



CODIFY

# Projeto de Banco de Dados II

Plataforma de cursos online - Codify



CODIFY

Turma 94

Integrantes:

- Henrique Naoki Teruya - 14578324
- João Victor Fonseca Souza - 4442884
- Rodrigo Kenji Sagara Nishimi - 14749001
- Thiago Kenji Saito Espindola - 14671051



CODIFY

Henrique Naoki Teruya - Concepção da plataforma e levantamento dos requisitos e de dados.

João Victor Fonseca Souza - Desenvolvimento do backend e consultas SQL.

Rodrigo Kenji Sagara Nishimi - Modelagem e criação do BD e população no SQL e otimizações.

Thiago Kenji Saito Espindola - Desenvolveu o design e implementou o frontend



# O Ecossistema Codify

A Codify é um ecossistema digital de educação e carreira. Nossa missão é conectar profissionais em busca de desenvolvimento contínuo a empresas que procuram talentos qualificados, transformando o aprendizado em oportunidades de trabalho reais através de uma plataforma integrada e intuitiva.



# Funcionalidades

## Requisitos Funcionais

- **Gestão de Conteúdo:** Permitir a criação, edição e publicação de cursos e aulas.
- **Gestão de Usuários:** Suportar o cadastro e login de usuários.
- **Experiência do Aluno:** Oferecer um catálogo de cursos com filtros, permitir a inscrição, acompanhar o progresso e buscar vagas.

## Requisitos Não Funcionais

- **Performance:** O sistema deve ser rápido.
- **Usabilidade:** A interface deve ser intuitiva, limpa e fácil de usar.

# Dados necessários

## Dados sobre o Curso:

- Título do Curso
- Descrição Detalhada
- Categoria
- Nível
- Habilidades

## Dados sobre o Usuário (Aluno):

- Nome Completo
- Email
- Senha
- Cursos Matriculados
- Progresso por Curso

## Dados sobre a Vaga:

- Título da Vaga
- Nome da Empresa
- Descrição da Vaga e Requisitos
- Nível
- Modalidade
- Localização (se aplicável)

# O uso da IA

- Ideias iniciais
- Estruturação e Refinamento
- Personas e Funcionalidades
- Estratégia de UX

## IAG

- ChatGPT
- Gemini



# Modelagem do BD



# TABELAS

Estrutura do Banco de Dados da Codify: Organizando Informações Essenciais

## 13 Tabelas:

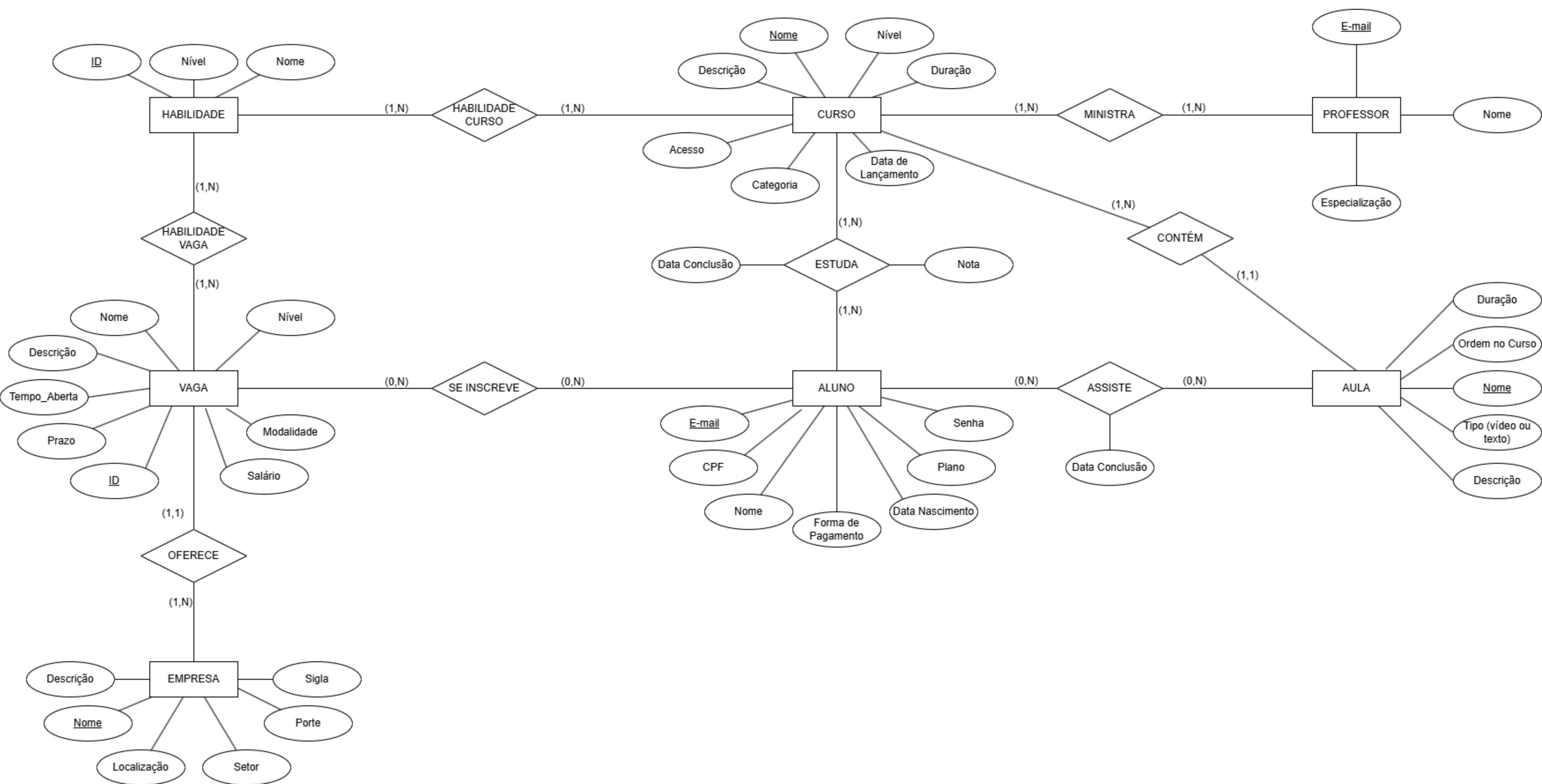
Entidades:

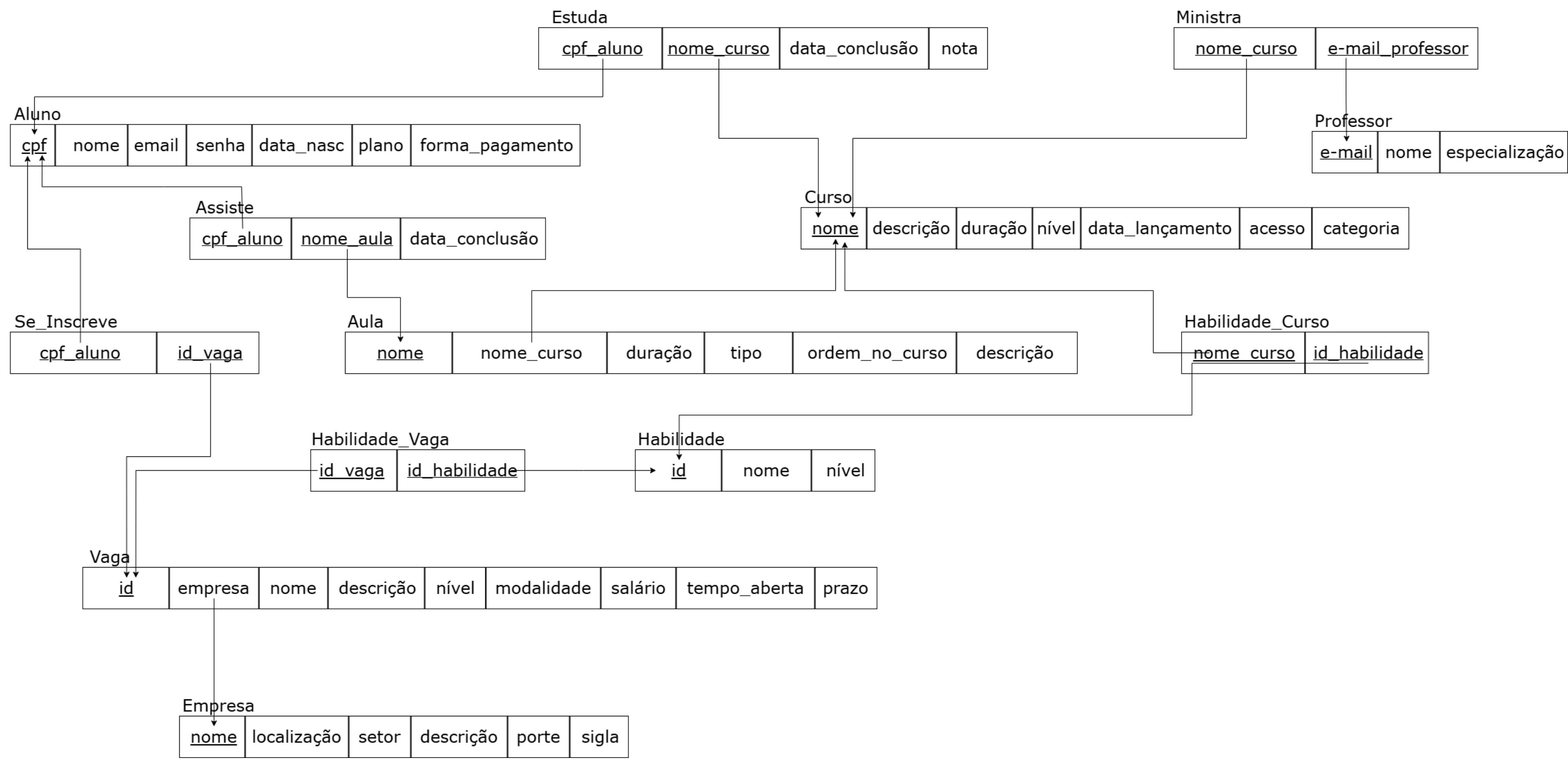
- Aluno
- Professor
- Curso
- Aula
- Empresa
- Vaga
- Habilidade

Relacionamentos:

- Estuda (Aluno-Curso)
- Assiste (Aluno-Aula)
- Ministra (Professor-Curso)
- Se Inscreve (Aluno-Vaga)
- Oferece (Curso-Habilidade)
- Requisita (Vaga-Habilidade)







# SQL e Otimizações



# OTIMIZAÇÕES

O SGBD que utilizamos foi o PostgreSQL;

As estratégias de otimização foram feitas com o auxílio do Github Copilot e Gemini.

As estratégias de otimização utilizadas foram:

- Indexação estratégica;
- Particionamento de Tabelas Grandes.



