

UNIVERSIDADE DO OESTE DE SANTA CATARINA

BRUNO MARLON SCHMIDT, SERGIO CHRISTOFOLI, ALAN CRISTIANO BENDER

COVTRACKER: Sistema de Monitoramento de Casos e Estatísticas do Coronavírus

São Miguel do Oeste – SC

2021

BRUNO MARLON SCHMIDT, SERGIO CHRISTOFOLI, ALAN CRISTIANO BENDER

COVTRACKER: Sistema de Monitoramento de Casos e Estatísticas do Coronavírus

Relatório de formação apresentado ao Curso de Ciência da Computação, Área das Ciências Exatas, da Universidade do Oeste de Santa Catarina como requisito parcial à obtenção de nota para trabalho final no componente de Banco de Dados I.

Orientador: Prof. Roberson Junior Fernandes Alves

São Miguel do Oeste – SC

2021

## SUMÁRIO

<b>1. INTRODUÇÃO.....</b>	<b>3</b>
<b>2. REQUISITOS.....</b>	<b>4</b>
<b>3. MODELO RELACIONAL.....</b>	<b>6</b>
3.1. DICIONÁRIO DE DADOS.....	7
<b>4. BANCO DE DADOS.....</b>	<b>9</b>
4.1. RELATÓRIOS.....	9
<b>5. CONSIDERAÇÕES FINAIS.....</b>	<b>10</b>
<b>REFERÊNCIAS.....</b>	<b>11</b>

## 1. INTRODUÇÃO

O objetivo deste estudo é, através do conhecimento de banco de dados e sua manipulação, apresentar um sistema automatizado para o monitoramento dos casos de COVID 19 pelo Brasil. Utilizando diversas ferramentas como DBeaver e Visual Paradigm, o PostgreSQL como SGBD, e a linguagem SQL, foi feito o desenvolvimento deste projeto e, no desenrolar deste trabalho, será explicitado as suas mais variadas funções que foram utilizadas, bem como os conceitos de manipulação de banco de dados explorados.

## 2. REQUISITOS

Para o desenvolvimento do banco de dados foram considerados os seguintes requisitos que serviram de “bússola” para o desenvolvimento do projeto:

- É necessário cadastrar as empresas com CNPJ e demais dados. O paciente está associado a uma empresa;
- Cadastrar os dados em geral, gerar relatórios e gráficos, além de emitir avisos a cada 48h para acompanhamento dos casos;
- Para cada paciente/usuário deve ser aplicado um questionário para abordar desde dados pessoais (peso, altura) a histórico de comorbidades e sintomas do momento;
- É necessário informar a empresa a qual o paciente está vinculado;
- Também devem ser registrados os tipos de usuário, sejam eles administradores, líderes, pacientes, etc;
- Para cada usuário devem ser coletadas informações de geolocalização (Cidade, etc);
- Casos considerados suspeitos para a COVID-19 têm as informações armazenadas e a evolução clínica acompanhada de acordo com os requisitos do Ministério da Saúde;
- Febre, tosse, falta de ar, dor no corpo, dor de garganta, calafrio, dor muscular, congestão nasal e coriza são sintomas que devem ser informados no aplicativo, com detalhamento de intensidade (pouco, moderado ou constante), sempre que alterações forem observadas;
- Ao comparar esses dados e detectar sinais de criticidade, no cadastro destes sintomas, o sistema deve emitir alertas ao usuário/paciente para procurar o serviço

de pronto atendimento;

