raquel Martins', 'relacao': 'conjugues', 'funcionario_id': '6', 'genero': 'feminino', 'idade': '27'}
{'id': '12', 'nome': 'Eduardo Martins', 'relacao': 'filhos', 'funcionario_id': '6', 'genero': 'masculino', 'idade': '45'}
{'id': '13', 'nome': 'Lúcia Souza', 'relacao': 'pais', 'funcionario_id': '7', 'genero': 'feminino', 'idade': ' 23'}
{'id': '14', 'nome': 'Ricardo Souza', 'relacao': 'netos', 'funcionario_id': '7', 'genero': 'masculino', 'idade': '35'}
{'id': '15', 'nome': 'Vera Lima', 'relacao': 'outros', 'funcionario_id': '8', 'genero': 'feminino', 'idade': '12'}
{'id': '16', 'nome': 'Juliana Lima', 'relacao': 'pais', 'funcionario_id': '8', 'genero': 'feminino', 'idade': '6'}
{'id': '17', 'nome': 'Felipe Pereira', 'relacao': 'tios', 'funcionario_id': '9', 'genero': 'masculino', 'idade': '60'}
{'id': '18', 'nome': 'Paula Pereira', 'relacao': 'conjugues', 'funcionario_id': '9', 'genero': 'feminino', 'idade': '30'}
{'id': '19', 'nome': 'Márcia Gomes', 'relacao': 'filhos', 'funcionario_id': '10', 'genero': 'feminino', 'idade': '15'}
{'id': '20', 'nome': 'Gustavo Gomes', 'relacao': 'netos', 'funcionario_id': '10', 'genero': 'masculino', 'idade': '14'}
```

```
Funcionarios
{'id': '1', 'nome': 'João Silva', 'salario': '3500', 'cargo_id': '1', 'departamento_id': '1'}
{'id': '2', 'nome': 'Maria Oliveira', 'salario': '4800', 'cargo_id': '2', 'departamento_id': '1'}
{'id': '3', 'nome': 'Carlos Santos', 'salario': '9000', 'cargo_id': '3', 'departamento_id': '2'}
{'id': '4', 'nome': 'Ana Costa', 'salario': '1500', 'cargo_id': '5', 'departamento_id': '2'}
{'id': '5', 'nome': 'José Almeida', 'salario': '6000', 'cargo_id': '2', 'departamento_id': '3'}
{'id': '6', 'nome': 'Luana Martins', 'salario': '7000', 'cargo_id': '3', 'departamento_id': '4'}
{'id': '7', 'nome': 'Felipe Souza', 'salario': '8500', 'cargo_id': '4', 'departamento_id': '3'}
{'id': '8', 'nome': 'Patricia Lima', 'salario': '4000', 'cargo_id': '2', 'departamento_id': '4'}
{'id': '9', 'nome': 'Ricardo Pereira', 'salario': '5500', 'cargo_id': '1', 'departamento_id': '5'}
{'id': '10', 'nome': 'Fernanda Gomes', 'salario': '3000', 'cargo_id': '5', 'departamento_id': '5'}
```