# OTIMIZAÇÕES

**Principais tabelas utilizadas nos testes de otimização:**

- **Aluno:** 10.000 registros
- **Se\_Inscreve:** 300.000 registros
- **Estuda:** 510.000 registros
- **Assiste:** 2.160.000 registros



# OTIMIZAÇÕES

## Indexação estratégica:

- Índices de Hash para buscas de igualdade: no nosso sistema as consultas principais e mais frequentes são de busca de cursos, aulas e e vagas de um usuário do sistema através do seu CPF.

```
-- ÍNDICES DE HASH PARA BUSCAS DE IGUALDADE PARA AS TABELAS MAIS VOLUMOSAS
CREATE INDEX idx_estuda_cpf_aluno_hash ON Estuda USING HASH (cpf_aluno);
CREATE INDEX idx_assiste_cpf_aluno_hash ON Assiste USING HASH (cpf_aluno);
CREATE INDEX idx_se_inscreve_cpf_aluno_hash ON Se_Inscreve USING HASH (cpf_aluno);

-- ÍNDICES B-TREE PARA CHAVES ESTRANGEIRAS (ACELERAR JOINS)
CREATE INDEX idx_assiste_nome_aula ON Assiste (nome_aula);
CREATE INDEX idx_se_inscreve_id_vaga ON Se_Inscreve (id_vaga);
CREATE INDEX idx_aula_nome_curso ON Aula (nome_curso);
CREATE INDEX idx_vaga_empresa ON Vaga (empresa);
CREATE INDEX idx_habilidade_curso_id_habilidade ON Habilidade_Curso (id_habilidade);
CREATE INDEX idx_ministra_nome_curso ON Ministra (nome_curso);
CREATE INDEX idx_habilidade_vaga_id_habilidade ON Habilidade_Vaga (id_habilidade);
```

# OTIMIZAÇÕES

## Indexação estratégica:

- As chaves primárias já são indexadas por padrão pelo SGBD. Porém, as chaves estrangeiras, utilizadas nas operações JOIN não são. Indexá-las acelera as operações de JOIN.

```
-- ÍNDICES B-TREE PARA CHAVES ESTRANGEIRAS (ACELERAR JOINS)
CREATE INDEX idx_assiste_nome_aula ON Assiste (nome_aula);
CREATE INDEX idx_se_inscreve_id_vaga ON Se_Inscreve (id_vaga);
CREATE INDEX idx_aula_nome_curso ON Aula (nome_curso);
CREATE INDEX idx_vaga_empresa ON Vaga (empresa);
CREATE INDEX idx_habilidade_curso_id_habilidade ON Habilidade_Curso (id_habilidade);
CREATE INDEX idx_ministra_nome_curso ON Ministra (nome_curso);
CREATE INDEX idx_habilidade_vaga_id_habilidade ON Habilidade_Vaga (id_habilidade);
```

# OTIMIZAÇÕES

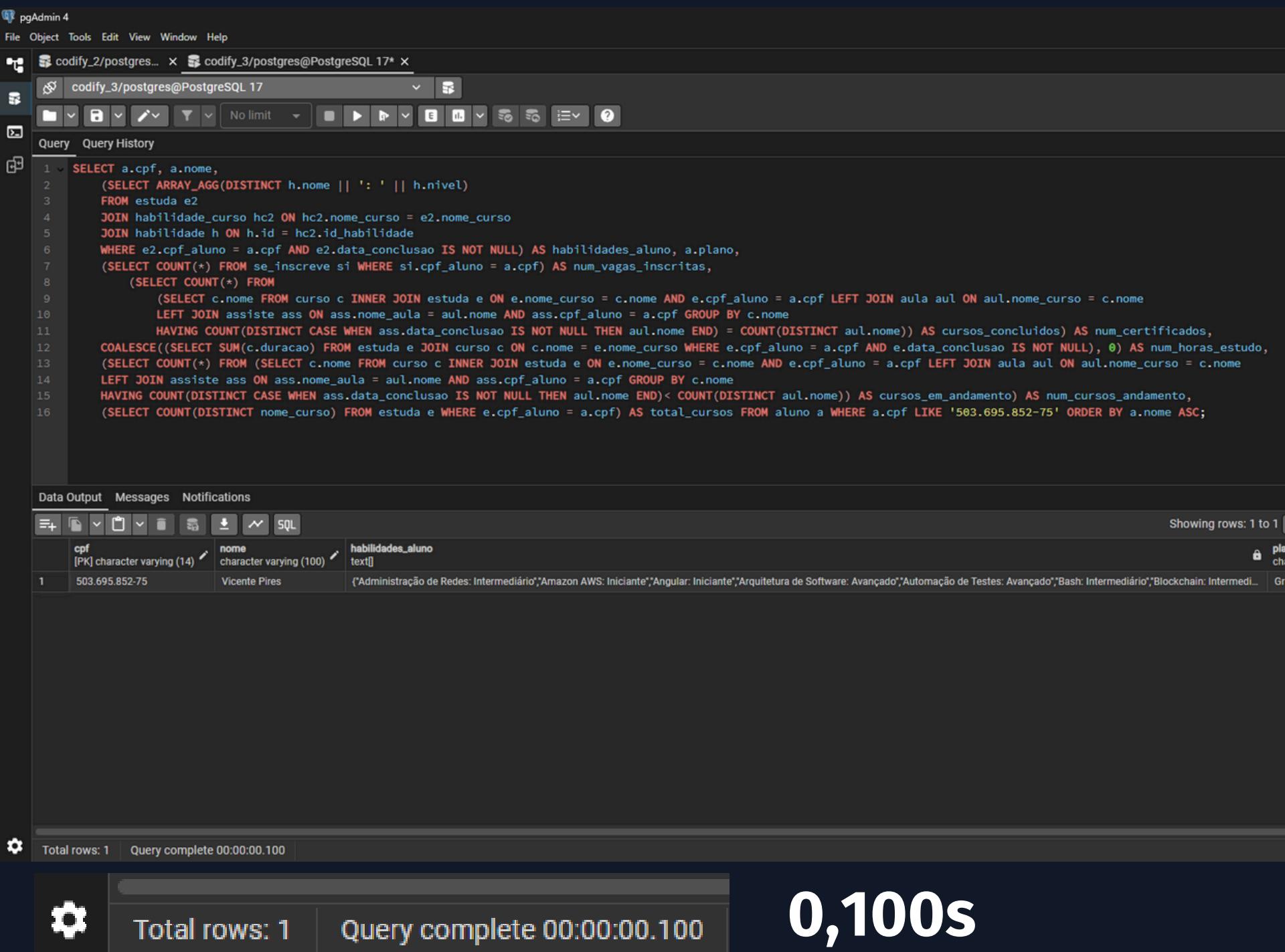
**Particionamento de Tabelas com muitos registros:** faz com que uma query consulte apenas uma parte dos registros da tabela, em vez de ler a tabela inteira. Dividimos as tabelas Estuda, Assiste e Se\_Inscreve em 4 partições.

```
-- Tabela de Relacionamento Assiste (Aluno-Aula)
CREATE TABLE Assiste (
    cpf_aluno      VARCHAR(14)      NOT NULL,
    nome_aula       VARCHAR(100)     NOT NULL,
    data_conclusao DATE,
    PRIMARY KEY (cpf_aluno, nome_aula),
    FOREIGN KEY (cpf_aluno)   REFERENCES Aluno(cpf),
    FOREIGN KEY (nome_aula)    REFERENCES Aula(nome)
)PARTITION BY HASH (cpf_aluno);

CREATE TABLE assiste_p1 PARTITION OF Assiste FOR VALUES WITH (MODULUS 4, REMAINDER 0);
CREATE TABLE assiste_p2 PARTITION OF Assiste FOR VALUES WITH (MODULUS 4, REMAINDER 1);
CREATE TABLE assiste_p3 PARTITION OF Assiste FOR VALUES WITH (MODULUS 4, REMAINDER 2);
CREATE TABLE assiste_p4 PARTITION OF Assiste FOR VALUES WITH (MODULUS 4, REMAINDER 3);
```

