**Modificações feitas no diagrama desde o último envio:**

Agora a relação entre Aluno e Vaga “SE QUALIFICA” passou a ser “SE INSCREVE”, e a verificação se um aluno está ou não qualificado para uma vaga é feita com base nas habilidades desenvolvidas pelo aluno ao completar um curso.

Vaga passou a ter um nome.

Habilidade passou a ser uma entidade própria, evitando problemas possíveis de acontecer com o caso anterior com Vaga e Curso tendo atributos multivariados para as habilidades. Agora descobrimos a habilidade requisitada por uma vaga e a habilidade oferecida por um curso através de uma relação utilizando as chaves estrangeiras.

**Consultas SQL**

**Seleciona o e-mail, o nome, a data de nascimento, o status do plano, o número de vagas no qual cada aluno se inscreveu e a idade para os alunos inscritos em uma vaga.**

def get\_alunos(self, nome: str = "", status\_plano: str = "", email: str = ""):

current\_date = datetime.now().date()

query = f"""

SELECT A.email, A.nome, TO\_CHAR(A.data\_nascimento, 'YYYY-MM-DD') AS data\_nascimento, A.status\_plano, COUNT(S.id\_vaga) AS numero\_inscricoes, DATE\_PART('year', AGE('{current\_date}', A.data\_nascimento))::INT AS idade

FROM Aluno A LEFT JOIN Se\_Inscreve S ON A.email = S.email\_aluno

"""

filtros = []

if nome:

filtros.append(f"LOWER(A.nome) LIKE '%{nome.lower()}%'")

if status\_plano:

filtros.append(f"LOWER(A.status\_plano) = '{status\_plano.lower()}'")

if email:

filtros.append(f"LOWER(A.email) LIKE '%{email.lower()}%'")

if filtros:

query += " WHERE " + " AND ".join(filtros)

query += """

GROUP BY A.email, A.nome, A.data\_nascimento, A.status\_plano

ORDER BY A.nome ASC

"""

return self.db.execute\_select\_all(query)

**Lista o número de alunos com plano ativo.**

SELECT COUNT(\*) as count FROM Aluno

WHERE status\_plano = 'ativo'"

**Lista o número de alunos com plano inativo.**

SELECT COUNT(\*) as count FROM Aluno

WHERE status\_plano = 'inativo'"

**Lista o nome, a descrição, a duração, o nível, a data de lançamento, o número de alunos que concluiram e as habilidades oferecidas e seu nível para um curso.**

def get\_cursos(self, nome: str = "", duracao: int = None, nivel: str = ""):

SELECT C.nome, C.descricao, C.duracao, C.nivel, TO\_CHAR(C.data\_lancamento, 'DD/MM/YYYY') AS data\_lancamento, COUNT(E.email\_aluno) AS numero\_alunos\_concluidos, STRING\_AGG(DISTINCT H.nome || ': ' || H.nivel, ', ') AS habilidades

FROM Curso C LEFT JOIN Estuda E ON C.nome = E.nome\_curso AND E.data\_conclusao IS NOT NULL LEFT JOIN habilidade\_curso HC ON C.nome = HC.nome\_curso LEFT JOIN Habilidade H ON HC.id\_habilidade = H.id

filtros = []

if nome:

filtros.append(f"LOWER(C.nome) LIKE '%{nome.lower()}%'")

if duracao is not None:

filtros.append(f"C.duracao = {duracao}")

if nivel:

filtros.append(f"LOWER(C.nivel) = '{nivel.lower()}'")

if filtros:

query += " WHERE " + " AND ".join(filtros)

query += """

GROUP BY C.nome, C.descricao, C.duracao, C.nivel, C.data\_lancamento

"""

query += " ORDER BY C.nome ASC"

return self.db.execute\_select\_all(query)

**Lista o número de curso disponiblizados.**

SELECT COUNT(\*)

FROM Curso

**Lista o nome, a localização, o setor e o número de vagas oferecidas das empresas.**

def get\_empresas(self, nome: str = "", setor: str = "", localizacao: str = ""):

query = """

SELECT E.nome AS empresa\_nome, E.localizacao, E.setor, COUNT(V.id) AS numero\_vagas