```
Pagamentos Salarios
{'id': '1', 'data_pagamento': '2024-07-01', 'valor': '3500', 'funcionario_id': '1'}
{'id': '2', 'data_pagamento': '2024-07-01', 'valor': '4800', 'funcionario_id': '2'}
{'id': '3', 'data_pagamento': '2024-07-01', 'valor': '9000', 'funcionario_id': '3'}
{'id': '4', 'data_pagamento': '2024-07-01', 'valor': '1500', 'funcionario_id': '4'}
{'id': '5', 'data_pagamento': '2024-07-01', 'valor': '6000', 'funcionario_id': '5'}
{'id': '6', 'data_pagamento': '2024-07-01', 'valor': '7000', 'funcionario_id': '6'}
{'id': '7', 'data_pagamento': '2024-07-01', 'valor': '8500', 'funcionario_id': '7'}
{'id': '8', 'data_pagamento': '2024-07-01', 'valor': '4000', 'funcionario_id': '8'}
{'id': '9', 'data_pagamento': '2024-07-01', 'valor': '5500', 'funcionario_id': '9'}
{'id': '10', 'data_pagamento': '2024-07-01', 'valor': '3000', 'funcionario_id': '10'}
{'id': '11', 'data_pagamento': '2024-08-01', 'valor': '3500', 'funcionario_id': '1'}
{'id': '12', 'data_pagamento': '2024-08-01', 'valor': '4800', 'funcionario_id': '2'}
{'id': '13', 'data_pagamento': '2024-08-01', 'valor': '9000', 'funcionario_id': '3'}
{'id': '14', 'data_pagamento': '2024-08-01', 'valor': '1500', 'funcionario_id': '4'}
{'id': '15', 'data_pagamento': '2024-08-01', 'valor': '6000', 'funcionario_id': '5'}
{'id': '16', 'data_pagamento': '2024-08-01', 'valor': '7000', 'funcionario_id': '6'}
{'id': '17', 'data_pagamento': '2024-08-01', 'valor': '8500', 'funcionario_id': '7'}
{'id': '18', 'data_pagamento': '2024-08-01', 'valor': '4000', 'funcionario_id': '8'}
{'id': '19', 'data_pagamento': '2024-08-01', 'valor': '5500', 'funcionario_id': '9'}
{'id': '20', 'data_pagamento': '2024-08-01', 'valor': '3000', 'funcionario_id': '10'}
{'id': '21', 'data_pagamento': '2024-09-01', 'valor': '3500', 'funcionario_id': '1'}
{'id': '22', 'data_pagamento': '2024-09-01', 'valor': '4800', 'funcionario_id': '2'}
{'id': '23', 'data_pagamento': '2024-09-01', 'valor': '9000', 'funcionario_id': '3'}
{'id': '24', 'data_pagamento': '2024-09-01', 'valor': '1500', 'funcionario_id': '4'}
{'id': '25', 'data_pagamento': '2024-09-01', 'valor': '6000', 'funcionario_id': '5'}
{'id': '26', 'data_pagamento': '2024-09-01', 'valor': '7000', 'funcionario_id': '6'}
{'id': '27', 'data_pagamento': '2024-09-01', 'valor': '8500', 'funcionario_id': '7'}
{'id': '28', 'data_pagamento': '2024-09-01', 'valor': '4000', 'funcionario_id': '8'}
{'id': '29', 'data_pagamento': '2024-09-01', 'valor': '5500', 'funcionario_id': '9'}
{'id': '30', 'data_pagamento': '2024-09-01', 'valor': '3000', 'funcionario_id': '10'}