# Testes Otimizações

## Não Otimizado



```
pgAdmin 4
File Object Tools Edit View Window Help
codify_2/postgres@PostgreSQL 17* codify_3/postgres@PostgreSQL 17*
Query History
Query
SELECT a.cpf, a.nome,
(SELECT ARRAY_AGG(DISTINCT h.nome || ': ' || h.nivel)
FROM estuda e2
JOIN habilidade_curso hc2 ON hc2.nome_curso = e2.nome_curso
JOIN habilidade h ON h.id = hc2.id_habilidade
WHERE e2.cpf_aluno = a.cpf AND e2.data_conclusao IS NOT NULL) AS habilidades_aluno, a.plano,
(SELECT COUNT(*) FROM se_inscreve si WHERE si.cpf_aluno = a.cpf) AS num_vagas_inscritas,
(SELECT COUNT(*) FROM
(SELECT c.nome FROM curso c INNER JOIN estuda e ON e.nome_curso = c.nome AND e.cpf_aluno = a.cpf LEFT JOIN aula aul ON aul.nome_curso = c.nome
LEFT JOIN assiste ass ON ass.nome_aula = aul.nome AND ass.cpf_aluno = a.cpf GROUP BY c.nome
HAVING COUNT(DISTINCT CASE WHEN ass.data_conclusao IS NOT NULL THEN aul.nome END) = COUNT(DISTINCT aul.nome)) AS cursos_concluidos) AS num_certificados,
COALESCE((SELECT SUM(c.duracao) FROM estuda e JOIN curso c ON c.nome = e.nome_curso WHERE e.cpf_aluno = a.cpf AND e.data_conclusao IS NOT NULL), 0) AS num_horas_estudo,
(SELECT COUNT(*) FROM (SELECT c.nome FROM curso c INNER JOIN estuda e ON e.nome_curso = c.nome AND e.cpf_aluno = a.cpf LEFT JOIN aula aul ON aul.nome_curso = c.nome
LEFT JOIN assiste ass ON ass.nome_aula = aul.nome AND ass.cpf_aluno = a.cpf GROUP BY c.nome
HAVING COUNT(DISTINCT CASE WHEN ass.data_conclusao IS NOT NULL THEN aul.nome END) < COUNT(DISTINCT aul.nome)) AS cursos_em_andamento) AS num_cursos_andamento,
(SELECT COUNT(DISTINCT nome_curso) FROM estuda e WHERE e.cpf_aluno = a.cpf) AS total_cursos FROM aluno a WHERE a.cpf LIKE '503.695.852-75' ORDER BY a.nome ASC;
```

