Universidade Federal de Ouro Preto
Campus Morro do Cruzeiro

BCC 321

Banco de Dados I

Trabalho Prático

Jonathas Lopes Moreira

Professor: Luiz Henrique de Campos

Ouro Preto, 22 de Junho de 2015

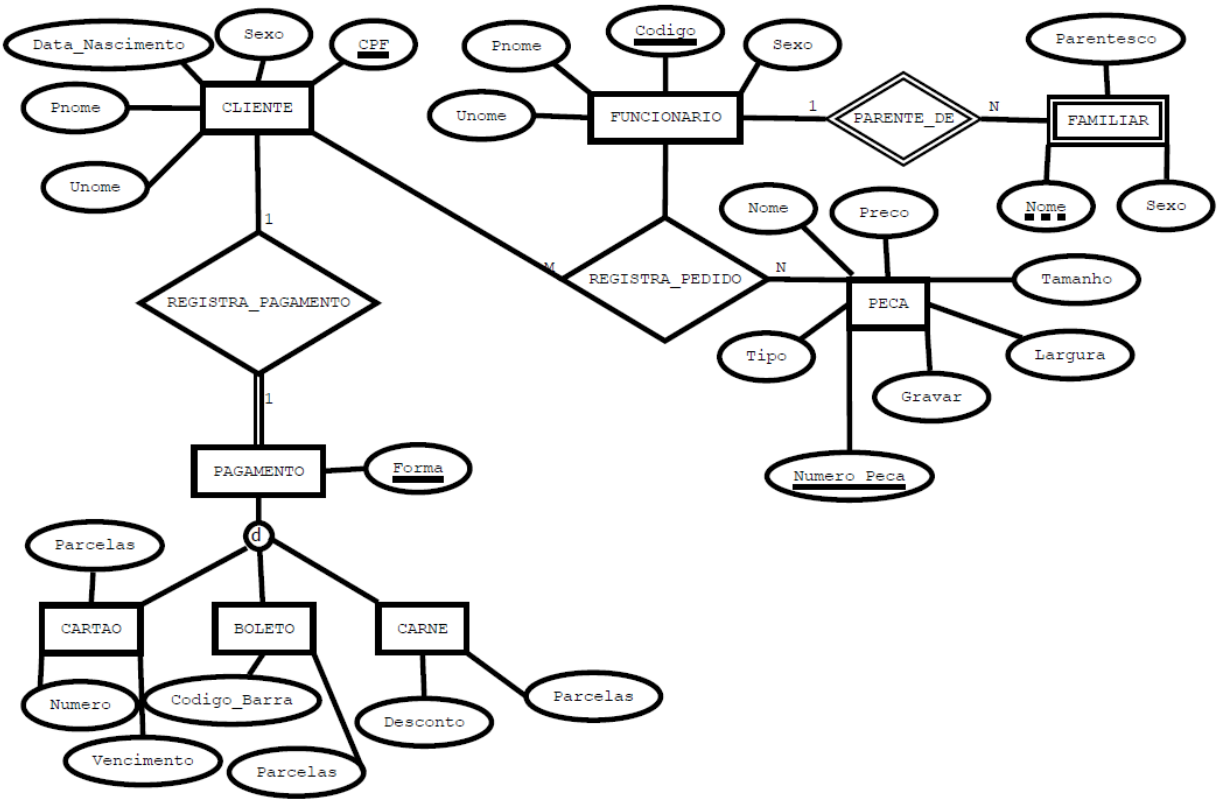
Sumário

1. Requisitos de dados

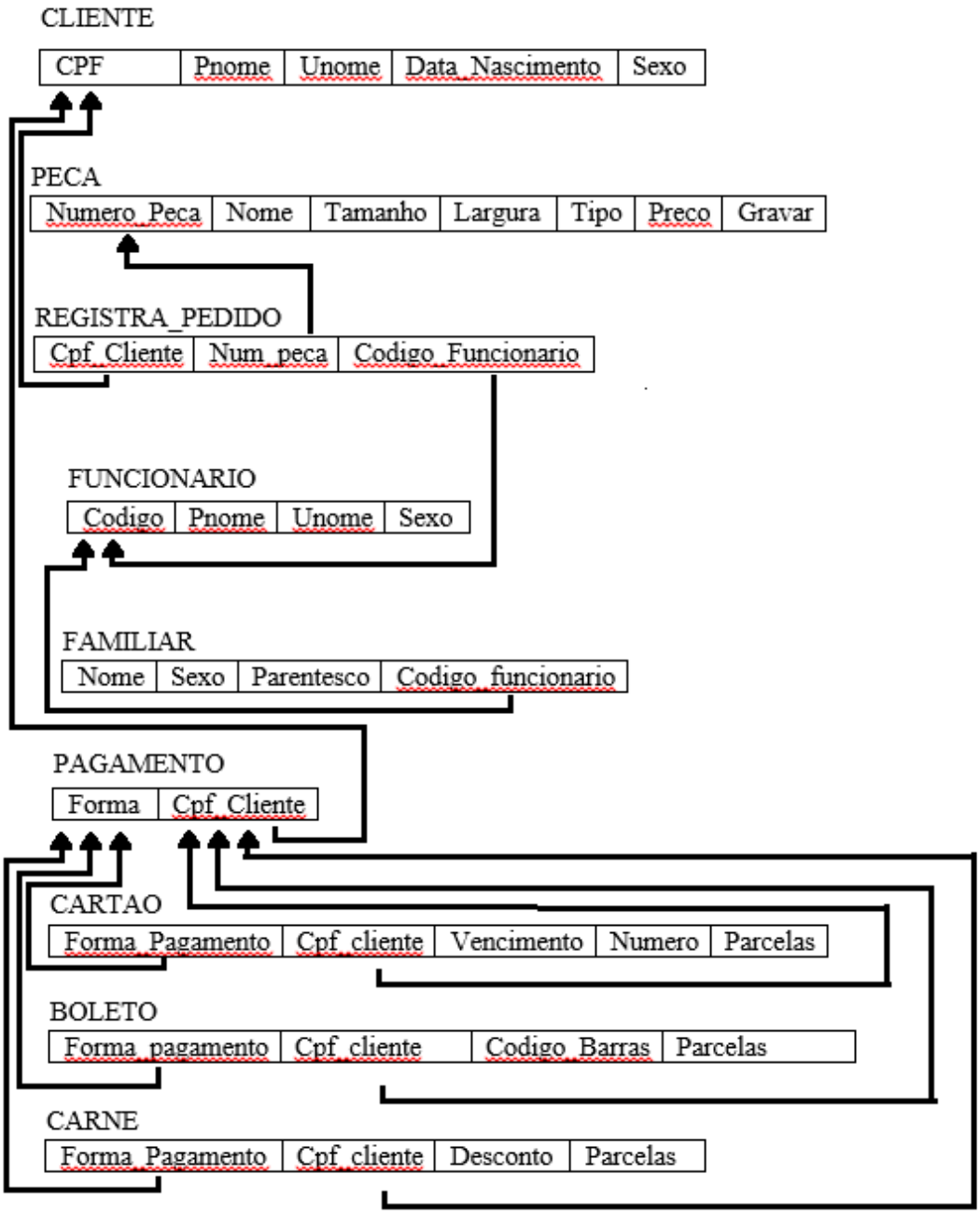
Esse trabalho consiste em uma aplicação de banco de dados referente a consultas a uma pequena joalheria conforme os seguintes requisitos:

- É necessário o cadastro de todos os clientes, para cada cliente deve-se armazenar seu CPF, primeiro e último nome, assim como sua data de nascimento e sexo.
- Deve-se armazenar os dados sobre cada funcionário que atende o cliente, o mesmo que possui um código que o identifica, primeiro e último nome, e sexo, cada funcionário também possui um familiar, deve se armazenar o nome, sexo, e grau de parentesco que o funcionário possui com seu familiar.
- Possui-se na loja três tipos de peças, anéis, cordões, e placas. Para cada uma deve-se armazenar seu número que a identifica, nome, tamanho, largura, tipo (ouro, prata ou folheado), preço, e caso seja placa, deve-se ter a opção gravar para se escrever um nome na placa.
- Todos os pedidos de todos os clientes devem ser armazenados em algum local sendo possível identificar o cpf de cada cliente que fez o pedido, qual(is) peça(s) ele pediu, assim como o funcionário que atendeu cada pedido.
- Existem três formas de pagamento (boleto, carne, e cartão), todas devem informar o número de parcelas que o cliente pagará, além de, caso via boleto, deve-se apresentar código de barras, caso cartão, deve-se apresentar o seu número e dia do vencimento, caso seja carne, deve-se apresentar o desconto.
