Relatório de Aula Prática - Programação e Desenvolvimento de Banco de Dados

Análise e Desenvolvimento de Sistemas

Jéssica Baptista Viana

ANHANGUERA

Rio de Janeiro, 21 de outubro de 2024.

INTRODUÇÃO

Neste exercício prático, o objetivo principal é desenvolver um banco de dados utilizando SQL, focando na criação, manipulação e consulta de dados. A atividade faz parte da disciplina de Programação e Desenvolvimento de Banco de Dados e envolve a construção de um sistema de gerenciamento para uma loja, passando pela criação de tabelas, inserção de dados e consultas.

O MySQL Workbench foi a ferramenta escolhida para a execução do projeto, pois é um ambiente integrado que facilita o desenvolvimento e a manutenção de bancos de dados MySQL. O exercício simula um cenário real de criação e gestão de um sistema de banco de dados, permitindo aos alunos uma experiência prática e direta com SQL, aplicando comandos para definição, manipulação e consulta de dados.

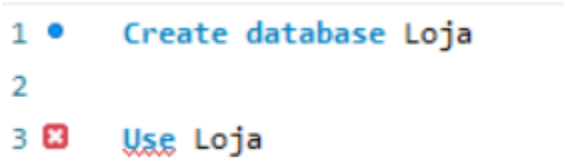
As etapas envolvem a criação de um banco de dados chamado "Loja", com tabelas inter-relacionadas, obedecendo a regras de integridade e baseadas em um diagrama de entidade-relacionamento (DER). Após isso, o banco será populado com dados fictícios, e consultas específicas serão feitas usando comandos SQL, incluindo a criação de uma "View" para exibir as contas que ainda não foram pagas. Esse exercício tem como finalidade reforçar os conhecimentos sobre o uso de SQL em um contexto de negócios real.

DESENVOLVIMENTO

Etapas do Desenvolvimento do Exercício Proposto :

1. Criação do Banco de Dados

A primeira etapa consiste na criação de um banco de dados chamado "Loja". Para isso, usamos o comando SQL CREATE DATABASE, e em seguida, selecionamos o banco para utilização com o comando USE.



2. Criação das Tabelas

Na segunda etapa, serão criadas as tabelas que compõem o banco de dados, conforme o diagrama de entidade-relacionamento (DER) e as especificações dadas. Cada tabela será configurada com suas colunas, chaves primárias e estrangeiras.

- **Tabela Estado:** Responsável por armazenar informações sobre os estados brasileiros, contendo um identificador único (ID), nome e sigla (UF).

```sql
Create table  Estado (
idEstado int not null auto_increment,
Nome varchar(50) not null,
UF char(2),
primary key (idEstado)
);
```

- **Tabela Município:** Armazena os municípios e faz referência à tabela **Estado** através de uma chave estrangeira. Além do nome e do código IBGE, os municípios estão associados a um estado.

```sql
Create table Municipio (
id int not null auto_increment,
Nome varchar(80) not null,
codIBGE int not null,
Estado_idEstado int not null,
primary key (id),
constraint fk_Municipio_Estado
foreign key (Estado_idEstado)
references Estado (idEstado)
on delete no action);
```

- **Tabela Cliente:** Contém os dados dos clientes, como nome, CPF, celular e endereço. A tabela inclui uma chave estrangeira para o município, referenciando a tabela **Municipio**.

```
23  Create table Cliente(
24  ID int not null auto_increment,
25  Nome varchar(80) not null,
26  CPF char(11) not null,
27  Celular char(11),
28  EndLogradouro varchar(100) not null,
29  EndNumero varchar(10) not null,
30  EndMunicipio int not null,
31  EndCEP char(8),
32  Municipio_ID int not null,
33  primary key (ID),
34  constraint fk_Cliente_Municipio
35  foreign key (Municipio_ID)
36  references municipio (id)
37  on delete no action
38  on update no action
39  );
```

- **Tabela ContasReceber:** Responsável por armazenar as contas a receber dos clientes, incluindo informações como data da conta, data de vencimento, valor e situação da conta (1 - registrada, 2 - cancelada, 3 - paga). Há uma chave estrangeira referenciando a tabela **Cliente**.

```
40
41  create table ContasReceber (
42  id int not null  auto_increment,
43  FaturaVenda int,
44  DataConta date not null,
45  DataVencimento date not null,
46  Valor decimal (18,2) not null,
47  Situacao ENUM('1','2','3'),
48  Cliente_ID int not null,
49  primary key(id),
50  constraint fk_ContasReceber_Cliente
51  foreign key (Cliente_ID)
52  references Cliente (id)
53  on delete no action
54  on update no action
55  );
56
```

3. Inserção de Dados nas Tabelas

Após a criação das tabelas, são inseridos dados nas tabelas **Estado**, **Municipio**, **Cliente**, e **ContasReceber**, utilizando o comando `INSERT INTO`.