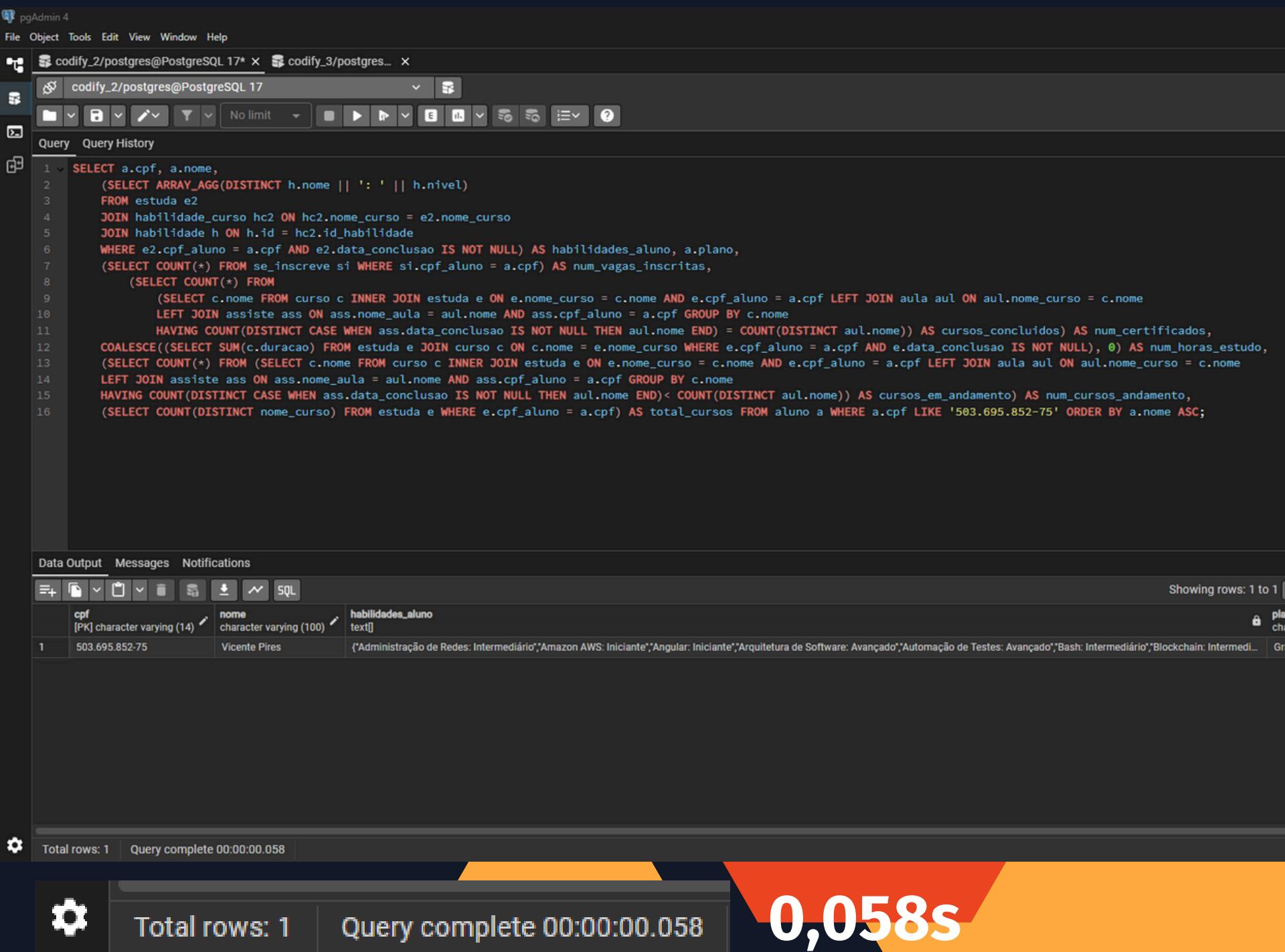
Data Output Messages Notifications

cpf	nome	habilidades_aluno
503.695.852-75	Vicente Pires	('Administração de Redes: Intermediário','Amazon AWS: Iniciante','Angular: Iniciante','Arquitetura de Software: Avançado','Automação de Testes: Avançado','Bash: Intermediário','Blockchain: Intermediário','C# e .NET: Intermediário','Cloud Computing: Intermediário','Data Science com Python: Intermediário','Docker e Containerização: Intermediário','E-commerce: Intermediário','Engenharia de Software: Avançado','Frontend: Avançado','Golang: Intermediário','Java: Avançado','Machine Learning: Intermediário','MongoDB: Intermediário','Node.js: Intermediário','Python: Avançado','React: Avançado','SQL: Avançado','Spring Boot: Intermediário','Vue.js: Intermediário')

Total rows: 1 Query complete 00:00:00.100

0,100s

## Otimizado



```
pgAdmin 4
File Object Tools Edit View Window Help
codify_2/postgres@PostgreSQL 17* codify_3/postgres@PostgreSQL 17*
Query History
Query
SELECT a.cpf, a.nome,
(SELECT ARRAY_AGG(DISTINCT h.nome || ': ' || h.nivel)
FROM estuda e2
JOIN habilidade_curso hc2 ON hc2.nome_curso = e2.nome_curso
JOIN habilidade h ON h.id = hc2.id_habilidade
WHERE e2.cpf_aluno = a.cpf AND e2.data_conclusao IS NOT NULL) AS habilidades_aluno, a.plano,
(SELECT COUNT(*) FROM se_inscreve si WHERE si.cpf_aluno = a.cpf) AS num_vagas_inscritas,
(SELECT COUNT(*) FROM
(SELECT c.nome FROM curso c INNER JOIN estuda e ON e.nome_curso = c.nome AND e.cpf_aluno = a.cpf LEFT JOIN aula aul ON aul.nome_curso = c.nome
LEFT JOIN assiste ass ON ass.nome_aula = aul.nome AND ass.cpf_aluno = a.cpf GROUP BY c.nome
HAVING COUNT(DISTINCT CASE WHEN ass.data_conclusao IS NOT NULL THEN aul.nome END) = COUNT(DISTINCT aul.nome)) AS cursos_concluidos) AS num_certificados,
COALESCE((SELECT SUM(c.duracao) FROM estuda e JOIN curso c ON c.nome = e.nome_curso WHERE e.cpf_aluno = a.cpf AND e.data_conclusao IS NOT NULL), 0) AS num_horas_estudo,
(SELECT COUNT(*) FROM (SELECT c.nome FROM curso c INNER JOIN estuda e ON e.nome_curso = c.nome AND e.cpf_aluno = a.cpf LEFT JOIN aula aul ON aul.nome_curso = c.nome
LEFT JOIN assiste ass ON ass.nome_aula = aul.nome AND ass.cpf_aluno = a.cpf GROUP BY c.nome
HAVING COUNT(DISTINCT CASE WHEN ass.data_conclusao IS NOT NULL THEN aul.nome END) < COUNT(DISTINCT aul.nome)) AS cursos_em_andamento) AS num_cursos_andamento,
(SELECT COUNT(DISTINCT nome_curso) FROM estuda e WHERE e.cpf_aluno = a.cpf) AS total_cursos FROM aluno a WHERE a.cpf LIKE '503.695.852-75' ORDER BY a.nome ASC;
```