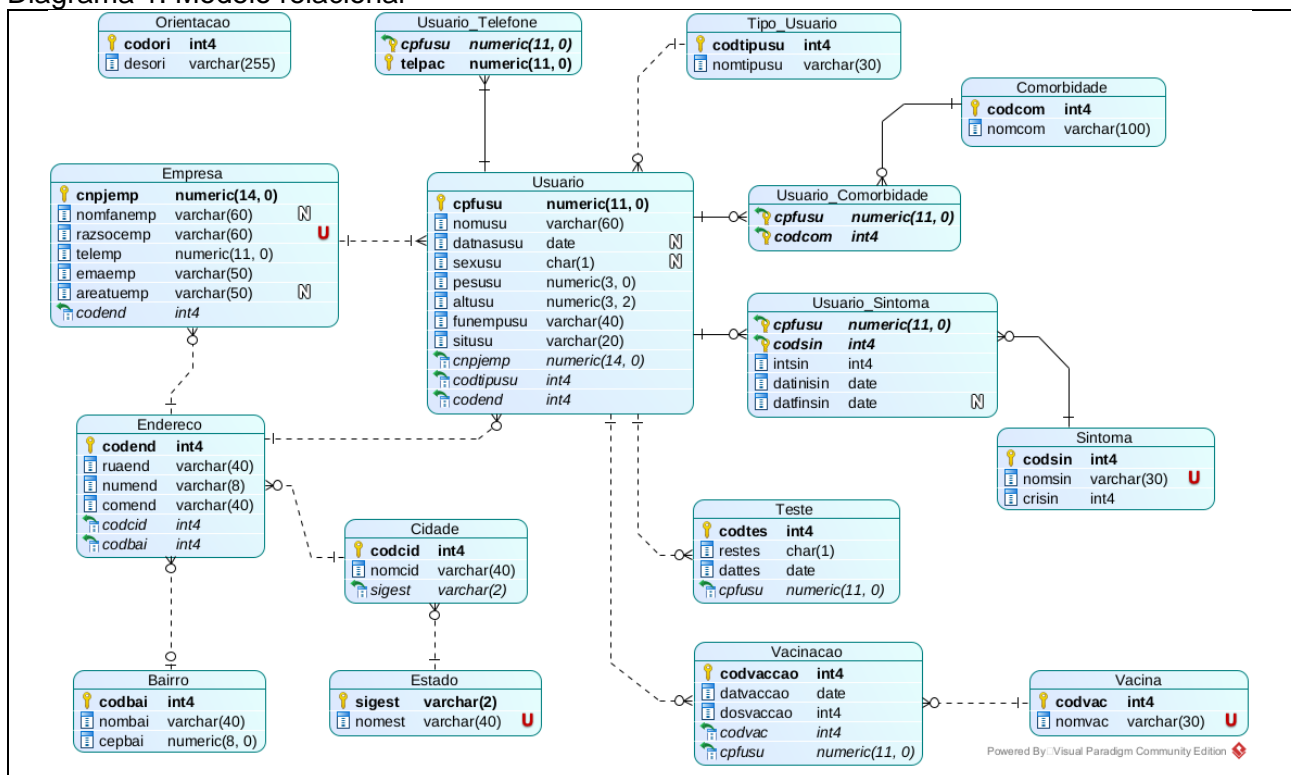
- O sistema deve permitir consultas por cidade, região, etc;
- O sistema deve permitir emitir relatórios diversificados e com estatísticas de casos por localização;
- O sistema deve permitir importar dados nacionais sobre covid de outras plataformas nos formatos CSV, XLS, XML, JSON;
- O sistema deve permitir o cadastro de orientações sobre a doença caso o usuário tenha dúvidas, algo como uma ajuda;
- O sistema deve controlar o nível de acesso dos diversos tipos de usuário;

Como este projeto lida apenas com a parte voltada à banco de dados, alguns dos requisitos que dependem de um backend não puderam ser satisfeitos.

### 3. MODELO RELACIONAL

O Diagrama 1, a seguir, apresenta o modelo relacional do nosso projeto, feito no programa Visual Paradigm, nele pode-se observar visualmente a interação lógica entre os diversos componentes que formam a estrutura lógica do projeto.

Diagrama 1: Modelo relacional



Fonte: Os autores (2021).