- **Inserção na tabela Estado:**

```
57  INSERT INTO estado (Nome,UF)
58  VALUES
59  ('Paraná','PR'),
60  ('Santa Catarina','SC'),
61  ('Rio de Janeiro','RJ')
```

	idEstado	Nome	UF
▶	1	São Paulo	SP
	2	Paraná	PR
	3	Santa Catarina	SC
	4	Rio de Janeiro	RJ
*	NULL	NULL	NULL

- **Inserção na tabela Municipio:**

```
INSERT INTO municipio (Nome, CodIBGE, Estado_idEstado)
VALUES
 ('Bom Sucesso', '4103206',2),
 ('Aurora','4201901',3),
 ('Armação de Búzios','3300100',4)
```

	id	Nome	codIBGE	Estado_idEstado
	1	Ribeirao Preto	3543402	1
	2	Bom Sucesso	4103206	2
	3	Aurora	4201901	3
▶	4	Armação de Búzios	3300100	4
*	NULL	NULL	NULL	NULL

- **Inserção na tabela Cliente:**

```
INSERT INTO Cliente (Nome, CPF, Celular, EndLogradouro,EndNumero,EndMunicipio, EndCEP,Municipio_ID)
VALUES
('Jessica',15697458794,5596974236,'Rua 5','101',2,'2278947',1),
('Juliana',1565558794,5596979236,'Rua 12','121',9,'2856478',2),
('Ronaldo',1569665794,5788955536,'Rua C','155',8,'5554785',3),
('Ramon',1569785594,5596988236,'Rua A','199',6,'9558522',4)
```

ID	Nome	CPF	Celular	EndLogradouro	EndNumero	EndMunicipio	EndCEP	Municipio_ID
1	Jessica	15697458794	5596974236	Rua 5	101	2	2278947	1
2	Juliana	1565558794	5596979236	Rua 12	121	9	2856478	2
3	Ronaldo	1569665794	5788955536	Rua C	155	8	5554785	3
4	Ramon	1569785594	5596988236	Rua A	199	6	9558522	4
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- **Inserção na tabela ContasReceber:**

```
INSERT INTO ContasReceber (FaturaVenda,DataConta,DataVencimento,Valor,Situacao,Cliente_ID)
VALUES
(4389,'2024-08-10','2024-08-15',199.00, 1,1)
(4339,'2024-07-10','2024-08-10',159.74, 3,4),
(NULL,'2024-02-15','2024-03-15',355.74,2 ,3),
(8389,'2024-03-18','2024-04-19',199.84, 1,2)
```

	id	FaturaVenda	DataConta	DataVencimento	Valor	Situacao	Cliente_ID
	1	4339	2024-07-10	2024-08-10	159.74	3	3
▶	2	NULL	2024-02-15	2024-03-15	355.74	2	4
	3	8389	2024-03-18	2024-04-19	199.84	3	2
	4	4389	2024-08-10	2024-08-15	199.00	1	1
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

etapa 3

	ContasId	ClienteNOme	ClienteCPF	DataVencimento	Valor
▶	4	Jessica	15697458794	2024-08-15	199.00

4 - Criação da View para contas não pagas:

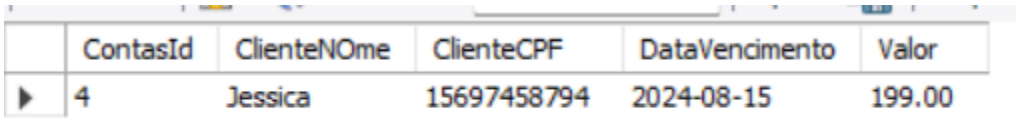
- Criamos uma view chamada **ContasNaopagas**, que retorna as contas que ainda não foram pagas. Esta view inclui informações como o ID da conta, nome do cliente, CPF, data de vencimento e valor da conta.

```
CREATE VIEW ContasNaopagas as
Select cr.id as  ContasId,
nome as ClienteNOme,
c.cpf as ClienteCPF,
cr.DataVencimento,
cr.Valor
from contasreceber cr
Join
cliente c on Cliente_ID = c.id
where
cr.Situacao = '1';
```

Consulta eResultado da View criada:

Para visualizar os dados da view criada, podemos utilizar o seguinte comando:

```
select * from contasnaopagas
```



	ContasId	ClienteNOme	ClienteCPF	DataVencimento	Valor
▶	4	Jessica	15697458794	2024-08-15	199.00

CONCLUSÃO

A realização deste exercício prático foi fundamental para aplicar de maneira concreta os conceitos aprendidos na disciplina de Programação e Desenvolvimento de Banco de Dados. Durante as etapas, foi possível explorar desde a criação de um banco de dados utilizando a linguagem SQL até a manipulação e consulta de informações essenciais para a gestão de uma loja.

A criação de tabelas com chaves primárias e estrangeiras permitiu entender a importância dos relacionamentos entre diferentes entidades, garantindo a integridade dos dados. Além disso, a implementação de comandos de inserção e consulta consolidou o aprendizado sobre manipulação de dados e a utilização de **views**, como exemplificado pela criação da `ContasNaopagas`, que torna o processo de análise de contas pendentes mais ágil e eficiente.

Relatório de Aula Prática - Programação e Desenvolvimento de Banco de Dados

Essa prática proporcionou uma visão clara de como os sistemas de banco de dados são fundamentais no suporte a operações do dia a dia em um ambiente empresarial, reforçando a importância do SQL como uma ferramenta poderosa na organização e consulta de grandes volumes de dados. A partir deste exercício, compreendemos de forma prática a relevância de um banco de dados bem estruturado no funcionamento de sistemas comerciais.