



BANCO DE DADOS II

CAMPUS BIRIGUI

PROJETO BANCO

EMILY DA SILVA COSTA BI3003892

LEONARDO PAVAN CUNHA MATTOS BI3004015

VINICIUS DE SOUZA SANTOS BI3008061

NOVEMBRO DE 2021

SUMÁRIO

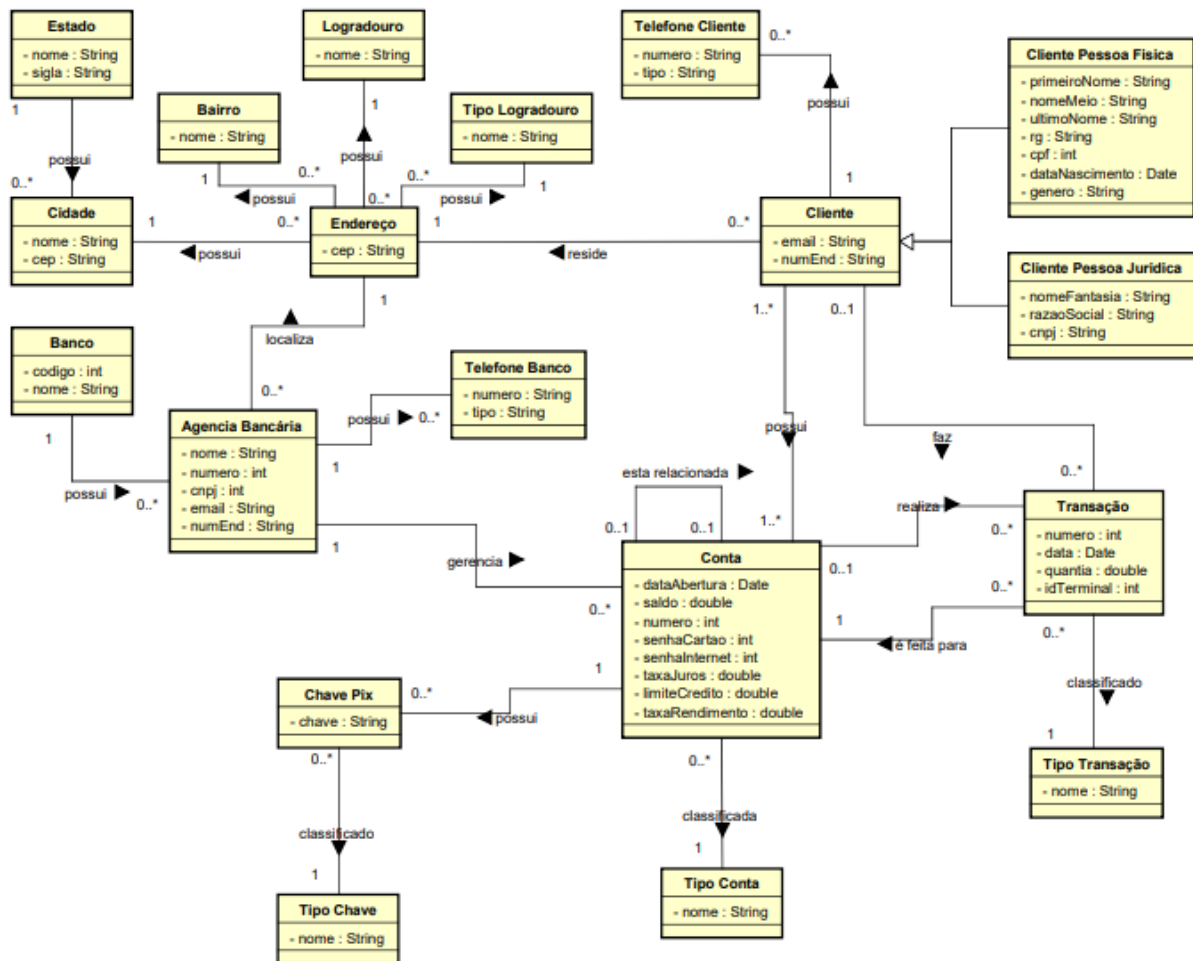
DIAGRAMA DE CLASSES	2
LISTA DE TABELAS	3
ESTRUTURA BANCO DE DADOS	4
ESTADO	4
CIDADE	5
BAIRRO	5
LOGRADOURO	5
TIPO LOGRADOURO	5
ENDEREÇO	6
BANCO	6
AGÊNCIA BANCÁRIA	6
TELEFONE BANCO	7
CLIENTE	7
CLIENTE PESSOA FÍSICA	7
CLIENTE PESSOA JURÍDICA	8
TELEFONE CLIENTE	8
CONTA	9
TIPO CONTA	9
CHAVE PIX	10
TIPO CHAVE	10
TRANSAÇÃO	10
TIPO TRANSAÇÃO	11
CONSTRUÇÃO DAS TABELAS SQL	11
ESTADO	11
CIDADE	12