Como ponto central do modelo, temos a tabela “Usuario”, que como o nome sugere, armazena diversos dados relativos à identificação e situação do usuário em relação ao COVID 19. A partir de então, diversas outras tabelas surgem para entrar mais detalhadamente em alguns destes dados do usuário, como as tabelas “Empresa”, “Teste”, “Usuario\_Sintoma”, “Usuario\_Comorbidade”, “Tipo\_Usuario” e “Usuario\_Telefone”.










No geral, a ramificação entre várias tabelas ocorre para que se evite problemas na organização do modelo que quebrem alguma das três regras de normalização de dados, garantindo assim, consistência no projeto e nas suas representações.

Vale ressaltar que a nomenclatura utilizada para nomeação dos atributos consiste na utilização das três primeiras letras de cada palavra.

### 3.1. DICIONÁRIO DE DADOS








Segue nas figuras 1 e 2 o dicionário de dados com as informações sobre as tabelas e atributos criadas no modelo relacional.

Figura 1: Dicionário de dados.

1. Data Dictionary						
Entity Name	Entity Description					
Column Name	Column Description	Data Type	Length	Primary Key	Nullable	Unique
 Bairro	Cadastro do bairro					
cepbai	Cep do bairro /ou cidade	numeric	8	false	false	false
codbai	Código do bairro	int4	10	true	false	false
nombai	Nome do bairro	varchar	40	false	false	false
 Cidade	Cadastro da Cidade					
codcid	Código da cidade	int4	10	true	false	false
nomcid	Nome da cidade	varchar	40	false	false	false
sigest		varchar	2	false	false	false
 Comorbidade	Cadastro de comorbidades					
codcom	Código da comorbidade	int4	0	true	false	false
nomcom	Nome da comorbidade	varchar	100	false	false	true
 Empresa	Empresa Responsavel pelo paciente					
areatuemp	Área de atuação da empresa	varchar	50	false	true	false
cnpjemp	CNPJ da empresa	numeric	14	true	false	false
codend		int4	10	false	false	false
emaemp	Email da empresa	varchar	50	false	false	false
nomfanemp	Nome Fantasia da Empresa	varchar	60	false	true	false
razsoemp	Razão Social da Empresa	varchar	60	false	false	true
temp	Cadastro do telefone da empresa	numeric	11	false	false	false
 Endereco	Cadastro do endereço					
codbai		int4	10	false	true	false
codcid		int4	10	false	false	false
codend	Código do endereço	int4	10	true	false	false
comend	Complemento do endereço	varchar	40	false	true	false
numend	Número do endereço	varchar	8	false	true	false
ruaend	Rua do endereço	varchar	40	false	false	false
 Estado	Cadastro do Estado					
nomest	Nome do estado	varchar	40	false	false	true
sigest	Sigla do estado	varchar	2	true	false	false
 Orientacao	Cadastro de orientações					
codori	Código da orientação	int4	10	true	false	false
desori	Descrição da orientação	varchar	255	false	false	false
 Sintoma	Cadastro do sintoma					
codsin	Código do Sintoma	int4	10	true	false	false
crisin	Criticidade do sintoma (1 = Baixa, 2 = Média, 3 = Alta)	int4	1	false	false	false
nomsin	Nome do Sintoma	varchar	30	false	false	true
 Teste	Cadastro do teste de covid.					
codtes	Código do teste	int4	10	true	false	false
cpfusu		numeric	11	false	false	false
dattes	Data da realização do teste de covid	date	0	false	false	false
restes	Resultado do teste de covid. (P = Positivo, N = Negativo)	char	1	false	false	false

Fonte: Os autores (2021).