Data Output Messages Notifications

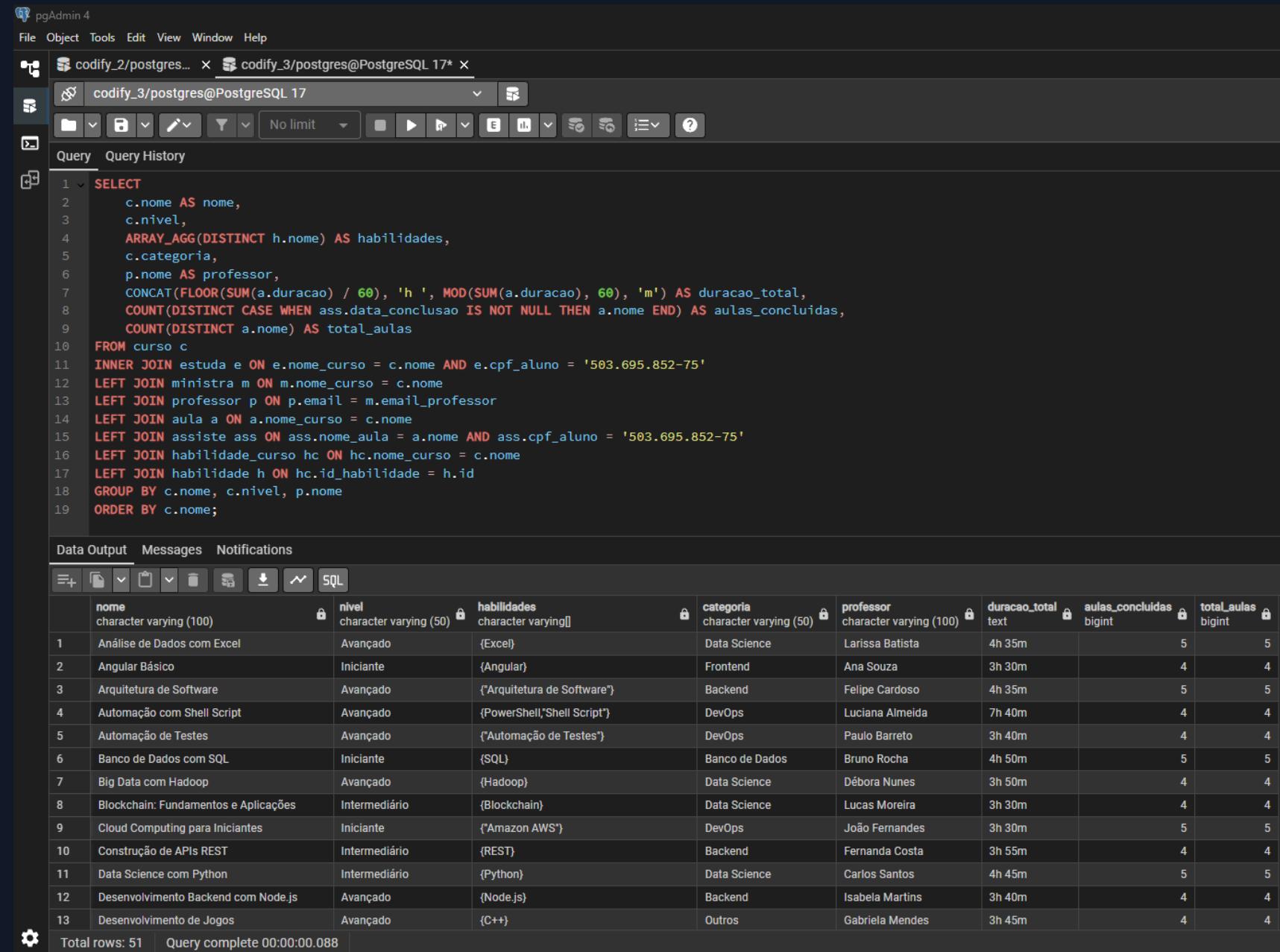
cpf	nome	habilidades_aluno
503.695.852-75	Vicente Pires	('Administração de Redes: Intermediário','Amazon AWS: Iniciante','Angular: Iniciante','Arquitetura de Software: Avançado','Automação de Testes: Avançado','Bash: Intermediário','Blockchain: Intermediário','C# e .NET: Intermediário','Cloud Computing: Intermediário','Data Science com Python: Intermediário','Docker e Containerização: Intermediário','E-commerce: Intermediário','Engenharia de Software: Avançado','Frontend: Avançado','Golang: Intermediário','Java: Avançado','Machine Learning: Intermediário','MongoDB: Intermediário','Node.js: Intermediário','Python: Avançado','React: Avançado','SQL: Avançado','Spring Boot: Intermediário','Vue.js: Intermediário')

Total rows: 1 Query complete 00:00:00.058

0,058s

# Testes Otimizações

## Não Otimizado



The screenshot shows the pgAdmin 4 interface with two tabs: 'codify\_2/postgres@PostgreSQL 17\*' and 'codify\_3/postgres@PostgreSQL 17'. The left tab contains the unoptimized SQL query. The right tab shows the results of the query, displaying 51 rows of course information including name, level, skills, category, professor, duration, completed classes, and total classes. The results are presented in a standard table format.

```
1 SELECT
2     c.nome AS nome,
3     c.nivel,
4     ARRAY_AGG(DISTINCT h.nome) AS habilidades,
5     c.categoria,
6     p.nome AS professor,
7     CONCAT(FLOOR(SUM(a.duracao) / 60), 'h ', MOD(SUM(a.duracao), 60), 'm') AS duracao_total,
8     COUNT(DISTINCT CASE WHEN ass.data_conclusao IS NOT NULL THEN a.nome END) AS aulas_concluidas,
9     COUNT(DISTINCT a.nome) AS total_aulas
10    FROM curso c
11   INNER JOIN estuda e ON e.nome_curso = c.nome AND e.cpf_aluno = '503.695.852-75'
12  LEFT JOIN ministra m ON m.nome_curso = c.nome
13  LEFT JOIN professor p ON p.email = m.email_professor
14  LEFT JOIN aula a ON a.nome_curso = c.nome
15  LEFT JOIN assiste ass ON ass.nome_aula = a.nome AND ass.cpf_aluno = '503.695.852-75'
16  LEFT JOIN habilidade_curso hc ON hc.nome_curso = c.nome
17  LEFT JOIN habilidade h ON hc.id_habilidade = h.id
18 GROUP BY c.nome, c.nivel, p.nome
19 ORDER BY c.nome;
```