- O aplicação dentre outras operações, deve ser capaz de informar o nome de todas as compras de um cliente através de seu nome, o nome de todos os clientes que um funcionário atendeu, o número de familiares de um funcionário, a forma de pagamento e o valor total da compra de um cliente através de seu nome.
- Com um nível de dificuldade um pouco maior, deve apresentar a capacidade de separar as pessoas que compraram nos três grupos de pagamento, para cada grupo, deve se informar o nome completo do(s) cliente(s), o tipo do grupo ou forma de pagamento (boleto, cartão ou carne), os dados específicos de cada forma, além do valor total e o número de parcelas.
- Operações como inserção, atualização e exclusão de dados também devem estar disponíveis.

2. DER


Data_Nascimento
Sexo
CPF
Pnome
CLIENTE
Unome
1
REGISTRA_PAGAMENTO
1
PAGAMENTO
Forma
d
Parcelas
CARTAO
BOLETO
CARNE
Numero
Codigo_Barra
Vencimento
Parcelas
Desconto
Parcelas
Pnome
Codigo
Sexo
Unome
FUNCIONARIO
1
PARENTE_DE
N
Parentesco
FAMILIAR
Nome
Sexo
M
REGISTRA_PEDIDO
N
Nome
Preco
Tamanho
PECA
Tipo
Largura
Gravar
Numero_Peca

3. Esquema Relacional



4. Comandos Sql e do Postgres utilizados na implementação do banco de dados

Antes de mais nada, por utilizar o sistema operacional Windows, foi necessário configurar o Prompt de Comando para o psql aceitar caracteres brasileiros com o comando: **cmd.exe /c chcp 1252**, ativando então a porta 1252.