{'id': '31', 'data_pagamento': '2024-10-01', 'valor': '3500', 'funcionario_id': '1'}
{'id': '32', 'data_pagamento': '2024-10-01', 'valor': '4800', 'funcionario_id': '2'}
{'id': '33', 'data_pagamento': '2024-10-01', 'valor': '9000', 'funcionario_id': '3'}
{'id': '34', 'data_pagamento': '2024-10-01', 'valor': '1500', 'funcionario_id': '4'}
{'id': '35', 'data_pagamento': '2024-10-01', 'valor': '6000', 'funcionario_id': '5'}
{'id': '36', 'data_pagamento': '2024-10-01', 'valor': '7000', 'funcionario_id': '6'}
{'id': '37', 'data_pagamento': '2024-10-01', 'valor': '8500', 'funcionario_id': '7'}
{'id': '38', 'data_pagamento': '2024-10-01', 'valor': '4000', 'funcionario_id': '8'}
{'id': '39', 'data_pagamento': '2024-10-01', 'valor': '5500', 'funcionario_id': '9'}
{'id': '40', 'data_pagamento': '2024-10-01', 'valor': '3000', 'funcionario_id': '10'}
{'id': '41', 'data_pagamento': '2024-11-01', 'valor': '3500', 'funcionario_id': '1'}
{'id': '42', 'data_pagamento': '2024-11-01', 'valor': '4800', 'funcionario_id': '2'}
{'id': '43', 'data_pagamento': '2024-11-01', 'valor': '9000', 'funcionario_id': '3'}
{'id': '44', 'data_pagamento': '2024-11-01', 'valor': '1500', 'funcionario_id': '4'}
{'id': '45', 'data_pagamento': '2024-11-01', 'valor': '6000', 'funcionario_id': '5'}
{'id': '46', 'data_pagamento': '2024-11-01', 'valor': '7000', 'funcionario_id': '6'}
{'id': '47', 'data_pagamento': '2024-11-01', 'valor': '8500', 'funcionario_id': '7'}
{'id': '48', 'data_pagamento': '2024-11-01', 'valor': '4000', 'funcionario_id': '8'}
{'id': '49', 'data_pagamento': '2024-11-01', 'valor': '5500', 'funcionario_id': '9'}
{'id': '50', 'data_pagamento': '2024-11-01', 'valor': '3000', 'funcionario_id': '10'}
{'id': '51', 'data_pagamento': '2024-12-01', 'valor': '4500', 'funcionario_id': '1'}
{'id': '52', 'data_pagamento': '2024-12-01', 'valor': '4800', 'funcionario_id': '2'}
{'id': '53', 'data_pagamento': '2024-12-01', 'valor': '9000', 'funcionario_id': '3'}
{'id': '54', 'data_pagamento': '2024-12-01', 'valor': '2500', 'funcionario_id': '4'}
{'id': '55', 'data_pagamento': '2024-12-01', 'valor': '6000', 'funcionario_id': '5'}
{'id': '56', 'data_pagamento': '2024-12-01', 'valor': '7000', 'funcionario_id': '6'}
{'id': '57', 'data_pagamento': '2024-12-01', 'valor': '9500', 'funcionario_id': '7'}
{'id': '58', 'data_pagamento': '2024-12-01', 'valor': '4000', 'funcionario_id': '8'}
{'id': '59', 'data_pagamento': '2024-12-01', 'valor': '5500', 'funcionario_id': '9'}
{'id': '60', 'data_pagamento': '2024-12-01', 'valor': '3000', 'funcionario_id': '10'}
```

4.2. Listar os funcionários, com seus cargos, departamentos e os respectivos dependentes.