1. DESCRIÇÃO

Nosso projeto tem como objetivo aplicar os conhecimentos adquiridos durante o período de um semestre na disciplina de banco de dados 2.

O projeto retrata um banco de dados utilizado para registrar e administrar os registros bancários de uma determinada região, onde é possível visualizar de uma forma geral cada cadastro, seja de clientes, transações, endereço, banco, e seus detalhes, como a conta de um cliente, o seu tipo, suas chaves pix, sua agência, entre outros detalhes.

2. DIAGRAMA DE CLASSES



3. MAPEAMENTO OBJETO-RELACIONAL

Estado (id, nome, sigla)

Cidade (id, nome, cep, #idEstado)

Bairro (id, nome)

Logradouro (id, nome)

Tipo Logradouro (id, nome)

Endereço (id, cep, #idCidade, #idBairro, #idLogradouro, #idTipoLogradouro)

Banco (id, codigo, nome)

Agência Bancária (id, numero, nome, cnpj, email, numEnd, #idBanco, #idEndereco)

Telefone Banco (id, numero, tipo, #idAgenciaBancaria)

Cliente (id, email, numEnd, #idEndereco)

Cliente Pessoa Física (id, primeiroNome, nomeMeio, ultimoNome, rg, cpf, dataNascimento, genero, #idCliente)

Cliente Pessoa Jurídica (id, nomeFantasia, razaoSocial, cnpj, #idCliente)

Telefone Cliente (id, numero, tipo, #idCliente)

Conta (id, dataAbertura, saldo, numero, senhaCartao, senhaInternet, taxaJuros, limiteCredito, taxaRendimento, #idAgenciaBancaria, #idCliente, #idConta, #idTipoConta)

Tipo Conta (id, nome)

Chave Pix (id, chave, #idConta, #idTipoChave)

Tipo Chave (id, nome)

Transação (id, numero, data, valor, idTerminal, #idTipoTransação, #idCliente, #idContaOrigem, #idContaDestino)

Tipo Transação (id, nome)

4. LISTA DE CLASSES

- ESTADO;
- CIDADE;
- BAIRRO;
- LOGRADOURO;
- TIPO LOGRADOURO;
- ENDEREÇO;
- BANCO;
- AGÊNCIA BANCÁRIA;
- TELEFONE BANCO;
- CLIENTE;
- CLIENTE PESSOA FÍSICA;
- CLIENTE PESSOA JURÍDICA;
- TELEFONE CLIENTE;
- CONTA;
- TIPO CONTA;
- CHAVE PIX;
- TIPO CHAVE;
- TRANSAÇÃO;
- TIPO TRANSAÇÃO.

5. ESTRUTURA BANCO DE DADOS

5.1. ESTADO

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim		+		
nome	varchar2(100)	Sim				1	1
sigla	varchar2(2)	Sim				2	1

5.2. CIDADE

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
nome	varchar2(200)	Sim				1	1
cep	varchar2(11)	Não				2	1
idEstado	number	Sim		Estado	id		

5.3. BAIRRO

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
nome	varchar2(200)	Sim				1	1

5.4. LOGRADOURO

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
nome	varchar2(200)	Sim				1	1

5.5. TIPO LOGRADOURO

Campo	Tipo	Obrigatório?	Chave Primária?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
nome	varchar2(200)	Sim				1	1

5.6. ENDEREÇO

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
cep	varchar2(11)	Não				1	1
idCidade	number	Sim		Cidade	id		
idBairro	number	Sim		Bairro	id		
idLogradouro	number	Sim		Logradouro	id		
idTipoLogradouro	number	Sim		Tipo Logradouro	id		

