

Instituto Infnet

**Projeto de bloco de Fundamentos de Dados
- TP3 -**

Nome: Cristyan Resende de Souza

**Elói Mendes - MG
2024**

Considerações sobre o projeto:

Este trabalho teve como objetivo criar um sistema de controle de dados de funcionários, utilizando Python e SQLite. Aprendi a integrar diferentes tabelas e realizar consultas que trouxeram insights sobre salários, cargos e dependentes.

O projeto reforçou a importância de uma boa modelagem de dados, facilitando a extração de informações relevantes. As consultas permitiram analisar aumentos salariais, estagiários com filhos e a distribuição de dependentes por departamento.

Essa experiência aprimorou minhas habilidades em programação e análise de dados, além de me dar uma visão mais clara do ciclo de desenvolvimento de sistemas. Estou satisfeito com os resultados e ansioso para aplicar o que aprendi em futuros projetos.

Consultas SQL:

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

```
query_funcionarios = "SELECT * FROM Funcionarios ORDER BY Nome ASC"
df_funcionarios = pd.read_sql_query(query_funcionarios, conn)
print("Funcionários em ordem crescente")
print(df_funcionarios)
```

Funcionários em ordem crescente				
	Id	Nome	Idade	Cargo_id
0	4	Ana Costa	42	3
1	7	Camila Rocha	29	1
2	9	Clara Mendes	31	3
3	8	Diego Martins	38	2
4	10	Fábio Lima	25	1
5	1	João Silva	30	1
6	6	Luana Almeida	33	2
7	2	Maria Oliveira	28	2
8	3	Pedro Santos	35	1
9	5	Ricardo Pereira	26	3

```
query_cargos = "SELECT * FROM Cargos ORDER BY Nome_cargo ASC"
df_cargos = pd.read_sql_query(query_cargos, conn)
print("Cargos em ordem crescente")
print(df_cargos)
```

Cargos em ordem crescente		
	Id	Nome_cargo
0	1	Analista
1	5	Coordenador
2	2	Desenvolvedor
3	3	Estagiário
4	4	Gerente

```

query_departamentos = "SELECT * FROM Departamentos ORDER BY
Nome_departamento ASC"
df_departamentos = pd.read_sql_query(query_departamentos, conn)
print("Departamentos em ordem crescente")
print(df_departamentos)

```

Departamentos em ordem crescente

	Id	Nome_departamento
0	4	Financeiro
1	5	Marketing
2	2	RH
3	1	TI
4	3	Vendas

```

query_historicos = "SELECT * FROM Historico_de_salarios ORDER BY Mes ASC"
df_historicos = pd.read_sql_query(query_historicos, conn)
print("Históricos de salários em ordem crescente")
print(df_historicos)

```

Históricos de salários em ordem crescente

	Id	Mes	Salario	Funcionario_id
0	1	2023-04-01	6000	1
1	2	2023-04-01	8000	2
2	3	2023-04-01	3000	3
3	4	2023-04-01	10000	4
4	5	2023-04-01	9000	5
5	6	2023-04-01	8000	6
6	7	2023-04-01	6000	7
7	8	2023-04-01	8000	8
8	9	2023-04-01	3000	9
9	10	2023-04-01	6000	10
10	11	2023-05-01	6200	1
11	12	2023-05-01	8200	2
12	13	2023-05-01	3100	3
13	14	2023-05-01	10200	4
14	15	2023-05-01	9100	5
15	16	2023-05-01	8200	6
16	17	2023-05-01	6200	7
17	18	2023-05-01	8200	8
18	19	2023-05-01	3100	9
19	20	2023-05-01	6200	10
20	21	2023-06-01	6400	1
21	22	2023-06-01	8400	2
22	23	2023-06-01	3200	3
23	24	2023-06-01	10400	4
24	25	2023-06-01	9200	5
25	26	2023-06-01	8400	6
26	27	2023-06-01	6400	7
27	28	2023-06-01	8400	8
28	29	2023-06-01	3200	9
29	30	2023-06-01	6400	10

```

query_dependentes = "SELECT * FROM Dependentes ORDER BY
Nome_dependente ASC"
df_dependentes = pd.read_sql_query(query_dependentes, conn)
print("Dependentes em ordem crescente")
print(df_dependentes)

```

Dependentes em ordem crescente

	Id	Nome_dependente	Idade	Relacao	Funcionario_id
0	1	Ana Silva	5	Filha	1
1	3	Camila Oliveira	2	Filha	2
2	10	Clara Lima	4	Filha	9
3	5	Juliana Costa	4	Filha	4
4	4	Luis Santos	7	Filho	3
5	6	Mariana Lima	1	Filha	5
6	2	Pedro Silva	3	Filho	1
7	9	Rafael Martins	2	Filho	8
8	7	Roberto Mendes	6	Filho	6
9	8	Sofia Rocha	3	Filha	7