```python
def listar_funcionarios():
    # csv
    for funcionario in funcionarios:
        cargo = [cargo for cargo in cargos if cargo["id"] == funcionario["cargo_id"]][0]
        departamento = [
            departamento
            for departamento in departamentos
            if departamento["id"] == funcionario["departamento_id"]
        ][0]
        dependentes = [
            dependente
            for dependente in dependentes
            if dependente["funcionario_id"] == funcionario["id"]
        ]
        print(f"Funcionario: {funcionario['nome']}, id: {funcionario['id']}")
        print(f"Cargo: {cargo['descricao']}, Nivel: {cargo['nivel']}")
        print(f"Departamento: {departamento['nome']}")
        print(
            f"Dependentes: {', '.join([dependentes['nome'] for dependentes in dependentes])}"
        )
        print("\n")


listar_funcionarios()
```

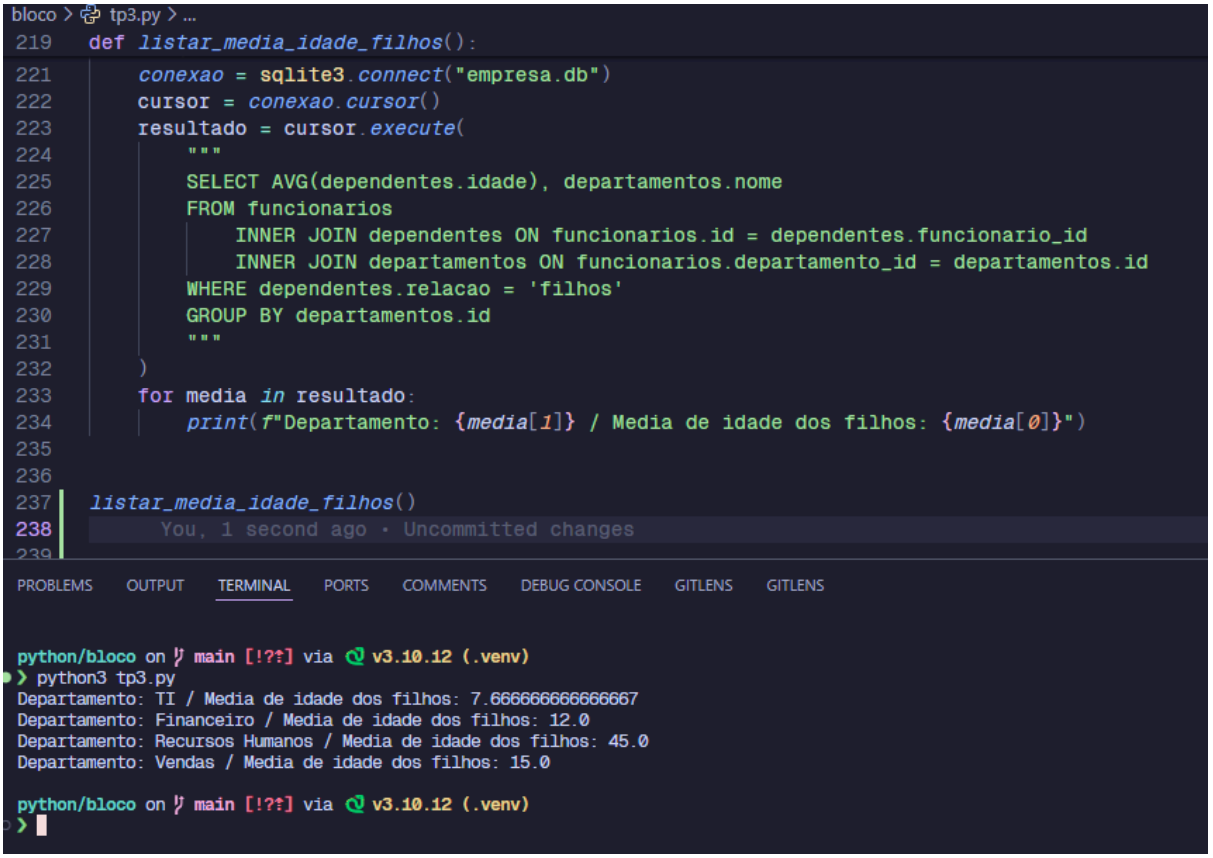
```
python/bloco on  main [!?] via  v3.10.12 (.venv)
❯ python3 tp3.py
Funcionario: João Silva, id: 1
Cargo: Desenvolvedor, Nivel: analista
Departamento: TI
Dependentes: Luana Silva, Ana Silva


Funcionario: Maria Oliveira, id: 2
Cargo: Analista de Sistemas, Nivel: analista
Departamento: TI
Dependentes: Camila Oliveira, Jorge Oliveira


Funcionario: Carlos Santos, id: 3
Cargo: Gerente de Projetos, Nivel: gerente
Departamento: Marketing
Dependentes: Pedro Santos, Sônia Santos


Funcionario: Ana Costa, id: 4
Cargo: Estagiário de Marketing, Nivel: estagiario
Departamento: Marketing
Dependentes: Marcos Costa, Patrícia Costa