5.7. BANCO

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
código	number	Sim				1	1
nome	varchar2(100)	Sim				2	1

5.8. AGÊNCIA BANCÁRIA

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
número	number	Sim				3	2
nome	varchar2(100)	Sim				1	2
cnpj	varchar(18)	Sim				2	1
email	varchar2(100)	Sim					
numEnd	varchar2(10)	Sim					
idBanco	number	Sim		Banco	id	1/3	1
idEndereco	number	Sim		Endereco	id		

5.9. TELEFONE BANCO

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
número	varchar2(13)	Sim				1	1
tipo	varchar2(20)	Sim					
idAgencia Bancaria	number	Sim		Agência Bancária	id		

5.10. CLIENTE

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
email	varchar2(100)	Não					
numEnd	varchar2(10)	Sim					
idEndereco	number	Sim		Endereço	id		

5.11. CLIENTE PESSOA FÍSICA

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
primeiroNome	varchar2(50)	Sim					
segundoNome	varchar2(50)	Sim					
ultimoNome	varchar2(50)	Sim					
rg	varchar2(12)	Sim				1	1
cpf	number	Sim				2	1
dataNascimento	date	Sim					
genero	varchar2(20)	Sim					
idcliente	number	Sim		Cliente	id	3	1

5.12. CLIENTE PESSOA JURÍDICA

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
nomeFantasia	varchar2(100)	Sim					
razaoSocial	varchar2(100)	Sim					
cnpj	varchar2(17)	Sim				1	1
idCliente	number	Sim		Cliente	id		

5.13. TELEFONE CLIENTE

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
número	varchar2(13)	Sim				1	1
tipo	varchar2(20)	Sim					
idCliente	number	Sim		Cliente	id	1	2

5.14. CONTA

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
dataAbertura	date	Sim					
numero	number	Sim				1	2
saldo	float	Sim					
senhaCartao	number	Sim					
senhaInternet	number	Sim					
taxaJuros	float	Não					
limiteCredito	float	Não					
taxaRendimento	float	Não					
idAgenciaBancaria	number	Sim		Agência Bancária	id	1	1
idCliente	number	Sim		Cliente	id		
idConta	number	Não		Conta	id		
idTipoconta	number	Sim		Tipo Conta	id		

5.15. TIPO CONTA

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
nome	varchar2(20)	Sim				1	1

5.16. CHAVE PIX

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
chave	varchar2(100)	Sim				1	1
idConta	number	Sim		Conta	id		
idTipoChave	number	Sim		Tipo Chave	id		

5.17. TIPO CHAVE

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
nome	varchar2(100)	Sim				1	1

5.18. TRANSAÇÃO

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
número	number	Sim				1	1
data	date	Sim					
valor	double	Sim					
idTerminal	number	Não					
idTipoTransação	number	Sim		Tipo Transação	id		
idCliente	number	Não		Cliente	id		
idContaOrigem	number	Não		Conta	id		
idContaDestino	number	Sim		Conta	id		

5.19. TIPO TRANSAÇÃO

Campo	Tipo	Obrigatório?	Chave Primária ?	Chave Estrangeira		Chave Única	
				Tabela	Campo	Grupo	Ordem
id	number	Sim	Sim				
nome	varchar2(100)	Sim				1	1

6. CONSTRUÇÃO DAS TABELAS SQL

6.1. ESTADO

```
CREATE TABLE "ESTADO"
(
  "ID" NUMBER,
  "NOME" VARCHAR2(100),
  "SIGLA" VARCHAR2(2),
  CONSTRAINT "ESTADO_PK" PRIMARY KEY ("ID")
  USING INDEX ENABLE,
  CONSTRAINT "ESTADO_CK_01" CHECK (ID IS NOT NULL) ENABLE,
  CONSTRAINT "ESTADO_CK_02" CHECK (NOME IS NOT NULL) ENABLE,
  CONSTRAINT "ESTADO_CK_03" CHECK (SIGLA IS NOT NULL) ENABLE,
  CONSTRAINT "ESTADO_UK_01" UNIQUE ("NOME")
  USING INDEX ENABLE,
  CONSTRAINT "ESTADO_UK_02" UNIQUE ("SIGLA")
  USING INDEX ENABLE
)
/
```

```
CREATE OR REPLACE EDITIONABLE TRIGGER "BI_ESTADO"
before insert on "ESTADO"
for each row
begin
  if :NEW."ID" is null then
    select "ESTADO_SEQ".nextval into :NEW."ID" from sys.dual;
  end if;
end;

/
ALTER TRIGGER "BI_ESTADO" ENABLE
/
```

