**Centro Universitário UNISATC**

Engenharia de Software 3.ª fase – Banco de Dados II – Prof.Jorge Luiz da Silva

**TRABALHO FINAL COM BASE EM METODOLOGIAS ATIVAS DE APRENDIZAGEM**

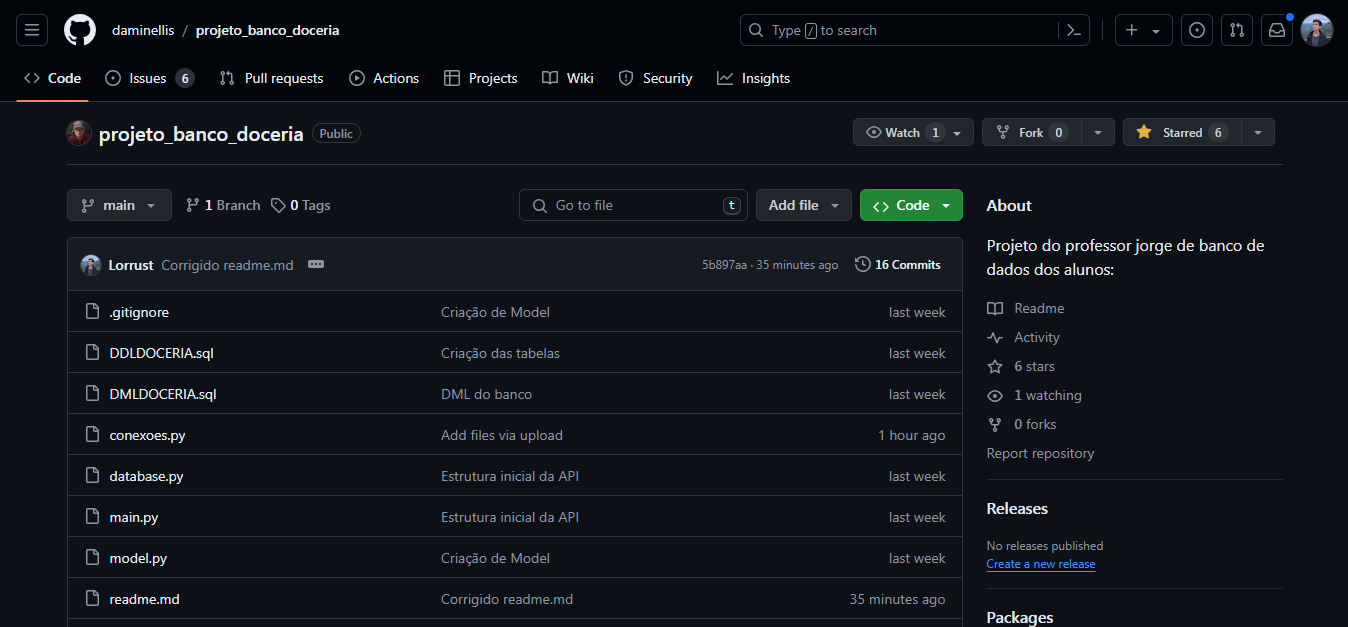
**Projeto de banco de dados para um sistema de doceria**

[Matheus Daminelli](https://github.com/daminellis)  
[Matheus Gastaldi](https://github.com/Matheus2037)  
[Lucas Silva](https://github.com/Lorrust)  
[João Carlos](https://github.com/Churima)  
[Jean Guichard](https://github.com/Guichardx2)  
[Gabriel Morona](https://github.com/M0rona)