Funcionario: José Almeida, id: 5
Cargo: Analista de Sistemas, Nivel: analista
Departamento: Financeiro
Dependentes: Bruna Almeida, Carlos Almeida


Funcionario: Luana Martins, id: 6
Cargo: Gerente de Projetos, Nivel: gerente
Departamento: Recursos Humanos
Dependentes: Raquel Martins, Eduardo Martins


Funcionario: Felipe Souza, id: 7
Cargo: Diretor de TI, Nivel: diretor
Departamento: Financeiro
Dependentes: Lúcia Souza, Ricardo Souza


Funcionario: Patrícia Lima, id: 8
Cargo: Analista de Sistemas, Nivel: analista
Departamento: Recursos Humanos
Dependentes: Vera Lima, Juliana Lima


Funcionario: Ricardo Pereira, id: 9
Cargo: Desenvolvedor, Nivel: analista
Departamento: Vendas
Dependentes: Felipe Pereira, Paula Pereira


Funcionario: Fernanda Gomes, id: 10
Cargo: Estagiário de Marketing, Nivel: estagiario
Departamento: Vendas
Dependentes: Márcia Gomes, Gustavo Gomes
```

4.3.Listar os funcionários que tiveram aumento salarial nos últimos 3 meses.

```python
def listar_funcionarios_aumento_salarial():
    # csv
    funcionarios_com_aumento = []
    for funcionario in funcionarios:
        salarios = [
            salario
            for salario in pagamentos_salarios
            if salario["funcionario_id"] == funcionario["id"]
        ]
        salarios = sorted(
            salarios,
            key=lambda x: datetime.datetime.strptime(x["data_pagamento"], "%Y-%m-%d"),
            reverse=True,
        )
        if salarios[0]["valor"] > salarios[3]["valor"]:
            funcionarios_com_aumento.append(funcionario)
    for funcionario in funcionarios_com_aumento:
        print(f"Funcionario: {funcionario['nome']}, id: {funcionario['id']}")


listar_funcionarios_aumento_salarial()
    You, 1 second ago • Uncommitted changes
```

```
python/bloco on  main [!?⇡] via  v3.10.12 (.venv)
❯ python3 tp3.py
Funcionario: João Silva, id: 1
Funcionario: Ana Costa, id: 4
Funcionario: Felipe Souza, id: 7

python/bloco on  main [!?⇡] via  v3.10.12 (.venv)
❯
```

4.4.Listar a média de idade dos filhos dos funcionários por departamento.



```
bloco > tp3.py > ...
219  def listar_media_idade_filhos():
221      conexao = sqlite3.connect("empresa.db")
222      cursor = conexao.cursor()
223      resultado = cursor.execute(
224          """
225          SELECT AVG(dependentes.idade), departamentos.nome
226          FROM funcionarios
227              INNER JOIN dependentes ON funcionarios.id = dependentes.funcionario_id
228              INNER JOIN departamentos ON funcionarios.departamento_id = departamentos.id
229          WHERE dependentes.relacao = 'filhos'
230          GROUP BY departamentos.id
231          """
232      )
233      for media in resultado:
234          print(f"Departamento: {media[1]} / Media de idade dos filhos: {media[0]}")
235
236
237  listar_media_idade_filhos()
238      You, 1 second ago • Uncommitted changes
239

PROBLEMS  OUTPUT  TERMINAL  PORTS  COMMENTS  DEBUG CONSOLE  GITLENS  GITLENS

python/bloco on  main [!?⇡] via  v3.10.12 (.venv)
❯ python3 tp3.py
Departamento: TI / Media de idade dos filhos: 7.666666666666667
Departamento: Financeiro / Media de idade dos filhos: 12.0
Departamento: Recursos Humanos / Media de idade dos filhos: 45.0
Departamento: Vendas / Media de idade dos filhos: 15.0

python/bloco on  main [!?⇡] via  v3.10.12 (.venv)
❯
```

4.5.Listar qual estagiário possui filho.