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

```
query_dados_completos = """
SELECT Funcionarios.Nome, Cargos.Nome_cargo,
Departamentos.Nome_departamento, Dependentes.Nome_dependente
FROM Funcionarios
JOIN Cargos ON Funcionarios.Cargo_id = Cargos.Id
JOIN Departamentos ON Funcionarios.Departamento_id = Departamentos.Id
LEFT JOIN Dependentes ON Funcionarios.Id = Dependentes.Funcionario_id;
"""
```

```
df_dados_completos = pd.read_sql(query_dados_completos, conn)
print("Funcionário, cargos, departamentos e dependentes:")
print(df_dados_completos)
```

Funcionário, cargos, departamentos e dependentes:

	Nome	Nome_cargo	Nome_departamento	Nome_dependente
0	João Silva	Analista	TI	Ana Silva
1	João Silva	Analista	TI	Pedro Silva
2	Maria Oliveira	Desenvolvedor	TI	Camila Oliveira
3	Pedro Santos	Analista	RH	Luis Santos
4	Ana Costa	Estagiário	RH	Juliana Costa
5	Ricardo Pereira	Estagiário	Vendas	Mariana Lima
6	Luana Almeida	Desenvolvedor	TI	Roberto Mendes
7	Camila Rocha	Analista	Vendas	Sofia Rocha
8	Diego Martins	Desenvolvedor	Vendas	Rafael Martins
9	Clara Mendes	Estagiário	TI	Clara Lima
10	Fábio Lima	Analista	RH	None

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

```
query_idade_media_filhos = """
SELECT Departamentos.Nome_departamento, AVG(Dependentes.Idade) AS
Media_Idade_Filhos
FROM Funcionarios
JOIN Dependentes ON Funcionarios.Id = Dependentes.Funcionario_id
JOIN Departamentos ON Funcionarios.Departamento_id = Departamentos.Id
WHERE Dependentes.Relacao IN ('Filho', 'Filha')
GROUP BY Departamentos.Nome_departamento;
"""
```

```
df_idade_media_filhos = pd.read_sql(query_idade_media_filhos, conn)
print("Média de idade dos filhos por departamento:")
print(df_idade_media_filhos)
```

Média de idade dos filhos por departamento:		
	Nome_departamento	Media_Idade_Filhos
0	RH	5.5
1	TI	4.0
2	Vendas	2.0

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

```
query_maior_salario_medio = """
```

```
SELECT Funcionarios.Nome, AVG(Historico_de_salarios.Salario) AS salario_medio
FROM Funcionarios
```

```
JOIN Historico_de_salarios ON Funcionarios.Id =
```

```
Historico_de_salarios.Funcionario_id
```

```
GROUP BY Funcionarios.Nome
```

```
ORDER BY salario_medio DESC
```

```
LIMIT 1
```

```
"""
```

```
df_maior_salario_medio = pd.read_sql_query(query_maior_salario_medio, conn)
```

```
print('Maior salário médio:')
```

```
print(df_maior_salario_medio)
```

Maior salário médio:		
	Nome	salario_medio
0	Ana Costa	10200.0

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

```
query_media_salario_dp = """
```

```
SELECT Departamentos.Nome_departamento, AVG(Historico_de_salarios.Salario)
AS salario_medio_dp
```

```
FROM Funcionarios
```

```
JOIN Historico_de_salarios ON Funcionarios.Id =
```

```
Historico_de_salarios.Funcionario_id
```

```
JOIN Departamentos ON Funcionarios.Departamento_id = Departamentos.Id
```

```
GROUP BY Departamentos.Nome_departamento
```

```
ORDER BY salario_medio_dp ASC
```

```
"""
```

```
df_salario_medio_dp = pd.read_sql_query(query_media_salario_dp, conn)
```

```
print("Média de salários por departamento:")
```

```
print(df_salario_medio_dp)
```

Média de salários por departamento:		
	Nome_departamento	salario_medio_dp
0	TI	6425.000000
1	RH	6500.000000
1	RH	6500.000000
2	Vendas	7833.333333

Consultas Python:

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

```
df_historico = pd.read_sql_query("SELECT * FROM Historico_de_salarios", conn)
df_historico['Mes'] = pd.to_datetime(df_historico['Mes'])
ultimo_mes = df_historico['Mes'].max()
meses_atras = ultimo_mes - timedelta(days=90)
df_ultimos_3_meses = df_historico[df_historico['Mes'].between(meses_atras,
ultimo_mes)]
resultados = []
for funcionario_id, grupo in df_ultimos_3_meses.groupby('Funcionario_id'):
    salario_inicial = grupo['Salario'].min()
    salario_final = grupo['Salario'].max()
    if salario_final > salario_inicial:
        nome_funcionario = pd.read_sql_query(f"SELECT Nome FROM Funcionarios
WHERE Id = {funcionario_id}", conn).iloc[0]['Nome']
        resultados.append({
            'Nome': nome_funcionario,
            'Salario_Inicial': salario_inicial,
            'Salario_Final': salario_final
        })
    })
```

```
df_resultados = pd.DataFrame(resultados)
print("Funcionários com aumento salarial nos últimos 3 meses:")
print(df_resultados)
```

```
Funcionários com aumento salarial nos últimos 3 meses:
   Nome  Salario_Inicial  Salario_Final
0  João Silva           3500           3900
1  Maria Oliveira       4000           4400
```