FROM Empresa E LEFT JOIN Vaga V ON E.nome = V.empresa

"""

filtros = []

if nome:

filtros.append(f"LOWER(E.nome) LIKE '%{nome.lower()}%'")

if setor:

filtros.append(f"LOWER(E.setor) LIKE '%{setor.lower()}%'")

if localizacao:

filtros.append(f"LOWER(E.localizacao) LIKE '%{localizacao.lower()}%'")

if filtros:

query += " WHERE " + " AND ".join(filtros)

query += """

GROUP BY

E.nome, E.localizacao, E.setor

ORDER BY

E.nome ASC

"""

return self.db.execute\_select\_all(query)

**Lista o número de empresas parceiras.**

SELECT COUNT(\*)

FROM empresa

**Lista o e-mail, o nome e a especialização dos professores.**

def get\_professores(self, nome: str = "", especializacao: str = "", email: str = ""):

query = "SELECT \*

FROM Professor"

filtros = []

if nome:

filtros.append(f"LOWER(nome) LIKE '%{nome.lower()}%'")

if especializacao:

filtros.append(f"LOWER(especializacao) LIKE '%{especializacao.lower()}%'")

if email:

filtros.append(f"LOWER(email) LIKE '%{email.lower()}%'")

if filtros:

query += " WHERE " + " AND ".join(filtros)

query += " ORDER BY nome ASC"

return self.db.execute\_select\_all(query)

**Lista o número de professores cadastrados.**

SELECT COUNT(\*) as count

FROM professor

**Lista o id, o nome, a descrição, a localização, o nome que empresa que oferece, o número de alunos inscritos, as habilidades necessárias e os seus respectivos níveis para as vagas.**

def get\_vagas(self, id: int = None, nome: str = "", empresa: str = "", requisitos: str = "", ordenar\_por: str = "numero\_inscritos", ordenar\_ordem: str = "DESC"):

query = """

SELECT V.id, V.nome AS vaga\_nome, V.descricao, E.localizacao, E.nome AS empresa\_nome, COUNT(DISTINCT S.email\_aluno) AS numero\_inscritos, STRING\_AGG(DISTINCT H.nome || ': ' || H.nivel, ', ') AS requisitos

FROM Vaga V LEFT JOIN Empresa E ON V.Empresa = E.nome LEFT JOIN se\_inscreve S ON V.id = S.id\_vaga LEFT JOIN Habilidade\_Vaga HV ON V.id = HV.id\_vaga LEFT JOIN Habilidade H ON HV.id\_habilidade = H.id

"""

filtros = []

if id is not None:

filtros.append(f"V.id = {id}")

if nome:

filtros.append(f"LOWER(V.nome) LIKE '%{nome.lower()}%'")

if empresa:

filtros.append(f"LOWER(E.nome) LIKE '%{empresa.lower()}%'")

if requisitos:

filtros.append(f"""

EXISTS (

SELECT 1

FROM Habilidade\_Vaga HV JOIN Habilidade H ON HV.id\_habilidade = H.id

WHERE HV.id\_vaga = V.id AND LOWER(H.nome) LIKE '%{requisitos.lower()}%' )

""")

if filtros:

query += " WHERE " + " AND ".join(filtros)

query += """

GROUP BY

V.id, V.nome, V.descricao, E.localizacao, E.nome

ORDER BY

{ordenar\_por} {ordenar\_ordem}

""".format(ordenar\_por=ordenar\_por, ordenar\_ordem=ordenar\_ordem)

return self.db.execute\_select\_all(query)

**Lista o número vagas disponíveis.**

SELECT COUNT(\*)

FROM Vaga

**Lista os alunos inscritos para uma vaga, com os atributos úteis para essa busca.**

def get\_vagas\_inscritas\_por\_aluno(self, email\_aluno: str, vaga\_nome: str = "", empresa\_nome: str = "", localizacao: str = "", requisitos: str = "", ordenar\_por: str = "numero\_inscritos", ordenar\_ordem: str = "DESC"):

aluno\_query = f"""

SELECT nome AS aluno\_nome

FROM Aluno

WHERE email = '{email\_aluno}'

