

ANHANGUERA EDUCACIONAL
SCRN (ASA NORTE) 708/709 - BLOCO B, S/N, ASA NORTE - BRASÍLIA – DF

**SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE
SISTEMAS**

AUTOR(ES):
LUCAS LOPES RIBEIRO

PROGRAMAÇÃO E DESENVOLVIMENTO DE BANCO DE DADOS
Criação de Banco de Dados com Operações e Manipulação e Acesso aos Dados.

BRASILIA

2024

Lucas Lopes Ribeiro

PROGRAMAÇÃO E DESENVOLVIMENTO E BANCO DE DADOS
Criação de Banco de Dados com Operações e Manipulação e Acesso aos Dados.

Portifólio de Análise orientada a objetos apresentado ao(s) tutor(es): André Martins e Vinicius Mendes Gomes da Silva, do curso de Tecnologia em Análise e Desenvolvimento de Sistemas da instituição Anhanguera Educacional.

BRASILIA

2024

SUMARIO

1 INTRODUÇÃO.....	4
2 DESENVOLVIMENTO	5
3 CONCLUSÃO	10

1 INTRODUÇÃO

Este trabalho aborda a criação e manipulação de um banco de dados denominado “Loja”, utilizando a linguagem SQL. As principais etapas incluem a criação da estrutura do banco de dados, desenvolvimento da estrutura do banco de dados com base em um diagrama entidade-relacionamento (DER) pré-definido, utilização do MySQL Server e do software MySQL Workbench para criar a base de dados.

Em seguida será realizada a etapa de inserção de registros, populando as tabelas do banco de dados com registros fictícios. O script conterá os comandos de manipulação (DML) necessários para inserir pelo menos três registros em cada tabela.

O trabalho se encerra com a criação da visão “ContasNaoPagas”, que retornará todas as contas pendentes de pagamento contendo o ID da conta a receber, nome e CPF do cliente associado à conta, data de vencimento da conta e o valor da conta.

2 DESENVOLVIMENTO

Criação do banco de dados “LOJA” e suas tabelas.

Nesta etapa será definida a estrutura do banco seu comportamento.

A tabela ContasReceber é definida com as seguintes colunas:

ID: Chave primária, auto incrementada e não nula.

Cliente_id: Referência ao ID do cliente associado à conta.

FaturaVendaID: Armazena o número da fatura de venda.

DataConta e DataVencimento: Campos para datas de Criação e vencimento da conta.

Valor: Valor da conta a receber.

Situacao: Enumerador para indicar a situação da conta (valores '1', '2', '3').

As demais tabelas “Cliente” “Município” e “Estado” seguem a mesma ordem, cada uma a tipagem adequada das suas colunas e registros que devem receber.

The screenshot shows a database management tool interface. On the left, the 'Navigator' pane displays a list of schemas: 'exercicio', 'livraria', 'projeto', 'sakila', 'sys', and 'world'. Below this, the 'Administration' tab is active, showing 'Schemas' and 'Information'. The main area displays the 'SCRIPT_PROJETO_PORTIFOLIO*' window with the following SQL code:

```

1 • create database LOJA; Use LOJA;
2 • CREATE TABLE ContasReceber (
3     id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
4     Cliente_id INT,
5     FaturaVendaID BIGINT,
6     DataConta DATE NOT NULL,
7     DataVencimento DATE NOT NULL,
8     Valor DECIMAL(18,2) NOT NULL,
9     Situacao ENUM('1', '2', '3') NOT NULL);
10 • CREATE TABLE Cliente (
11     ID INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
12     Nome VARCHAR(80),
13     CPF CHAR(15),
14     celular CHAR(15),
15     EndLougradouro VARCHAR(100),
16     EndNumero VARCHAR(15),
17     EndMunicípio INT,
18     EndCep CHAR(10),
19     Município_ID INT);
20 • CREATE TABLE Município (
21     id INT PRIMARY KEY AUTO_INCREMENT,
22     Estado_ID INT,
23     nome VARCHAR(80),
24     codigo_IBGE INT);
25 • CREATE TABLE Estado (
26     ID INT PRIMARY KEY AUTO_INCREMENT NOT NULL,
27     Nome VARCHAR(50).
  
```

Nesta etapa do trabalho será realizado a inserção de registros fictícios nas tabelas do banco de dados “Loja”. Cada tabela representa um conjunto específico de informações, e os valores inseridos refletem dados de clientes e contas a receber.

Utilizamos o comando INSERT INTO para adicionar registros à tabela ContasReceber.

Os campos inseridos incluem:

Cliente_id: Referência ao ID do cliente associado à conta.

FaturaVendaID: Número da fatura de venda.

DataConta e DataVencimento: Datas de criação e vencimento da conta.

Valor: Valor da conta a receber.

Situacao: Enumerador que indica a situação da conta (valores ‘1’, ‘2’ ou ‘3’).