```
237  def listar_estagiario_filho():
238      # sql
239      conexao = sqlite3.connect("empresa.db")
240      cursor = conexao.cursor()
241      resultado = cursor.execute(
242          """
243          SELECT funcionarios.nome, dependentes.nome
244          FROM funcionarios
245              INNER JOIN dependentes ON funcionarios.id = dependentes.funcionario_id
246              INNER JOIN cargos ON funcionarios.cargo_id = cargos.id
247          WHERE cargos.nivel = 'estagiario' AND dependentes.relacao = 'filhos'
248          """
249      )
250      for estagiario in resultado:
251          print(f"Estagiario: {estagiario[0]} / Filho: {estagiario[1]}")
252
253
254  listar_estagiario_filho()
255      You, 1 second ago • Uncommitted changes

ROBLEMS  OUTPUT  TERMINAL  PORTS  COMMENTS  DEBUG CONSOLE  GITLENS  GITLENS

ython/bloco on  main [!?⇡] via  v3.10.12 (.venv)
❯ python3 tp3.py
Estagiario: Fernanda Gomes / Filho: Márcia Gomes

ython/bloco on  main [!?⇡] via  v3.10.12 (.venv)
❯
```

4.6.Listar o funcionário que teve o salário médio mais alto.

```python
def listar_funcionario_salario_medio():
    # sql
    conexao = sqlite3.connect("empresa.db")
    cursor = conexao.cursor()
    resultado = cursor.execute(
        """
        SELECT funcionarios.nome, AVG(pagamentos_salario.valor), funcionarios.id
        FROM funcionarios
            INNER JOIN pagamentos_salario ON funcionarios.id = pagamentos_salario.funcionario_id
        GROUP BY funcionarios.id
        ORDER BY AVG(pagamentos_salario.valor) DESC
        LIMIT 1;
    """
    )
    for salario in resultado:
        print(
            f"Funcionario: {salario[0]} / Salario medio: {salario[1]} / id: {salario[2]}"
        )


listar_funcionario_salario_medio()
    You, 1 second ago • Uncommitted changes
```

```
PROBLEMS  OUTPUT  TERMINAL  PORTS  COMMENTS  DEBUG CONSOLE  GITLENS  GITLENS

python/bloco on  main [!?] via  v3.10.12 (.venv)
❯ python3 tp3.py
Funcionario: Carlos Santos / Salario medio: 9000.0 / id: 3
```

4.7.Listar o analista que é pai de 2 (duas) meninas.

```python
def listar_analista_2_filhas():
    # sql
    conexao = sqlite3.connect("empresa.db")
    cursor = conexao.cursor()
    resultado = cursor.execute(
        """
        SELECT analistas.id, analistas.nome
        FROM (
            SELECT
                COUNT(dependentes.id) as filhas,
                funcionarios.id as id,
                funcionarios.nome as nome
            FROM funcionarios
            INNER JOIN cargos ON funcionarios.cargo_id = cargos.id
            INNER JOIN dependentes ON funcionarios.id = dependentes.funcionario_id
            WHERE cargos.nivel = 'analista'
                AND dependentes.genero = 'feminino'
                AND dependentes.relacao = 'filhos'
            GROUP BY funcionarios.id
        ) as analistas
        WHERE filhas >= 2
        """
    )
    for funcionario in resultado:
        print(funcionario)      You, 40 minutes ago • s