"""

aluno\_nome = self.db.execute\_select\_one(aluno\_query)['aluno\_nome']

vagas\_query = f"""

SELECT V.id AS vaga\_id, V.nome AS vaga\_nome, E.nome AS empresa\_nome, E.localizacao, (SELECT COUNT(\*)

FROM se\_inscreve si\_sub

WHERE si\_sub.id\_vaga = v.id

) AS numero\_inscritos, json\_agg(DISTINCT (H.nome || ': ' || H.nivel)

) AS requisitos

FROM Se\_Inscreve S INNER JOIN Vaga V ON S.id\_vaga = V.id LEFT JOIN Empresa E ON V.empresa = E.nome LEFT JOIN Habilidade\_Vaga HV ON V.id = HV.id\_vaga LEFT JOIN Habilidade H ON HV.id\_habilidade = H.id

WHERE S.email\_aluno = '{email\_aluno}'

"""

if vaga\_nome:

vagas\_query += f" AND LOWER(V.nome) LIKE '%{vaga\_nome.lower()}%'"

if empresa\_nome:

vagas\_query += f" AND LOWER(e.nome) LIKE '%{empresa\_nome.lower()}%'"

if localizacao:

vagas\_query += f" AND LOWER(e.localizacao) LIKE '%{localizacao.lower()}%'"

if requisitos:

vagas\_query += f"""

AND EXISTS (

SELECT 1

FROM habilidade\_vaga hv

JOIN habilidade h ON hv.id\_habilidade = h.id

WHERE hv.id\_vaga = v.id

AND LOWER(h.nome) LIKE '%{requisitos.lower()}%'

)

"""

vagas\_query += f"""

GROUP BY V.id, V.nome, E.nome, E.localizacao

ORDER BY {ordenar\_por} {ordenar\_ordem}

"""

vagas\_inscritas = self.db.execute\_select\_all(vagas\_query)

habilidades\_query = f"""

SELECT STRING\_AGG(h.nome || ': ' || h.nivel, ', ') AS habilidade

FROM Estuda E INNER JOIN Habilidade\_Curso HC ON E.nome\_curso = HC.nome\_curso INNER JOIN Habilidade H ON HC.id\_habilidade = H.id

WHERE E.email\_aluno = '{email\_aluno}'

"""

habilidades\_aluno = self.db.execute\_select\_all(habilidades\_query)

return {

"aluno\_nome": aluno\_nome,

"habilidades\_aluno": [{"habilidade": habilidades\_aluno[0]['habilidade']}],

"vagas\_inscritas": vagas\_inscritas if vagas\_inscritas else []

}

**Lista o nome e o nível das habilidades de um curso.**

SELECT H.nome, H.nivel

FROM Habilidade\_Curso HC

JOIN Habilidade H ON HC.id\_habilidade = H.id

WHERE HC.nome\_curso = '{nome\_curso}'

**Lista o nome, o e-mail, a data de conclusão e a nota de um aluno inscrito em um dado curso.**

SELECT A.nome AS aluno\_nome, A.email AS aluno\_email, TO\_CHAR(E.data\_conclusao, 'DD/MM/YYYY') AS data\_conclusao, E.nota

FROM Estuda E INNER JOIN Aluno A ON E.email\_aluno = A.email

WHERE E.nome\_curso = '{nome\_curso}' AND E.data\_conclusao IS NOT NULL

**Lista o nome e o nível de um habilidade requisitada por uma vaga com base no ID da vaga.**

SELECT H.nome, H.nivel

FROM Habilidade\_Vaga HV JOIN Habilidade H ON HV.id\_habilidade = H.id

WHERE HV.id\_vaga = {id\_vaga}

**Lista o e-mail, o nome, a data de nascimento e o status do plano de um aluno inscrito em uma vaga com base no ID da vaga.**

SELECT A.email, A.nome, A.data\_nascimento, A.status\_plano

FROM Aluno A INNER JOIN Se\_Inscreve S ON A.email = S.email\_aluno

WHERE S.id\_vaga = {id\_vaga}

**Lista o nome, o nível e a duração dos cursos feitos por um aluno com base no e-mail do aluno.**

SELECT C.nome, C.nivel, C.duracao

FROM Curso C INNER JOIN Estuda E ON C.nome = E.nome\_curso

WHERE E.email\_aluno = '{aluno["email"]}'