Tabela Cliente:

Utilizamos o mesmo comando INSERT INTO para adicionar registros à tabela Cliente.

Os campos inseridos incluem:

Nome, CPF, celular: Informações pessoais do cliente.

EndLougradouro, EndNumero, EndMunicipio, EndCep: Endereço do cliente.

Municipio_ID: Referência ao município associado ao cliente.

Tabela Estado e Municipio:

Para as tabelas Estado e Municipio, seguimos o mesmo padrão de inserção.

Adicionamos estados e municípios fictícios, associando-os por meio de IDs e códigos IBGE.

A imagem a seguir demonstrará o código utilizado a resposta do WorkBench indicando que as operações foram aceitas corretamente. .

```

PROJETO_PORTIFOLIO* x SCRIPT_PROJETO_PORTIFO...
Limit to 300 rows

1 • INSERT INTO ContasReceber (Cliente_id, FaturaVendaID, DataConta, DataVencimento, Valor, Situacao)
2 VALUES
3 (1, 7071995202, '2024-10-02', '2024-10-10', 10000.25, '3'),
4 (2, 55168162161, '2023-12-10', '2024-01-05', 5000.50, '1'),
5 (3, 22145661651, '2024-09-05', '2022-05-10', 3000.00, '2'),
6 (4, 81651815154, '2024-08-03', '2024-03-08', 8250.00, '1'),
7 (5, 95541846181, '2021-03-10', '2022-04-01', 1050.00, '1'),
8 (6, 68915165165, '2023-04-08', '2024-08-10', 20000.00, '2');
9 • INSERT INTO Cliente (Nome, CPF, celular, EndLougradouro, EndNumero, EndMunicipio, EndCep, Municipio_ID)
10 VALUES
11 ('Lucas', '05293110110', '(61)991250909', 'QRI 25 Residencial Santos Dumont', '16666', 25, '72594225', 30),
12 ('Eduardo Facção', '09188815009', '(46)981195594', 'Estrada da dor 666', '44444', 889, '72448956', 50),
13 ('Cassia Heller', '12345678900', '(55)998538445', 'Casinha do Renato Russo com Matagal Pesado', '4125', 789, '78445213', 75),
14 ('Douglas DUNDUN', '98765432177', '(66)984238477', 'Roleta Macabra Sorteou Nosso Numero', '4152', 143, '72451447', 13);
15 • INSERT INTO Estado (Nome, UF)
16 VALUES
17 ('São Paulo', 'SP'),
18 ('Rio de Janeiro', 'RJ'),
19 ('Minas Gerais', 'MG'),
20 ('Bahia', 'BA');
21 • INSERT INTO Municipio (Estado_ID, Nome, codigo_IBGE)
22 VALUES
23 (61, 'Brasilia', 30),
24 (11, 'São Paulo', 50),
25 (21, 'Rio de Janeiro', 75),
26 (31, 'Minas Gerais', 13);
27

```

Navigator

SCHEMAS

Filter objects

- loja
 - Tables
 - cliente
 - Columns
 - Indexes
 - Foreign Keys
 - Triggers
 - contasreceber
 - Columns
 - Indexes
 - Foreign Keys
 - Triggers
 - estado
 - Columns
 - Indexes
 - Foreign Keys
 - Triggers
 - municipio
 - Views
 - Stored Procedures

Administration Schemas

PROJETO_PORTIFOLIO* x SCRIPT_PROJETO_PORTIFO...

Limit to 300 rows

1 • INSERT INTO ContasReceber (Cliente_id, FaturaVendaID, DataConta, DataVencimento, Valor, Situacao)

Output

Action Output

#	Time	Action	Message
✓ 1	16:55:47	drop database loja	4 row(s) affected
✓ 2	16:55:59	create database loja	1 row(s) affected
✓ 3	16:55:59	use loja	0 row(s) affected
✗ 4	16:56:03	create database LOJA	Error Code: 1007. Can't create database 'loja'; database exists
✓ 5	16:56:09	CREATE TABLE ContasReceber (id INT PRIMARY KEY AUTO_INCREMENT NOT NULL,...	0 row(s) affected
✓ 6	16:56:09	CREATE TABLE Cliente (ID INT PRIMARY KEY AUTO_INCREMENT NOT NULL, Nom...	0 row(s) affected
✓ 7	16:56:09	CREATE TABLE Municipio (id INT PRIMARY KEY AUTO_INCREMENT, Estado_ID INT...	0 row(s) affected
✓ 8	16:56:09	CREATE TABLE Estado (ID INT PRIMARY KEY AUTO_INCREMENT NOT NULL, Nom...	0 row(s) affected
✓ 9	16:56:15	INSERT INTO ContasReceber (Cliente_id, FaturaVendaID, DataConta, DataVencimento, Valor...	6 row(s) affected Records: 6 Duplicates: 0 Warnings: 0
✓ 10	16:56:15	INSERT INTO Cliente (Nome, CPF, celular, EndLougradouro, EndNumero, EndMunicipio, End...	4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0
✓ 11	16:56:15	INSERT INTO Estado (Nome, UF) VALUES ('São Paulo', 'SP'), ('Rio de Janeiro', 'RJ'), (Minas ...	4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0
✓ 12	16:56:15	INSERT INTO Municipio (Estado_ID, Nome, codigo_IBGE) VALUES (61, 'Brasilia', 30), (11, 'S...	4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0