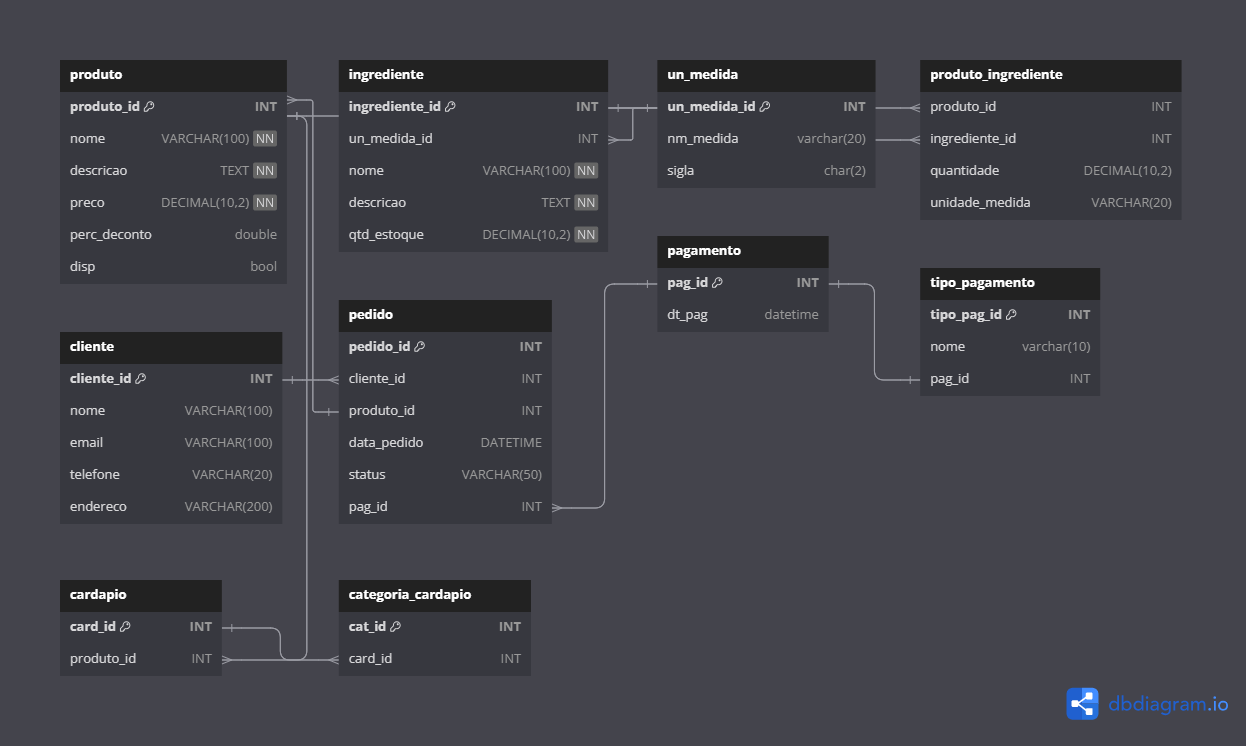
Criciúma, 19/06/2024

**URL do projeto no GitHub**

<https://github.com/daminellis/projeto_banco_doceria>

****

**Modelo ER Físico**



**Dicionário de Dados**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Tabela** | Avaliação | | | | | | |
| **Descrição** | Tabela responsável por armazenar os dados das avaliações criadas pelos professores | | | | | | |
| **Atributos** | | | | | | | |
| **Nome da Coluna** | | **Tipo do Dado** | **Valor min e max** | **Nulidade** | **PK** | **FK** | **Descrição** |
| cd\_avaliacao | | int | 1 – sem limite | NOT NULL | X |  | Código de identificador da avaliação |
| nm\_avaliacao | | varchar(200) | 10 - 200 | NOT NULL |  |  | Nome da avaliação |
|  | |  |  |  |  |  |  |
|  | |  |  |  |  |  |  |
|  | |  |  |  |  |  |  |
|  | |  |  |  |  |  |  |
| **Índice** | | | | | | | |
| **Nome do índice** | | **Clustered** | **NonClustered** | **Unique** | **Colunas** | | |
| Idx\_dt\_avaliacao | | X | X | X | cd\_avaliacao, dt\_avaliacao | | |
| Idx\_tipo | |  | X |  | cd\_avaliacao, dt\_avaliacao | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Tabela** | Avaliação | | | | | | |
| **Descrição** | Tabela responsável por armazenar os dados das avaliações criadas pelos professores | | | | | | |
| **Atributos** | | | | | | | |
| **Nome da Coluna** | | **Tipo do Dado** | **Valor min e max** | **Nulidade** | **PK** | **FK** | **Descrição** |
| cd\_avaliacao | | int | 1 – sem limite | NOT NULL | X |  | Código de identificador da avaliação |
| nm\_avaliacao | | varchar(200) | 10 - 200 | NOT NULL |  |  | Nome da avaliação |
|  | |  |  |  |  |  |  |
|  | |  |  |  |  |  |  |
|  | |  |  |  |  |  |  |
|  | |  |  |  |  |  |  |
| **Índice** | | | | | | | |
| **Nome do índice** | | **Clustered** | **NonClustered** | **Unique** | **Colunas** | | |
| Idx\_dt\_avaliacao | | X | X | X | cd\_avaliacao, dt\_avaliacao | | |
| Idx\_tipo | |  | X |  | cd\_avaliacao, dt\_avaliacao | | |