6.2. CIDADE

```
CREATE TABLE "CIDADE"
(
  "ID" NUMBER,
  "NOME" VARCHAR2(200),
  "ID_ESTADO" NUMBER,
  CONSTRAINT "CIDADE_PK" PRIMARY KEY ("ID")
  USING INDEX ENABLE,
  CONSTRAINT "CIDADE_CK_01" CHECK (ID IS NOT NULL) ENABLE,
  CONSTRAINT "CIDADE_CK_02" CHECK (NOME IS NOT NULL) ENABLE,
  CONSTRAINT "CIDADE_CK_03" CHECK (ID_ESTADO IS NOT NULL) ENABLE,
  CONSTRAINT "CIDADE_UK_01" UNIQUE ("NOME", "ID_ESTADO")
  USING INDEX ENABLE
```

```

    )
/
ALTER TABLE "CIDADE" ADD CONSTRAINT "CIDADE_FK_IDESTADO" FOREIGN KEY
("ID_ESTADO")
    REFERENCES "ESTADO" ("ID") ENABLE
/

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_CIDADE"
    before insert on "CIDADE"
    for each row
begin
    if :NEW."ID" is null then
        select "CIDADE_SEQ".nextval into :NEW."ID" from sys.dual;
    end if;
end;

/
ALTER TRIGGER "BI_CIDADE" ENABLE
/

```

6.3. BAIRRO

```

CREATE TABLE "BAIRRO"
(
    "ID" NUMBER,
    "NOME" VARCHAR2(100),
    CONSTRAINT "BAIRRO_PK" PRIMARY KEY ("ID")
    USING INDEX ENABLE,
    CONSTRAINT "BAIRRO_UK_01" UNIQUE ("NOME")
    USING INDEX ENABLE,
    CONSTRAINT "BAIRRO_CK_01" CHECK (ID IS NOT NULL) ENABLE,
    CONSTRAINT "BAIRRO_CK_02" CHECK (NOME IS NOT NULL) ENABLE
)
/

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_BAIRRO"
    before insert on "BAIRRO"
    for each row
begin
    if :NEW."ID" is null then
        select "BAIRRO_SEQ".nextval into :NEW."ID" from sys.dual;
    end if;
end;

/
ALTER TRIGGER "BI_BAIRRO" ENABLE
/

```

6.4. LOGRADOURO

```
CREATE TABLE "LOGRADOURO"
(
  "ID" NUMBER,
  "NOME" VARCHAR2(100),
  CONSTRAINT "LOGRADOURO_PK" PRIMARY KEY ("ID")
  USING INDEX ENABLE,
  CONSTRAINT "LOGRADOURO_CK_01" CHECK (ID IS NOT NULL) ENABLE,
  CONSTRAINT "LOGRADOURO_CK_02" CHECK (NOME IS NOT NULL) ENABLE,
  CONSTRAINT "LOGRADOURO_UK_01" UNIQUE ("NOME")
  USING INDEX ENABLE
)
/
```

```
CREATE OR REPLACE EDITIONABLE TRIGGER "BI_LOGRADOURO"
before insert on "LOGRADOURO"
for each row
begin
  if :NEW."ID" is null then
    select "LOGRADOURO_SEQ".nextval into :NEW."ID" from sys.dual;
  end if;
end;

/

ALTER TRIGGER "BI_LOGRADOURO" ENABLE

/
```

6.5. TIPO_LOGRADOURO

```
CREATE TABLE "TIPO_LOGRADOURO"
(
  "ID" NUMBER,
  "NOME" VARCHAR2(100),
  CONSTRAINT "TIPOLOGRADOURO_PK" PRIMARY KEY ("ID")
  USING INDEX ENABLE,
  CONSTRAINT "TIPO_LOGRADOURO_CK_01" CHECK (ID IS NOT NULL)
  ENABLE,
  CONSTRAINT "TIPO_LOGRADOURO_CK_02" CHECK (NOME IS NOT NULL)
  ENABLE,
  CONSTRAINT "TIPO_LOGRADOURO_UK_01" UNIQUE ("NOME")
  USING INDEX ENABLE
)
/
```



```

/

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_TIPO_LOGRADOURO"
before insert on "TIPO_LOGRADOURO"
for each row
begin
if :NEW."ID" is null then
select "TIPOLOGRADOURO_SEQ".nextval into :NEW."ID" from sys.dual;
end if;
end;

/

ALTER TRIGGER "BI_TIPO_LOGRADOURO" ENABLE

/