	nome	nível	habilidades	categoria	professor	duracao_total	aulas_concluidas	total_aulas
1	Análise de Dados com Excel	Avançado	{Excel}	Data Science	Larissa Batista	4h 35m	5	5
2	Angular Básico	Iniciante	{Angular}	Frontend	Ana Souza	3h 30m	4	4
3	Arquitetura de Software	Avançado	{"Arquitetura de Software"}	Backend	Felipe Cardoso	4h 35m	5	5
4	Automação com Shell Script	Avançado	{PowerShell,"Shell Script"}	DevOps	Luciana Almeida	7h 40m	4	4
5	Automação de Testes	Avançado	{"Automação de Testes"}	DevOps	Paulo Barreto	3h 40m	4	4
6	Banco de Dados com SQL	Iniciante	{SQL}	Banco de Dados	Bruno Rocha	4h 50m	5	5
7	Big Data com Hadoop	Avançado	{Hadoop}	Data Science	Débora Nunes	3h 50m	4	4
8	Blockchain: Fundamentos e Aplicações	Intermediário	{Blockchain}	Data Science	Lucas Moreira	3h 30m	4	4
9	Cloud Computing para Iniciantes	Iniciante	{"Amazon AWS"}	DevOps	João Fernandes	3h 30m	5	5
10	Construção de APIs REST	Intermediário	{REST}	Backend	Fernanda Costa	3h 55m	4	4
11	Data Science com Python	Intermediário	{Python}	Data Science	Carlos Santos	4h 45m	5	5
12	Desenvolvimento Backend com Node.js	Avançado	{Node.js}	Backend	Isabela Martins	3h 40m	4	4
13	Desenvolvimento de Jogos	Avançado	{C++}	Outros	Gabriela Mendes	3h 45m	4	4