**Script dos comandos DDL para criação do Banco de dados**

/\* CRIAÇÃO DE TABELA UNIDADE DE MEDIDA \*/

CREATE TABLE public.un\_medida (

un\_medida\_id int4 GENERATED BY DEFAULT AS IDENTITY( INCREMENT BY 1 MINVALUE 1 MAXVALUE 2147483647 START 1 CACHE 1 NO CYCLE) NOT NULL,

nm\_medida varchar(20) NOT NULL,

sigla bpchar(2) NOT NULL,

CONSTRAINT un\_medida\_pkey PRIMARY KEY (un\_medida\_id)

);

/\* CRIAÇÃO DE TABELA PRODUTO \*/

CREATE TABLE public.produto (

produto\_id int4 GENERATED BY DEFAULT AS IDENTITY( MINVALUE 0 NO MAXVALUE START 0 NO CYCLE) NOT NULL,

nome varchar(100) NOT NULL,

descricao text NOT NULL,

preco numeric(10, 2) NOT NULL,

perc\_deconto float4 NULL,

disp bool NULL,

CONSTRAINT produto\_pkey PRIMARY KEY (produto\_id)

);

/\* CRIAÇÃO DE TABELA INGREDIENTE \*/

CREATE TABLE public.ingrediente (

ingrediente\_id int4 GENERATED BY DEFAULT AS IDENTITY( MINVALUE 0 NO MAXVALUE START 0 NO CYCLE) NOT NULL,

un\_medida\_id int4 NULL,

nome varchar(100) NOT NULL,

descricao text NOT NULL,

qtd\_estoque numeric(10, 2) NOT NULL,

CONSTRAINT ingrediente\_pkey PRIMARY KEY (ingrediente\_id)

);

/\* CRIAÇÃO DE FOREIGN KEYS PARA INGREDIENTE \*/

ALTER TABLE public.ingrediente ADD CONSTRAINT ingrediente\_un\_medida\_id\_fkey FOREIGN KEY (un\_medida\_id) REFERENCES public.un\_medida(un\_medida\_id);

/\* CRIAÇÃO DE TABELA DERIVADA PRODUTO-INGREDIENTE \*/

CREATE TABLE public.produto\_ingrediente (

produto\_id int4 NULL,

ingrediente\_id int4 NULL,

quantidade numeric(10, 2) NOT NULL,

unidade\_medida varchar(20) NULL

);

/\* CRIAÇÃO DE FOREIGN KEYS PARA PRODUTO-INGREDIENTE \*/

ALTER TABLE public.produto\_ingrediente ADD CONSTRAINT produto\_ingrediente\_ingrediente\_id\_fkey FOREIGN KEY (ingrediente\_id) REFERENCES public.ingrediente(ingrediente\_id);

ALTER TABLE public.produto\_ingrediente ADD CONSTRAINT produto\_ingrediente\_produto\_id\_fkey FOREIGN KEY (produto\_id) REFERENCES public.produto(produto\_id);

/\* CRIAÇÃO DE TABELA CARDAPIO \*/

CREATE TABLE public.cardapio (

card\_id int4 GENERATED BY DEFAULT AS IDENTITY( INCREMENT BY 1 MINVALUE 1 MAXVALUE 2147483647 START 1 CACHE 1 NO CYCLE) NOT NULL,

produto\_id int4 NULL,

CONSTRAINT cardapio\_pkey PRIMARY KEY (card\_id)