```

6.6. ENDEREÇO

```

CREATE TABLE "ENDERECO"
(
  "ID" NUMBER,
  "CEP" VARCHAR2(100),
  "ID_CIDADE" NUMBER,
  "ID_BAIRRO" NUMBER,
  "ID_LOGRADOURO" NUMBER,
  "ID_TIPO_LOGRADOURO" NUMBER,
  CONSTRAINT "ENDERECO_PK" PRIMARY KEY ("ID")
  USING INDEX ENABLE,
  CONSTRAINT "ENDERECO_CK_01" CHECK ( ID IS NOT NULL) ENABLE,
  CONSTRAINT "ENDERECO_UK_01" UNIQUE ("CEP")
  USING INDEX ENABLE,
  CONSTRAINT "ENDERECO_CK_02" CHECK ( ID_CIDADE IS NOT NULL)
  ENABLE,
  CONSTRAINT "ENDERECO_CK_03" CHECK ( ID_BAIRRO IS NOT NULL)
  ENABLE,
  CONSTRAINT "ENDERECO_CK_04" CHECK ( ID_LOGRADOURO IS NOT
  NULL) ENABLE,
  CONSTRAINT "ENDERECO_CK_05" CHECK ( ID_TIPO_LOGRADOURO IS
  NOT NULL) ENABLE
)
/

ALTER TABLE "ENDERECO" ADD CONSTRAINT "ENDERECO_FK_IDBAIRRO"
FOREIGN KEY ("ID_BAIRRO")
REFERENCES "BAIRRO" ("ID") ENABLE

/

ALTER TABLE "ENDERECO" ADD CONSTRAINT "ENDERECO_FK_IDCIDADE"
FOREIGN KEY ("ID_CIDADE")
REFERENCES "CIDADE" ("ID") ENABLE

```

```

/
ALTER TABLE "ENDERECO" ADD CONSTRAINT "ENDERECO_FK_IDLOGRADOURO"
FOREIGN KEY ("ID_LOGRADOURO")
    REFERENCES "LOGRADOURO" ("ID") ENABLE
/
ALTER TABLE "ENDERECO" ADD CONSTRAINT
"ENDERECO_FK_TIPOLOGRADOURO" FOREIGN KEY ("ID_TIPO_LOGRADOURO")
    REFERENCES "TIPO_LOGRADOURO" ("ID") ENABLE
/

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_ENDERECO"
before insert on "ENDERECO"
for each row
begin
    if :NEW."ID" is null then
        select "ENDERECO_SEQ".nextval into :NEW."ID" from sys.dual;
    end if;
end;

/
ALTER TRIGGER "BI_ENDERECO" ENABLE
/

```

6.7. BANCO

```

CREATE TABLE "BANCO"
(
    "ID" NUMBER,
    "CODIGO" NUMBER,
    "NOME" VARCHAR2(100),
    CONSTRAINT "BANCO_PK" PRIMARY KEY ("ID")
USING INDEX ENABLE,
    CONSTRAINT "BANCO_CK_01" CHECK (CODIGO IS NOT NULL) ENABLE,
    CONSTRAINT "BANCO_UK_01" UNIQUE ("CODIGO", "NOME")
USING INDEX ENABLE,
    CONSTRAINT "BANCO_CK_02" CHECK (NOME IS NOT NULL) ENABLE
)
/

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_BANCO"
before insert on "BANCO"
for each row
begin
    if :NEW."ID" is null then
        select "BANCO_SEQ".nextval into :NEW."ID" from sys.dual;
    end if;
end;