Figura 2: Dicionário de dados (continuação)

	Tipo_Usuario						
	codtipusu	Código do tipo de usuário	int4	1	true	false	false
	nomtipusu	Nome do tipo de usuário (Paciente, Administrador...)	varchar	30	false	false	true
	Usuario	Cadastro do usuário					
	altusu	Altura do usuario em metros	numeric	3.2	false	false	false
	cnpiemp		numeric	14	false	false	false
	codend		int4	10	false	false	false
	codtipusu		int4	1	false	false	false
	cpfusu	CPF do usuário	numeric	11	true	false	false
	datnasusu	Data de nascimento do usuario.	date	0	false	true	false
	funempusu	Função do usuário na empresa	varchar	40	false	false	false
	nomusu	Nome do Usuário.	varchar	60	false	false	false
	pesusu	Peso do usuario	numeric	3	false	false	false
	sexusu	Sexo do usuário: (M = Masculino, F = Feminino).	char	1	false	true	false
	Usuario_Comorbidade						
	codcom		int4	0	true	false	false
	cpfusu		numeric	11	true	false	false
	Usuario_Sintoma	Relacionamento entre Paciente e Sintoma					
	codsin		int4	10	true	false	false
	cpfusu		numeric	11	true	false	false
	datfinsin	Data final do sintoma	date	0	false	true	false
	datinisin	Data de início do sintoma	date	0	false	false	false
	intsin	Intensidade do sintoma (1 = Baixa, 2 = Média, 3 = Alta)	int4	1	false	false	false
	Usuario_Telefone	Cadastro de telefones do usuário.					
	cpfusu		numeric	11	true	false	false
	telpac	Telefone do paciente	numeric	11	true	false	false
	Vacina	Cadastro da vacina					
	codvac	Código da vacina	int4	10	true	false	false
	nomvac	Nome da vacina	varchar	30	false	false	true
	Vacinacao	Cadastro da vacinação					
	codvac		int4	10	false	false	false
	codvaccac	Código da vacinação	int4	10	true	false	false
	cpfusu		numeric	11	false	false	false
	datvaccac	Data da vacinação	date	0	false	false	false
	dosvaccac	Dose da vacinação (1 ou 2)	int4	1	false	false	false

Fonte: Os autores (2021)

## 4. BANCO DE DADOS

Após a realização e normalização do modelo relacional fazendo o uso da ferramenta Visual Paradigm, foi seguido adiante no projeto com a criação dos scripts. Para isso foi utilizada a ferramenta Dbeaver, a linguagem SQL (Structured Query Language – Linguagem de consulta estruturada), e o sistema gerenciador de banco de dados PostgreSQL. Primeiramente foi gerado o script de criação da base de dados, e na sequência criado um script para inserção de dados na base.

### 4.1. RELATÓRIOS



Na base de dados foram criados quatro relatórios baseados respectivamente nos seguintes requisitos:

- 1) Relacione o código e nome de pacientes com idades entre 60 e 70 anos, que apresentaram febre. Relacione a consulta em ordem ascendente de nome;
- 2) Relacione o nome do paciente, nome da cidade de residência de pacientes do sexo feminino, residentes nos municípios de Maravilha, Descanso, Pinhalzinho, Chapecó e Itapiranga que apresentaram sintomas e não foram positivados com covid. Relacione o relatório pelo nome da cidade ascendente e o nome do paciente descendente;
- 3) Relacione o código da cidade, nome da cidade, quantidade de casos suspeitos de covid para todas as cidades. Ordene o relatório da cidade com mais casos suspeitos para a cidade com menos casos suspeitos;
- 4) Relacione a idade e quantidade de casos positivos de covid por idade, registrados no período agosto a outubro de 2020. Ordene o relatório pela idade com mais casos para a idade com menos casos.

Para a criação das views foram utilizados diversos comandos da linguagem SQL, como selects, inner e left joins, cláusulas where, order by e group by. As quatro views criadas foram nomeadas como vw\_rel\_1, vw\_rel\_2, vw\_rel\_3 e vw\_rel\_4 para os quatro respectivos relatórios.