);

/\* CRIAÇÃO DE FOREIGN KEYS PARA CARDAPIO \*/

ALTER TABLE public.cardapio ADD CONSTRAINT cardapio\_produto\_id\_fkey FOREIGN KEY (produto\_id) REFERENCES public.produto(produto\_id);

/\* CRIAÇÃO DE TABELA DERIVADA CATEGORIA-CARDAPIO \*/

CREATE TABLE public.categoria\_cardapio (

cat\_id int4 GENERATED BY DEFAULT AS IDENTITY( INCREMENT BY 1 MINVALUE 1 MAXVALUE 2147483647 START 1 CACHE 1 NO CYCLE) NOT NULL,

card\_id int4 NULL,

descricao varchar(100) DEFAULT '.'::character varying NOT NULL,

CONSTRAINT categoria\_cardapio\_pkey PRIMARY KEY (cat\_id)

);

/\* CRIAÇÃO DE FOREIGN KEYS PARA CATEGORIA-CARDAPIO \*/

ALTER TABLE public.categoria\_cardapio ADD CONSTRAINT categoria\_cardapio\_card\_id\_fkey FOREIGN KEY (card\_id) REFERENCES public.cardapio(card\_id);

/\* CRIAÇÃO DE TABELA CLIENTE \*/

CREATE TABLE public.cliente (

cliente\_id int4 GENERATED BY DEFAULT AS IDENTITY( INCREMENT BY 1 MINVALUE 1 MAXVALUE 2147483647 START 1 CACHE 1 NO CYCLE) NOT NULL,

nome varchar(100) NOT NULL,

email varchar(100) NOT NULL,

telefone varchar(20) NOT NULL,

endereco varchar(200) NOT NULL,

CONSTRAINT cliente\_pkey PRIMARY KEY (cliente\_id)

);

/\* CRIAÇÃO DE TABELA PAGAMENTO \*/

CREATE TABLE public.pagamento (

pag\_id int4 GENERATED BY DEFAULT AS IDENTITY( INCREMENT BY 1 MINVALUE 1 MAXVALUE 2147483647 START 1 CACHE 1 NO CYCLE) NOT NULL,

dt\_pag timestamp NOT NULL,

CONSTRAINT pagamento\_pkey PRIMARY KEY (pag\_id)

);

/\* CRIAÇÃO DE TABELA TIPO DE PAGAMENTO \*/

CREATE TABLE public.tipo\_pagamento (

tipo\_pag\_id int4 GENERATED BY DEFAULT AS IDENTITY( INCREMENT BY 1 MINVALUE 1 MAXVALUE 2147483647 START 1 CACHE 1 NO CYCLE) NOT NULL,

pag\_id int4 NULL,

nome varchar(10) NOT NULL,

CONSTRAINT tipo\_pagamento\_pkey PRIMARY KEY (tipo\_pag\_id)

);

/\* CRIAÇÃO DE FOREIGN KEYS PARA TIPO DE PAGAMENTO \*/

ALTER TABLE public.tipo\_pagamento ADD CONSTRAINT tipo\_pagamento\_pag\_id\_fkey FOREIGN KEY (pag\_id) REFERENCES public.pagamento(pag\_id);

/\* CRIAÇÃO DE TABELA DERIVADA CATEGORIA-CARDAPIO \*/

CREATE TABLE public.pedido (

pedido\_id int4 GENERATED BY DEFAULT AS IDENTITY( INCREMENT BY 1 MINVALUE 1 MAXVALUE 2147483647 START 1 CACHE 1 NO CYCLE) NOT NULL,

cliente\_id int4 NULL,

produto\_id int4 NULL,

pag\_id int4 NULL,

data\_pedido timestamp NOT NULL,

status varchar(50) NOT NULL,

valortotal numeric(15, 2) DEFAULT 0 NULL,

quantidade numeric(15, 2) DEFAULT 0 NULL,

CONSTRAINT pedido\_pkey PRIMARY KEY (pedido\_id)