5. Listar qual estagiário possui filho.

```
df_funcionarios = pd.read_sql_query("SELECT * FROM Funcionarios", conn)
df_cargos = pd.read_sql_query("SELECT * FROM Cargos", conn)
df_dependentes = pd.read_sql_query("SELECT * FROM Dependentes", conn)
df_estagiarios = df_funcionarios[df_funcionarios['Cargo_id'].isin(
    df_cargos[df_cargos['Nome_cargo'] == 'Estagiário']['Id']
)]
df_estagiarios_com_filhos = pd.merge(df_estagiarios, df_dependentes, left_on='Id',
right_on='Funcionario_id')
df_estagiarios_com_filhos =
df_estagiarios_com_filhos[df_estagiarios_com_filhos['Relacao'].isin(['Filho', 'Filha'])]
resultados = df_estagiarios_com_filhos[['Nome', 'Nome_dependente']]
resultados.columns = ['Nome_Estagiario', 'Nome_Filho']
print("Estagiários com filhos:")
print(resultados)
```

	Nome_Estagiario	Nome_Filho
0	Ana Costa	Juliana Costa
1	Ricardo Pereira	Mariana Lima
2	Clara Mendes	Clara Lima

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

```
df_funcionarios = pd.read_sql_query("SELECT * FROM Funcionarios", conn)
df_dependentes = pd.read_sql_query("SELECT * FROM Dependentes", conn)
df_cargos = pd.read_sql_query("SELECT * FROM Cargos", conn)
df_analistas = pd.merge(df_funcionarios, df_cargos, left_on='Cargo_id',
right_on='Id')
df_analistas_dependentes = pd.merge(df_analistas, df_dependentes, left_on='Id_x',
right_on='Funcionario_id')
df_analistas = df_analistas_dependentes[df_analistas_dependentes['Nome_cargo']
== 'Analista']
df_analistas_filhas = df_analistas[df_analistas['Relacao'] == 'Filha']
df_analistas_filhas_contagem =
df_analistas_filhas.groupby('Nome').size().reset_index(name='qtd_filhas')
df_analistas_duas_filhas =
df_analistas_filhas_contagem[df_analistas_filhas_contagem['qtd_filhas'] == 2]
if df_analistas_duas_filhas.empty:
    print("Nenhum dos analistas possui exatamente 2 filhas.")
else:
    print("Analistas que têm exatamente 2 filhas:")
    print(df_analistas_duas_filhas)
```

```
Nenhum dos analistas possui exatamente 2 filhas.
```

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

```
df_funcionarios = pd.read_sql_query("SELECT * FROM Funcionarios", conn)
df_cargos = pd.read_sql_query("SELECT * FROM Cargos", conn)
df_historico_salarios = pd.read_sql_query("SELECT * FROM Historico_de_salarios",
conn)
df_analistas =
df_funcionarios[df_funcionarios['Cargo_id'].isin(df_cargos[df_cargos['Nome_cargo']
== 'Analista']['Id'])]
df_salarios_analistas = pd.merge(df_analistas[['Id', 'Nome']], df_historico_salarios,
left_on='Id', right_on='Funcionario_id', how='inner')
df_salarios_analistas = df_salarios_analistas[(df_salarios_analistas['Salario'] >=
5000) & (df_salarios_analistas['Salario'] <= 9000)]
if not df_salarios_analistas.empty:
    df_maior_salario_analista =
df_salarios_analistas.loc[df_salarios_analistas['Salario'].idxmax()]
    print("Analista com o maior salário entre 5000 e 9000:")
```

```
print(f"Nome: {df_maior_salario_analista['Nome']}, Salário: {df_maior_salario_analista['Salario']}")
else:
    print("Nenhum analista possui salário entre 5000 e 9000.")
```

```
Analista com o maior salário entre 5000 e 9000:
Nome: João Silva, Salário: 6400
```

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

```
df_funcionarios = pd.read_sql_query("SELECT * FROM Funcionarios", conn)
df_dependentes = pd.read_sql_query("SELECT * FROM Dependentes", conn)
df_combined = pd.merge(df_dependentes, df_funcionarios, left_on='Funcionario_id',
right_on='Id')
df_count_dependents =
df_combined.groupby('Departamento_id').size().reset_index(name='qtd_dependente
s')
df_departamentos = pd.read_sql_query("SELECT * FROM Departamentos", conn)
df_final = pd.merge(df_count_dependents, df_departamentos,
left_on='Departamento_id', right_on='Id')
departamento_maior_dependentes =
df_final.loc[df_final['qtd_dependentes'].idxmax()]
print("Departamento com o maior número de dependentes:")
print(departamento_maior_dependentes[['Nome_departamento',
'qtd_dependentes']])
```

```
Departamento com o maior número de dependentes:
Nome_departamento    TI
qtd_dependentes       5
```