```

```

/
ALTER TRIGGER "BI_BANCO" ENABLE
/

```

6.8. AGENCIA_BANCARIA

```

CREATE TABLE "AGENCIA_BANCARIA"
(
    "ID" NUMBER,
    "NUMERO" NUMBER,
    "NOME" VARCHAR2(100),
    "CNPJ" VARCHAR2(18),
    "EMAIL" VARCHAR2(100),
    "NUM_END" VARCHAR2(10),
    "ID_BANCO" NUMBER,
    "ID_ENDERECO" NUMBER,
    CONSTRAINT "AGENCIA_BANCARIA_PK" PRIMARY KEY ("ID")
    USING INDEX ENABLE,
    CONSTRAINT "AGENCIA_BANCARIA_CK_01" CHECK (ID IS NOT NULL)
    ENABLE,
    CONSTRAINT "AGENCIA_BANCARIA_CK_02" CHECK (NUMERO IS NOT NULL)
    ENABLE,
    CONSTRAINT "AGENCIA_BANCARIA_CK_03" CHECK (NOME IS NOT NULL)
    ENABLE,
    CONSTRAINT "AGENCIA_BANCARIA_CK_04" CHECK (CNPJ IS NOT NULL)
    ENABLE,
    CONSTRAINT "AGENCIA_BANCARIA_CK_05" CHECK (EMAIL IS NOT NULL)
    ENABLE,
    CONSTRAINT "AGENCIA_BANCARIA_CK_06" CHECK (NUM_END IS NOT NULL)
    ENABLE,
    CONSTRAINT "AGENCIA_BANCARIA_CK_07" CHECK (ID_BANCO IS NOT
    NULL) ENABLE,
    CONSTRAINT "AGENCIA_BANCARIA_UK_01" UNIQUE ("ID_BANCO", "NOME")
    USING INDEX ENABLE,
    CONSTRAINT "AGENCIA_BANCARIA_CK_08" CHECK (ID_ENDERECO IS NOT
    NULL) ENABLE,
    CONSTRAINT "AGENCIA_BANCARIA_UK_02" UNIQUE ("CNPJ")
    USING INDEX ENABLE,
    CONSTRAINT "AGENCIA_BANCARIA_UK_03" UNIQUE ("ID_BANCO",
    "NUMERO")
    USING INDEX ENABLE
)
/

ALTER TABLE "AGENCIA_BANCARIA" ADD CONSTRAINT
"AGENCIA_BANCARIA_FK_BANCO" FOREIGN KEY ("ID_BANCO")
REFERENCES "BANCO" ("ID") ENABLE

```

```

/
ALTER TABLE "AGENCIA_BANCARIA" ADD CONSTRAINT
"AGENCIA_BANCARIA_FK_ENDERECO" FOREIGN KEY ("ID_ENDERECO")
REFERENCES "ENDERECO" ("ID") ENABLE
/

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_AGENCIA_BANCARIA"
before insert on "AGENCIA_BANCARIA"
for each row
begin
if :NEW."ID" is null then
select "AGENCIA_BANCARIA_SEQ".nextval into :NEW."ID" from sys.dual;
end if;
end;

/
ALTER TRIGGER "BI_AGENCIA_BANCARIA" ENABLE
/

```

6.9. TELEFONE_BANCO

```

CREATE TABLE "TELEFONE_BANCO"
(
"ID" NUMBER,
"NUMERO" VARCHAR2(13),
"TIPO" VARCHAR2(20),
"ID_AGENCIA_BANCARIA" NUMBER,
CONSTRAINT "TELEFONE_BANCO_PK" PRIMARY KEY ("ID")
USING INDEX ENABLE,
CONSTRAINT "TELEFONE_BANCO_CK_01" CHECK (ID IS NOT NULL) ENABLE,
CONSTRAINT "TELEFONE_BANCO_CK_02" CHECK (NUMERO IS NOT NULL)
ENABLE,
CONSTRAINT "TELEFONE_BANCO_CK_03" CHECK (TIPO IS NOT NULL)
ENABLE,
CONSTRAINT "TELEFONE_BANCO_CK_04" CHECK (ID_AGENCIA_BANCARIA
IS NOT NULL) ENABLE,
CONSTRAINT "TELEFONE_BANCO_UK_01" UNIQUE ("NUMERO")
USING INDEX ENABLE
)
/
ALTER TABLE "TELEFONE_BANCO" ADD CONSTRAINT
"TELEFONE_BANCO_FK_AGENCIA_BANCARIA" FOREIGN KEY
("ID_AGENCIA_BANCARIA")
REFERENCES "AGENCIA_BANCARIA" ("ID") ENABLE
/