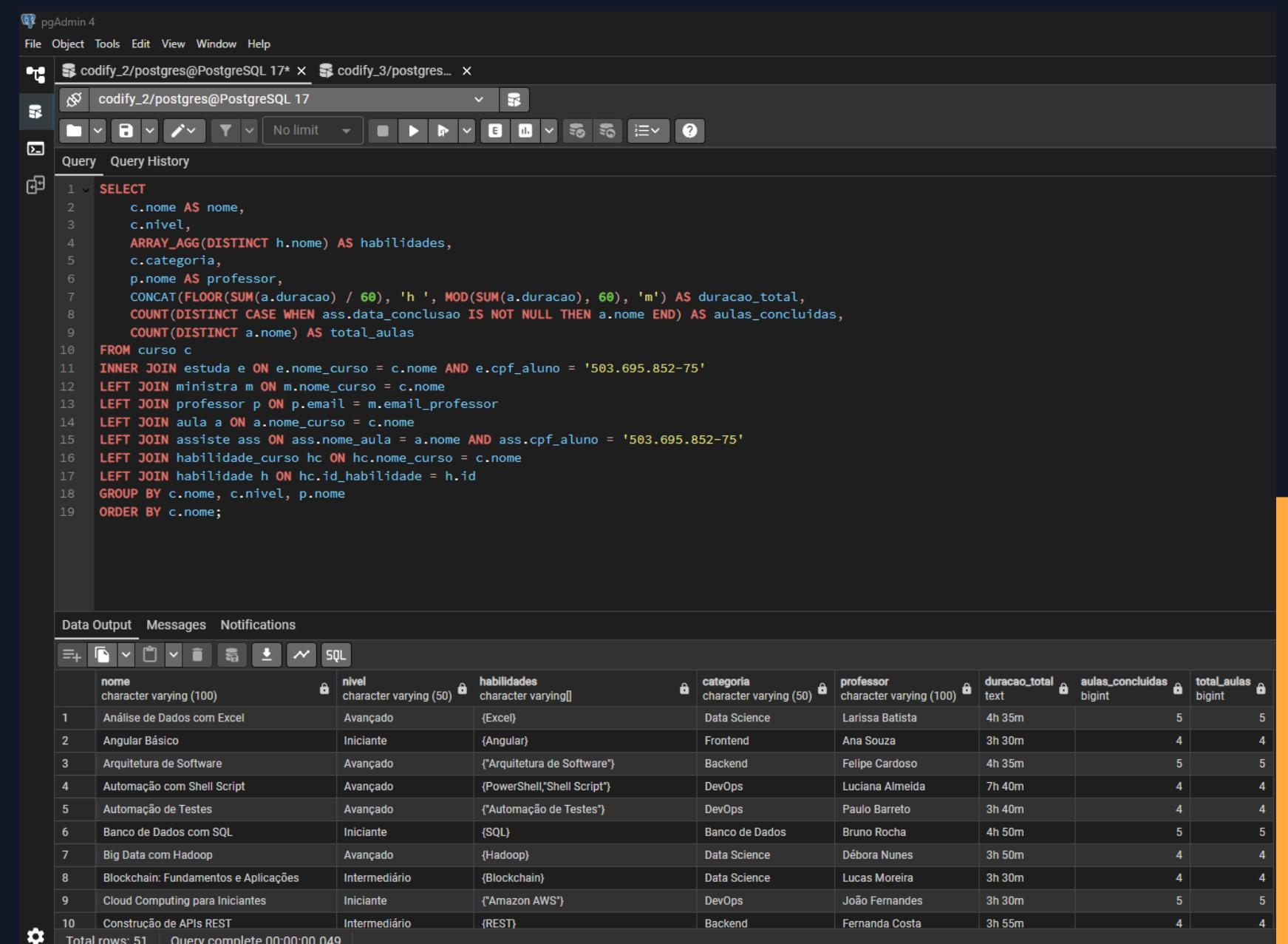


Total rows: 51

Query complete 00:00:00.088

0,088s

## Otimizado



The screenshot shows the pgAdmin 4 interface with two tabs: 'codify\_2/postgres@PostgreSQL 17\*' and 'codify\_3/postgres@PostgreSQL 17'. The left tab contains the optimized SQL query. The right tab shows the results of the query, displaying 51 rows of course information. The results are presented in a standard table format, identical to the unoptimized version.

```
1 SELECT
2     c.nome AS nome,
3     c.nivel,
4     ARRAY_AGG(DISTINCT h.nome) AS habilidades,
5     c.categoria,
6     p.nome AS professor,
7     CONCAT(FLOOR(SUM(a.duracao) / 60), 'h ', MOD(SUM(a.duracao), 60), 'm') AS duracao_total,
8     COUNT(DISTINCT CASE WHEN ass.data_conclusao IS NOT NULL THEN a.nome END) AS aulas_concluidas,
9     COUNT(DISTINCT a.nome) AS total_aulas
10    FROM curso c
11   INNER JOIN estuda e ON e.nome_curso = c.nome AND e.cpf_aluno = '503.695.852-75'
12  LEFT JOIN ministra m ON m.nome_curso = c.nome
13  LEFT JOIN professor p ON p.email = m.email_professor
14  LEFT JOIN aula a ON a.nome_curso = c.nome
15  LEFT JOIN assiste ass ON ass.nome_aula = a.nome AND ass.cpf_aluno = '503.695.852-75'
16  LEFT JOIN habilidade_curso hc ON hc.nome_curso = c.nome
17  LEFT JOIN habilidade h ON hc.id_habilidade = h.id
18 GROUP BY c.nome, c.nivel, p.nome
19 ORDER BY c.nome;
```

	nome	nível	habilidades	categoria	professor	duracao_total	aulas_concluidas	total_aulas
1	Análise de Dados com Excel	Avançado	{Excel}	Data Science	Larissa Batista	4h 35m	5	5
2	Angular Básico	Iniciante	{Angular}	Frontend	Ana Souza	3h 30m	4	4
3	Arquitetura de Software	Avançado	{"Arquitetura de Software"}	Backend	Felipe Cardoso	4h 35m	5	5
4	Automação com Shell Script	Avançado	{PowerShell,"Shell Script"}	DevOps	Luciana Almeida	7h 40m	4	4
5	Automação de Testes	Avançado	{"Automação de Testes"}	DevOps	Paulo Barreto	3h 40m	4	4
6	Banco de Dados com SQL	Iniciante	{SQL}	Banco de Dados	Bruno Rocha	4h 50m	5	5
7	Big Data com Hadoop	Avançado	{Hadoop}	Data Science	Débora Nunes	3h 50m	4	4
8	Blockchain: Fundamentos e Aplicações	Intermediário	{Blockchain}	Data Science	Lucas Moreira	3h 30m	4	4
9	Cloud Computing para Iniciantes	Iniciante	{"Amazon AWS"}	DevOps	João Fernandes	3h 30m	5	5
10	Construção de APIs REST	Intermediário	{REST}	Backend	Fernanda Costa	3h 55m	4	4



Total rows: 51

Query complete 00:00:00.049

0,049s

# Backend

# Criação do Backend

- Programação feita em Python utilizando o framework Flask.
- Criação das rotas e do database.
- Uso de IAG (Google Gemini) para auxílio nas consultas SQL e na criação do código.



# Frontend

# Criação do Frontend

## AIG

- Github Copilot

## Uso de IAG

- Auxiliar na construção do código.
- Auxiliar na criação do designe das páginas.

## Tecnologias

- NextJs
- React
- Typescript



# Telas:

Domine a programação do zero ao avançado

Resultados que Falam por Si

Cursos em Destaque

Weekly Dev Tips

Navegação: Home, Cursos, Tecnologias, Suporte, Sobre Nós

Informações legais: Termos de Uso, Política de Privacidade, PDR, Desenvolvedores, Contato, Redes Sociais

Criar sua conta

Nome completo \*

E-mail \*

Data de nascimento \*

Senha \*

Senha de segurança \*

Continuar

Entrar na sua conta

Entrar

Esqueci minha senha

Mais informações Criar conta

Entrar na sua conta

Entrar

Esqueci minha senha

Mais informações Criar conta

Weekly Dev Tips

Subscrever

Olá, Brayan Teixeira! %

Progresso dos Cursos

7 Módulos concluídos

510 Minutos de estudo

3 Certificados

Plano Rodadão Grátis

6 Cursos

Blockchain: Fundamentos e Aplicações

Excel Avançado

JavaScript

Kubernetes

Processamento de Dados com Apache Spark

Minhas Habilidades

Blockchain Intermediário

Excel Intermediário

JavaScript Intermediário

Kubernetes Intermediário

Processamento de Dados com Apache Spark Intermediário

Meus Cursos

# Telas:

The page displays two plan options: 'Plano Gratuito' (Free Plan) at R\$ 0 and 'Plano Pro' at R\$ 49. Both plans include features like resume builder, job search, and access to courses. A 'Começar Agora' (Start Now) button is present.

**Por que escolher nossa plataforma?**

- Cursos Completos
- Projetos Práticos
- Mentoria Especializada
- Certificados Reconhecidos
- Preparação para Entrevistas
- Conexão com o Mercado

**Perguntas Frequentes**

- Pode mudar de plano a qualquer momento?
- O que acontece se eu cancelar minha assinatura?
- Ou certificados são reconhecidos pelo mercado?
- Como funciona a mentoria?
- Pode acessar o conteúdo offline?
- Existe garantia de satisfação?

A footer link: [Ainda não disponibilizamos em outras cidades](#)

The page lists various companies as partners, each with a brief description and a 'Ver Mais' (View More) button. Examples include Desenvolvedor Front-end, Desenvolvedor Back-end, Clínica de Dados, Arquiteta de Qualidade, Desenvolvedor de UX, and Engenheiro de Software.

The dark-themed version of the 'Empresas Parceiras' page shows the same company profiles and layout as the light version.



# Obrigado!