O comando para abrir o banco de dados cujo nome é joalheria, e usuário jon e no localhost é:

psql -U jon -d joalheria -h localhost

A versão do Postgres utilizada conforme podemos ver com o comando **psql -V** é a:

(PostgreSQL) 9.4.2.

Os comandos do Postgres listados abaixo foram frequentemente utilizados:

\d: Para listar todas as tabelas existentes.

\d <Nome da Tabela> Para listar o esquema das tabelas.

O Banco de Dados foi criado com o comando:

create database joalheria;

Não foi usado o usuário padrão do Postgres, mas sim um superusuario **jon**, que foi criado da seguinte maneira:

create user jon superuser inherit createdb createrole;

Foi alterado o proprietário do Banco de dados para **jon** com o comando:

alter database joalheria owner to jon;

Criação das tabelas:

CLIENTE

```
create table cliente
(
        Cpf char(11) primary key,
        Pnome varchar(20) not null,
        Unome varchar(20) not null,
        Data_nascimento date not null,
        Sexo char not null
);
```

FUNCIONARIO

```
create table funcionario
(
        Codigo int  primary key,
        Pnome varchar(30) not null,
        Unome varchar(30) not null,
        Sexo char not null
);
```

PECA

```
create table peca
(
        Numero char(8) primary key,
        Nome varchar(30) not null,
        Tamanho float not null,
        Largura float not null,
        Tipo char(10) not null,
        Preco float not null,
        Gravar varchar(30),
);
```

REGISTRA_PEDIDO

```
create table registra_pedido
(
        Cpf_cliente char(11) not null references cliente on update cascade on delete cascade,
        Num_peca char(8) not null references peca on update cascade on delete cascade,
        Codigo_funcionario int not null references funcionario on update cascade on delete cascade,
        Primary key(cpf_cliente,num_peca)
);
```

FAMILIAR

```
create table familiar
(
        Nome varchar(11) not null,
        Sexo char not null,
        Parentesco varchar(30),
        Codigo_funcionario int not null references funcionário on update cascade on delete cascade,
        Primary key(nome,codigo_funcionario)
);
```

PAGAMENTO

```
create table pagamento
(
        Forma varchar(10) not null,
        Cpf_cliente char(11) not null unique references cliente on update cascade on delete cascade,
        Primary key(forma,cpf_cliente),
);
```

BOLETO

```
create table boleto
(
        Forma_pagamento varchar(10) not null,
        Cpf_cliente char(11)  not null,
        Codigo_barras char(20) not null unique,
        Parcelas int not null,
        Primary key(forma_pamamento,cpf),
        Foreign key(forma_pagamento, cpf_cliente) references pagamento on update cascade on delete cascade
);
```

CARNE

```
create table carne
(
        Forma_pagamento varchar(10) not null,
        Cpf_cliente char(11) not null,
        desconto float not null unique,
        Parcelas int not null,
        Primary key(forma_pamamento,cpf),
        Foreign key(forma_pagamento, cpf_cliente) references pagamento on
        update cascade on delete cascade
);
```

5. Comandos Sql utilizados para povoar o banco de dados

Os comandos para povoamento do banco de dados utilizados foram de três tipos:

Insert into cliente values ('11111111111', 'Jonathas', 'Moreira', '14-08-1993', 'M');

Insert into cliente (cpf,pnome,unome,data_nascimento,sexo) values ('11111111111', 'Jonathas', 'Moreira', '14-08-1993', 'M');

Ambos para inserção de uma tupla específica.

Para todas as tabelas foram criados arquivos .txt para inserção dos dados, para a tabela cliente por exemplo, o comando para povoamento foi:

Copy cliente from 'C:\Banco de Dados\clientes.txt';

Assim para as outras tabelas os arquivos foram: boletos.txt, carnes.txt, cartoes.txt, familiares.txt, funcionarios.txt pagamentos.txt, pecas.txt, registra_pedidos.txt.

6. Requisitos funcionais da aplicação

A aplicação permite a consulta, inserção, deleção e atualização do banco de dados joalheria, assim como consultas aninhadas correlacionadas.

7. Código da aplicação e comentários.

Se encontra no seguinte diretório o código:#

8. Conclusão

Esse trabalho teve como objetivo principal, a criação de uma aplicação que utiliza os conceitos aprendidos na disciplina de Banco de Dados na prática, a mesma que faz operações em um banco de dados (joalheria) que acessa um servidor local (localhost) no computador.

Foi utilizado apenas o material fornecido pelo professor e o livro Sistemas de Banco de Dados, Elmasri Navathe como referências bibliográficas.