```

```

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_TELEFONE_BANCO"
before insert on "TELEFONE_BANCO"
for each row
begin
if :NEW."ID" is null then
select "TELEFONE_BANCO_SEQ".nextval into :NEW."ID" from sys.dual;
end if;
end;

/
ALTER TRIGGER "BI_TELEFONE_BANCO" ENABLE
/

```

6.10. CLIENTE

```

CREATE TABLE "CLIENTE"
(
    "ID" NUMBER,
    "EMAIL" VARCHAR2(100),
    "NUM_END" VARCHAR2(10),
    "ID_ENDERECO" NUMBER,
    CONSTRAINT "CLIENTE_PK" PRIMARY KEY ("ID")
    USING INDEX ENABLE,
    CONSTRAINT "CLIENTE_CK_01" CHECK (ID IS NOT NULL) ENABLE,
    CONSTRAINT "CLIENTE_CK_02" CHECK (EMAIL IS NOT NULL) ENABLE,
    CONSTRAINT "CLIENTE_CK_03" CHECK (NUM_END IS NOT NULL) ENABLE
)
/
ALTER TABLE "CLIENTE" ADD CONSTRAINT "CLIENTE_FK" FOREIGN KEY
("ID_ENDERECO")
REFERENCES "ENDERECO" ("ID") ENABLE
/

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_CLIENTE"
before insert on "CLIENTE"
for each row
begin
if :NEW."ID" is null then
select "CLIENTE_SEQ1".nextval into :NEW."ID" from sys.dual;
end if;
end;

/
ALTER TRIGGER "BI_CLIENTE" ENABLE
/

```

6.11. CLIENTE_PESSOA_FISICA

```
CREATE TABLE "CLIENTE_PESSOA_FISICA"
(
    "ID" NUMBER,
    "PRIMEIRO_NOME" VARCHAR2(50),
    "SEGUNDO_NOME" VARCHAR2(50),
    "ULTIMO_NOME" VARCHAR2(50),
    "RG" VARCHAR2(21),
    "CPF" NUMBER,
    "DATA_NASCIMENTO" DATE,
    "GENERO" VARCHAR2(20),
    "ID_CLIENTE" NUMBER,
    CONSTRAINT "CLIENTE_PESSOA_FISICA_CK_01" CHECK ( ID IS NOT NULL)
ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_FISICA_CK_02" CHECK ( PRIMEIRO_NOME
IS NOT NULL) ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_FISICA_CK_03" CHECK ( SEGUNDO_NOME
IS NOT NULL) ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_FISICA_CK_04" CHECK ( ULTIMO_NOME IS
NOT NULL) ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_FISICA_CK_05" CHECK ( RG      IS NOT
NULL) ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_FISICA_CK_06" CHECK ( CPF IS NOT NULL)
ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_FISICA_CK_07" CHECK (
DATA_NASCIMENTO IS NOT NULL) ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_FISICA_CK_08" CHECK ( GENERO IS NOT
NULL) ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_FISICA_CK_09" CHECK ( ID_CLIENTE IS
NOT NULL) ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_FISICA_UK_01" UNIQUE ("RG")
USING INDEX ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_FISICA_UK_02" UNIQUE ("CPF")
USING INDEX ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_FISICA_UK_03" UNIQUE ("ID_CLIENTE")
USING INDEX ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_FISICA_PK" PRIMARY KEY ("ID")
USING INDEX ENABLE
)
/
ALTER TABLE "CLIENTE_PESSOA_FISICA" ADD CONSTRAINT
"CLIENTE_PESSOA_FISICA_FK" FOREIGN KEY ("ID_CLIENTE")
REFERENCES "CLIENTE" ("ID") ENABLE
/

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_CLIENTEPESSOAFISICA"
before insert on "CLIENTE_PESSOA_FISICA"
```

```

    for each row
begin
    if :NEW."ID" is null then
        select "CLIENTEPESSOA_FISICA_SEQ".nextval into :NEW."ID" from sys.dual;
    end if;
end;

/
ALTER TRIGGER "BI_CLIENTE_PESSOA_FISICA" ENABLE
/