**Lista o nome e o nível das habilidades oferecidas pelos cursos realizados por um aluno com base no seu e-mail.**

SELECT DISTINCT H.nome, H.nivel

FROM Habilidade\_Curso HC JOIN Habilidade H ON HC.id\_habilidade = H.id INNER JOIN Estuda E ON HC.nome\_curso = E.nome\_curso

WHERE E.email\_aluno = '{aluno["email"]}'

**Lista o nome, o nível, a duração, a data de lançamento e as habilidades oferecidas dos cursos sugeridos para um aluno para que ele se qualifique a uma vaga.**

SELECT subquery.nome, subquery.nivel, subquery.duracao, subquery.data\_lancamento, subquery.habilidade

FROM ( SELECT DISTINCT c.nome, c.nivel, c.duracao, TO\_CHAR(c.data\_lancamento, 'DD/MM/YYYY') AS data\_lancamento, H.nome AS habilidade

FROM curso c

INNER JOIN habilidade\_curso hc ON c.nome = hc.nome\_curso

INNER JOIN habilidade h ON hc.id\_habilidade = h.id

WHERE h.nome IN ('{habilidades\_str}')

**Lista o e-mail, nome e status plano de um aluno de acordo com o email.**

SELECT email, nome, status\_plano

FROM Aluno

WHERE LOWER(email) = LOWER('{email\_aluno}')

**Lista o número de curso concluídos por um aluno.**

SELECT COUNT(\*) AS cursos\_concluidos

FROM Estuda

WHERE LOWER(email\_aluno) = LOWER('{email\_aluno}') AND data\_conclusao IS NOT NULL

**Lista o número de vagas que um aluno se inscreveu.**

SELECT COUNT(\*) AS vagas\_inscritas

FROM Se\_Inscreve

WHERE LOWER(email\_aluno) = LOWER('{email\_aluno}')

**Lista as habilidades de um aluno.**

SELECT DISTINCT LOWER(h.nome || ': ' || h.nivel) AS habilidade

FROM Estuda E INNER JOIN Habilidade\_Curso HC ON E.nome\_curso = HC.nome\_curso INNER JOIN Habilidade H ON HC.id\_habilidade = H.id

WHERE LOWER(E.email\_aluno) = LOWER('{email\_aluno}') AND E.data\_conclusao IS NOT NULL

**Lista o nome, o nível, a data de conclusão e a nota dos cursos completados for um aluno.**

SELECT C.nome, C.nivel, TO\_CHAR(e.data\_conclusao, 'DD/MM/YYYY') AS data\_conclusao, E.nota

FROM Curso C INNER JOIN Estuda E ON C.nome = E.nome\_curso

WHERE LOWER(E.email\_aluno) = LOWER('{email\_aluno}') AND E.data\_conclusao IS NOT NULL

if filtro\_nome\_curso:

query\_detalhes\_cursos\_concluidos += f" AND LOWER(c.nome) LIKE '%{filtro\_nome\_curso}%'"

if filtro\_habilidade:

query\_detalhes\_cursos\_concluidos += f""" AND LOWER(c.nome) IN (

SELECT LOWER(nome\_curso)

FROM Habilidade\_Curso HC

INNER JOIN Habilidade H ON HC.id\_habilidade = H.id

WHERE LOWER(H.nome) LIKE '%{filtro\_habilidade}%')

"""

if filtro\_nivel:

query\_detalhes\_cursos\_concluidos += f" AND LOWER(c.nivel) LIKE '%{filtro\_nivel}%'"

query\_detalhes\_cursos\_concluidos += f" ORDER BY e.nota {ordem\_nota}"

**Lista o nome e o nível das habilidades oferecidas por um curso com base no nome do curso.**

SELECT LOWER(H.nome || ': ' || H.nivel) AS habilidade

FROM Habilidade\_Curso HC INNER JOIN Habilidade H ON HC.id\_habilidade = H.id

WHERE LOWER(HC.nome\_curso) = LOWER('{curso["nome"]}')