listar_analista_2_filhas()
```

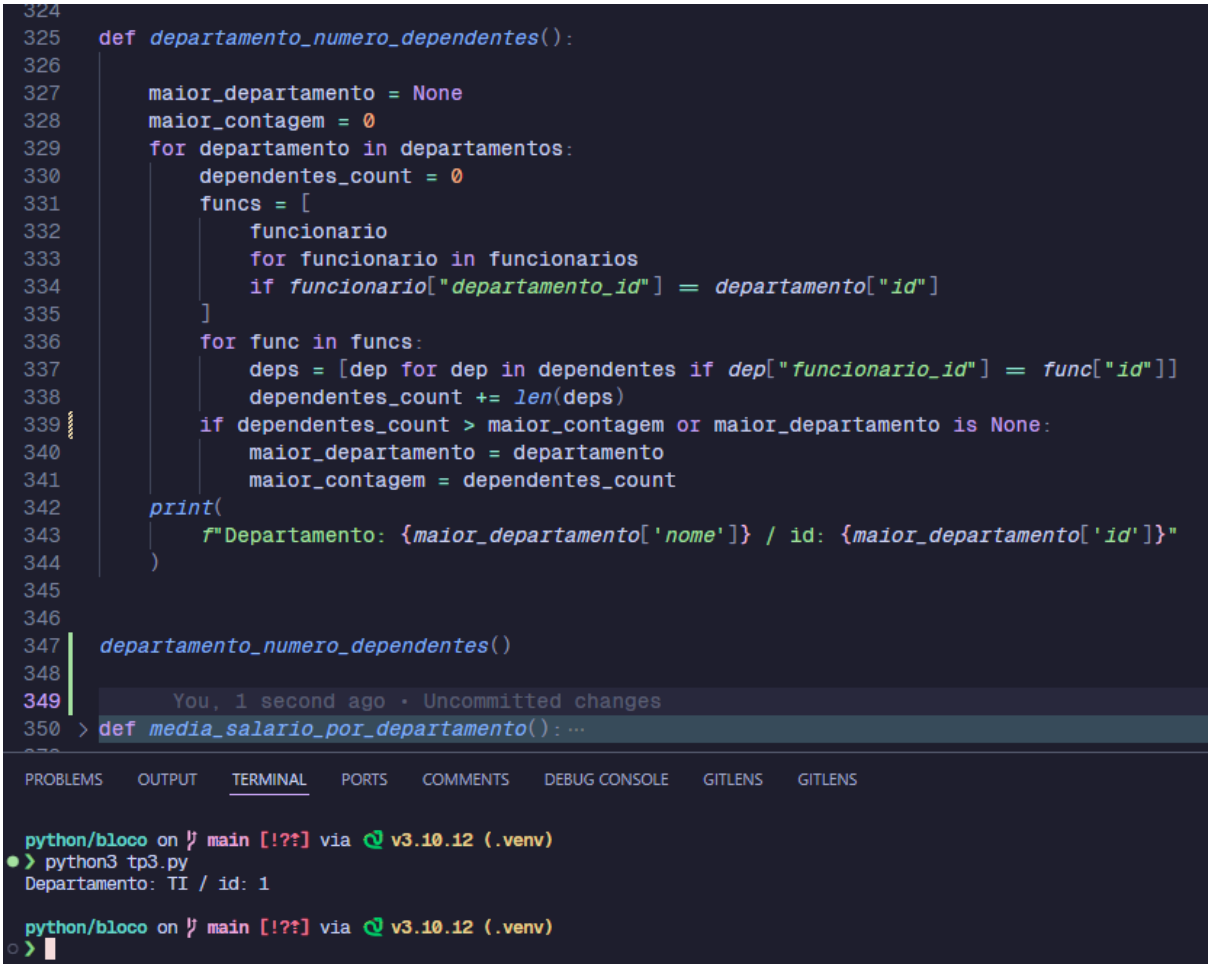
```
PROBLEMS  OUTPUT  TERMINAL  PORTS  COMMENTS  DEBUG CONSOLE  GITLENS  GITLENS

python/bloco on  main [!?] via  v3.10.12 (.venv)
❯ python3 tp3.py
(1, 'João Silva')
```

4.8.Listar o analista que tem o salário mais alto, e que ganhe entre 5000 e 9000.