Neste momento final do projeto, é realizado o script de consulta das informações, utilizando as funções VIEW e INNER JOIN, a fim de visualizar as contas que ainda não foram pagas, conforme solicitado no material do portfólio.

The screenshot displays a database management interface with a script editor and a results pane. The script editor shows a SQL query to create a view and select data. The results pane shows the execution output, including the creation of the view and the selection of data.

Script Editor:

```

1 • CREATE VIEW ContasNaoPagas AS
2   SELECT
3     CR.ID As 'ID da conta a receber',
4     C.Nome As 'Nome do Cliente',
5     C.CPF As 'CPF do Cliente',
6     CR.DataVencimento As 'Data de Vencimento',
7     CR.Valor As 'Valor da Conta'
8   FROM ContasReceber CR
9   Inner join Cliente C ON CR.Cliente_ID = C.ID
10  • where CR.Situacao = '1';

```

Result Grid:

	ID da conta a receber	Nome do Cliente	CPF do Cliente	Data de Vencimento	Valor da Conta
▶	2	Eduardo Facção	09188815009	2024-01-05	5000.50
	4	Douglas DUNDUN	98765432177	2024-03-08	8250.00

Output:

#	Time	Action	Message
✓ 10	16:56:15	INSERT INTO Cliente (Nome, CPF, celular, EndLougradouro, EndNumero, EndMunicipio, E...	4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0
✓ 11	16:56:15	INSERT INTO Estado (Nome, UF) VALUES ('São Paulo', 'SP'), ('Rio de Janeiro', 'RJ'), ('Min...	4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0
✓ 12	16:56:15	INSERT INTO Municipio (Estado_ID, Nome, codigo_IBGE) VALUES (61, 'Brasilia', 30), (11,...	4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0
✓ 13	16:58:12	CREATE VIEW ContasNaoPagas AS SELECT CR.ID As 'ID da conta a receber', C.No...	0 row(s) affected
✓ 14	16:58:12	select * from ContasNaoPagas LIMIT 0, 300	2 row(s) returned

Criação da Visualização (VIEW):

A linha `CREATE VIEW ContasNaoPagas AS` inicia a criação da visualização chamada “ContasNaoPagas”. Essa visualização será usada para apresentar os resultados da consulta.

A consulta seleciona os campos da tabela “ContasReceber”:

CR.ID: O ID da conta a receber.

C.Nome: O nome do cliente associado à conta.

C.CPF: O CPF do cliente.

CR.DataVencimento: A data de vencimento da conta.

CR.Valor: O valor da conta.

INNER JOIN:

O INNER JOIN combina registros das tabelas “ContasReceber” e “Cliente”.

Ele utiliza a coluna Cliente_ID como chave de ligação.

O resultado é uma combinação das informações das contas a receber com os dados dos clientes correspondentes.

Filtro por Situação:

O filtro where CR.Situacao = '1' garante que apenas contas com situação igual a ‘1’ sejam incluídas na visualização “ContasNaoPagas”.

O resultado final apresenta as contas não pagas, contendo os campos selecionados e os dados combinados das duas tabelas, conforme imagem apresentada anteriormente.

3 CONCLUSÃO

Utilizando a linguagem de código SQL, foi possível demonstrar a Criação do banco de dados e suas tabelas, que no caso solicitado, seria para demonstrar como uma loja funcionaria. Foi utilizado o MySQL Workbench para criar a base de dados, respeitando o modelo definido no Diagrama Entidade-Relacionamento (DER) para estruturar as tabelas. As estruturas de dados foram criadas com os comandos de definição de dados (DDL) em SQL, conforme imagens apresentadas ao longo do relatório.

Foi elaborado um script para inserir os registros dentro das tabelas, contendo comandos de manipulação de dados (DML). Essa etapa foi fundamental para simular dados reais e testar o funcionamento do sistema.

No script final, de consulta das contas não pagas, foram utilizados comandos de consulta (DQL) para criar uma visão (VIEW). Assim, a visão “ContasNaoPagas” foi projetada para retornar todas as contas que foram definidas na "Situação = 1 (contas não pagas).

Os campos exibidos na visão incluem o ID da conta a receber, o nome e CPF do cliente associado à conta, a data de vencimento e o valor da conta.