);

/\* CRIAÇÃO DE FOREIGN KEYS PARA PEDIDO \*/

ALTER TABLE public.pedido ADD CONSTRAINT pedido\_cliente\_id\_fkey FOREIGN KEY (cliente\_id) REFERENCES public.cliente(cliente\_id);

ALTER TABLE public.pedido ADD CONSTRAINT pedido\_cliente\_id\_fkey1 FOREIGN KEY (cliente\_id) REFERENCES public.cliente(cliente\_id);

ALTER TABLE public.pedido ADD CONSTRAINT pedido\_pag\_id\_fkey FOREIGN KEY (pag\_id) REFERENCES public.pagamento(pag\_id);

ALTER TABLE public.pedido ADD CONSTRAINT pedido\_produto\_id\_fkey FOREIGN KEY (produto\_id) REFERENCES public.produto(produto\_id);

**Script que popula as tabelas do Banco de dados**

/\* ADICIONA ALUNOS \*/

--INSERT INTO aluno (nm\_aluno, email)

-- VALUES ('FULANO DE TAL', 'fulano@satc.edu.br');

--GO

insert into aluno (nm\_aluno, email)

select 'MARISA MELO OLIVEIRA', 'email@satc.edu.br' ;

insert into aluno (nm\_aluno, email)

select 'MURILO CARVALHO CARDOSO', 'email@satc.edu.br' ;

insert into aluno (nm\_aluno, email)

select 'VINICIUS ROCHA RODRIGUES', 'email@satc.edu.br' ;

/\* ADICIONA AS ALTERNATIVAS DE CADA QUESTAO\*/

INSERT INTO questao\_item (cd\_questao, ds\_questao\_item, is\_correta) VALUES (1, 'Dados / Comunicação / Conhecimento' ,0);

INSERT INTO questao\_item (cd\_questao, ds\_questao\_item, is\_correta) VALUES (1, 'Análise / Dados / Conhecimento' ,0);

INSERT INTO questao\_item (cd\_questao, ds\_questao\_item, is\_correta) VALUES (1, 'Informação / Compreensão / Comunicação' ,0);

INSERT INTO questao\_item (cd\_questao, ds\_questao\_item, is\_correta) VALUES (1, 'Dados / Informação / Conhecimento' ,1);

GO

INSERT INTO questao\_item (cd\_questao, ds\_questao\_item, is\_correta) VALUES (2, 'F, V, F, F, V', 1);

INSERT INTO questao\_item (cd\_questao, ds\_questao\_item, is\_correta) VALUES (2, 'V, F, F, V, F', 0 );

INSERT INTO questao\_item (cd\_questao, ds\_questao\_item, is\_correta) VALUES (2, 'V, V, V, F, V', 0);

INSERT INTO questao\_item (cd\_questao, ds\_questao\_item, is\_correta) VALUES (2, 'F, F, V, V, F', 0);

GO

**Principais consultas mapeadas baseadas em regras de negócio (mínimo 4)**

--13. (Aberta) Utilizando a linguagem SQL, execute consultas que possam responder a seguinte questão: Qual(is) Carro(s) possui(em) mais sinistros cadastrados?

select s.placa, COUNT(\*)

from sinistro s inner join carro c

on s.placa = c.placa

group by s.placa

order by COUNT(\*) desc

--14. (Aberta) Quantos clientes cadastrados possuem letra “a” no meio e não no final?

select \* --count(\*)

from cliente

where nome like '%a%'

and nome not like '%a'

--15. (Aberta) Quantos clientes não possuem apólice?

select nome, cod\_apolice --\* --count(\*)

from cliente c left join apolice a

on c.cod\_cliente = a.cod\_cliente

where cod\_apolice is null

select nome, cod\_apolice

from cliente c inner join apolice a on c.cod\_cliente = a.cod\_cliente

--16. (Aberta) Quais carros não possuem apólice?

select \* --count(\*)

from carro c left join apolice a on c.placa = a.placa

where cod\_apolice is null

select \*

from apolice

where placa not in (select placa from carro)