```python
def listar_analista_salario_alto():
    # sql
    conexao = sqlite3.connect("empresa.db")
    cursor = conexao.cursor()
    resultado = cursor.execute(
        """
        SELECT
            funcionarios.nome,
            funcionarios.id,
            funcionarios.salario
        FROM funcionarios
        INNER JOIN cargos ON cargos.id = funcionarios.cargo_id
        WHERE cargos.nivel = 'analista'
        AND funcionarios.salario BETWEEN 5000 AND 9000
        ORDER BY funcionarios.salario DESC
        LIMIT 1
        """
    )
    for funcionario in resultado:
        print(
            f"Funcionário: {funcionario[0]} / id: {funcionario[1]} / Salário: {funcionario[2]}"
        )


listar_analista_salario_alto()
    You, 1 second ago • Uncommitted changes

> def departamento_numero_dependentes(): ...
```

```
PROBLEMS  OUTPUT  TERMINAL  PORTS  COMMENTS  DEBUG CONSOLE  GITLENS  GITLENS

python/bloco on  main [!?] via  v3.10.12 (.venv)
❯ python3 tp3.py
Funcionário: José Almeida / id: 5 / Salário: 6000.0
```

4.9.Listar qual departamento possui o maior número de dependentes.



```python
def departamento_numero_dependentes():

    maior_departamento = None
    maior_contagem = 0
    for departamento in departamentos:
        dependentes_count = 0
        funcs = [
            funcionario
            for funcionario in funcionarios
            if funcionario["departamento_id"] == departamento["id"]
        ]
        for func in funcs:
            deps = [dep for dep in dependentes if dep["funcionario_id"] == func["id"]]
            dependentes_count += len(deps)
        if dependentes_count > maior_contagem or maior_departamento is None:
            maior_departamento = departamento
            maior_contagem = dependentes_count
    print(
        f"Departamento: {maior_departamento['nome']} / id: {maior_departamento['id']}"
    )


departamento_numero_dependentes()

    You, 1 second ago • Uncommitted changes
def media_salario_por_departamento():
```

```
PROBLEMS  OUTPUT  TERMINAL  PORTS  COMMENTS  DEBUG CONSOLE  GITLENS  GITLENS

python/bloco on  main [!?⇡] via  v3.10.12 (.venv)
❯ python3 tp3.py
Departamento: TI / id: 1

python/bloco on  main [!?⇡] via  v3.10.12 (.venv)
❯
```

4.10.Listar a média de salário por departamento em ordem decrescente.

```python
def media_salario_por_departamento():
    # csv
    medias_por_departamento = []
    for departamento in departamentos:
        salarios = [
            float(func["salario"])
            for func in funcionarios
            if func["departamento_id"] == departamento["id"]
        ]
        medias_por_departamento.append(
            {
                "id": departamento["id"],
                "nome": departamento["nome"],
                "media_salarial": sum(salarios) / len(salarios),
            }
        )
    ordenado_descrescente = sorted(
        medias_por_departamento, key=lambda x: x["media_salarial"], reverse=True
    )
    for dep in ordenado_descrescente:
        print(
            f"Departamento: {dep['nome']} / id: {dep['id']} / Média: {dep['media_salarial']}"
        )


media_salario_por_departamento()
```

```
PROBLEMS  OUTPUT  TERMINAL  PORTS  COMMENTS  DEBUG CONSOLE  GITLENS  GITLENS

python/bloco on  main [!?⇡] via  v3.10.12 (.venv)
❯ python3 tp3.py
Departamento: Financeiro / id: 3 / Média: 7250.0
Departamento: Recursos Humanos / id: 4 / Média: 5500.0
Departamento: Marketing / id: 2 / Média: 5250.0
Departamento: Vendas / id: 5 / Média: 4250.0
Departamento: TI / id: 1 / Média: 4150.0

python/bloco on  main [!?⇡] via  v3.10.12 (.venv)
❯
```