Segue abaixo na Figura 3 o script SQL da criação dos relatórios:

Figura 3: Script de criação das views do relatório

```

1 CREATE VIEW vw_rel_1 AS
2 SELECT us.cpfusu AS CPF,nomusu AS Nome
3 FROM usuario u
4 INNER JOIN usuario_sintoma us ON u.cpfusu = us.cpfusu
5 INNER JOIN sintoma s ON s.codsin = us.codsin AND nomsin = 'Febre'
6 WHERE ((CURRENT_DATE - u.datnasusu)/365) BETWEEN 60 AND 70
7 ORDER BY nomusu ASC;
8
9 COMMENT ON VIEW vw_rel_1 IS 'Código e nome de pacientes com entre 60 e 70 anos que apresentaram febre. Ordem ascendente de nome'
10
11 CREATE VIEW vw_rel_2 AS
12 SELECT nomusu AS nome, nomcid AS cidade
13 FROM usuario u
14 INNER JOIN endereco e ON u.codend = e.codend
15 INNER JOIN cidade c ON e.codcid = c.codcid
16 AND nomcid IN ('Descanso', 'Pinhalzinho', 'Maravilha', 'Chapecó', 'Itapiranga')
17 WHERE u.sexusu = 'F' AND NOT u.cpfusu IN (
18     SELECT u.cpfusu FROM teste t
19     INNER JOIN usuario u ON t.cpfusu = u.cpfusu AND t.restes = 'P'
20 )
21 ORDER BY nomcid ASC, nomusu DESC;
22
23 COMMENT ON VIEW vw_rel_2 IS 'Nome do paciente e cidade de pacientes do sexo feminino residentes de Maravilha, Descanso, Pinhalzi
24
25 CREATE VIEW vw_rel_3 as
26 select c.codcid as codigo_cidade, nomcid as cidade , count(cpfusu) as suspeitas
27 from cidade c
28 left join endereco e on c.codcid = e.codcid
29 left join usuario u on e.codend = u.codend
30 and u.cpfusu in (
31     select cpfusu
32     from usuario_sintoma
33     inner join sintoma s on s.codsin = usuario_sintoma.codsin
34     group by cpfusu
35     having sum(crisin) >= 2
36 )
37 group by c.codcid
38 order by suspeitas desc, c.codcid asc;
39
40 COMMENT ON VIEW vw_rel_3 IS 'Relatório de casos suspeitos por cidade.';
41
42 CREATE VIEW vw_rel_4 AS
43 SELECT (((CURRENT_DATE - u.datnasusu)/365) AS idade, count(u.cpfusu) AS casos
44 FROM teste t
45 INNER JOIN usuario u ON t.cpfusu = u.cpfusu AND t.restes = 'P'
46 GROUP BY idade
47 ORDER BY casos DESC;
48
49 COMMENT ON VIEW vw_rel_4 IS 'Relatório de casos por idade';
50

```

Fonte: Os autores (2021)

## **5. CONSIDERAÇÕES FINAIS**

Por fim, é seguro dizer que os objetivos do estudo foram alcançados, já que, a demonstração de sua funcionalidade foi claramente exposta no decorrer do presente trabalho, sendo possível ver claramente como o manuseamento que foi feito com os dados dos usuários serviu para o monitoramento dos casos de COVID 19. Sua utilização se mostrou sólida para os parâmetros exigidos, sendo possível enxergar nitidamente o papel fundamental que o conhecimento em teoria de banco de dados exerceu, assim como o conhecimento das diversas ferramentas citadas. De fato, ainda há margem para melhorias, para novas implementações e para ampliações, contudo, isso não anula os resultados obtidos com o que foi feito até o momento.

## REFERÊNCIAS

ALVES, Roberson J. F. **Apostila de Banco de Dados**. São Miguel do Oeste: Unoesc, 2021. Material didático.

ROVER, Ardinete; MELLO, Regina Oneda. **Normas da ABNT**: orientações para a produção científica. 1 ed. Joaçaba: Unoesc, 2020. ISBN 978-85-8422-231-5.