```

6.12. CLIENTE_PESSOA_JURIDICA

```

CREATE TABLE "CLIENTE_PESSOA_JURIDICA"
(
    "ID" NUMBER,
    "NOME_FANTASIA" VARCHAR2(100),
    "RAZAO_SOCIAL" VARCHAR2(100),
    "CNPJ" VARCHAR2(17),
    "ID_CLIENTE" NUMBER,
    CONSTRAINT "CLIENTE_PESSOA_JURIDICA_CK_01" CHECK ( ID IS NOT
NULL) ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_JURIDICA_CK_02" CHECK (
NOME_FANTASIA IS NOT NULL) ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_JURIDICA_CK_03" CHECK ( RAZAO_SOCIAL
IS NOT NULL) ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_JURIDICA_CK_04" CHECK ( CNPJ IS NOT
NULL) ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_JURIDICA_CK_05" CHECK ( ID_CLIENTE IS
NOT NULL) ENABLE,
    CONSTRAINT "CLIENTE_PESSOA_JURIDICA_PK" PRIMARY KEY ("ID")
    USING INDEX ENABLE
)
/
ALTER TABLE "CLIENTE_PESSOA_JURIDICA" ADD CONSTRAINT
"CLIENTE_PESSOA_JURIDICA_FK" FOREIGN KEY ("ID_CLIENTE")
REFERENCES "CLIENTE" ("ID") ENABLE
/

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_CLIENTEPESSOAJURIDICA"
before insert on "CLIENTE_PESSOA_JURIDICA"
for each row
begin
    if :NEW."ID" is null then
        select "CLIENTEPESSOAJURIDICA_SEQ".nextval into :NEW."ID" from sys.dual;
    end if;
end;

```

```

/
ALTER TRIGGER "BI_CLIENTE_PESSOA_JURIDICA" ENABLE
/

```

6.13. TELEFONE_CLIENTE

```

CREATE TABLE "TELEFONE_CLIENTE"
(
    "ID" NUMBER,
    "NUMERO" VARCHAR2(13),
    "TIPO" VARCHAR2(20),
    "ID_CLIENTE" NUMBER,
    CONSTRAINT "TELEFONE_CLIENTE_PK" PRIMARY KEY ("ID")
USING INDEX ENABLE,
    CONSTRAINT "TELEFONE_CLIENTE_CK_01" CHECK ( ID IS NOT NULL)
ENABLE,
    CONSTRAINT "TELEFONE_CLIENTE_CK_02" CHECK ( NUMERO IS NOT NULL)
ENABLE,
    CONSTRAINT "TELEFONE_CLIENTE_CK_03" CHECK ( TIPO IS NOT NULL)
ENABLE
)
/
ALTER TABLE "TELEFONE_CLIENTE" ADD CONSTRAINT "TELEFONE_CLIENTE_FK"
FOREIGN KEY ("ID_CLIENTE")
REFERENCES "CLIENTE" ("ID") ENABLE
/

```

```

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_TELEFONECLIENTE"
before insert on "TELEFONE_CLIENTE"
for each row
begin
    if :NEW."ID" is null then
        select "TELEFONECLIENTE_SEQ".nextval into :NEW."ID" from sys.dual;
    end if;
end;
/
ALTER TRIGGER "BI_TELEFONE_CLIENTE" ENABLE
/

```

6.14. TIPO_CONTA

```

CREATE TABLE "TIPO_CONTA"
(
    "ID" NUMBER,
    "NOME" VARCHAR2(20),
    CONSTRAINT "TIPO_CONTA_PK" PRIMARY KEY ("ID")

```



```

        USING INDEX ENABLE,
        CONSTRAINT "TIPO_CONTA_CK_01" CHECK (ID IS NOT NULL) ENABLE,
        CONSTRAINT "TIPO_CONTA_CK_02" CHECK (NOME IS NOT NULL) ENABLE,
        CONSTRAINT "TIPO_CONTA_UK_01" UNIQUE ("NOME")
    USING INDEX ENABLE
)
/

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_TIPO_CONTA"
before insert on "TIPO_CONTA"
for each row
begin
    if :NEW."ID" is null then
        select "TIPO_CONTA_SEQ".nextval into :NEW."ID" from sys.dual;
    end if;
end;

/
ALTER TRIGGER "BI_TIPO_CONTA" ENABLE
/

```

6.15. CONTA

```

CREATE TABLE "CONTA"
(
    "ID" NUMBER,
    "DATA_ABERTURA" DATE,
    "NUMERO" NUMBER,
    "SALDO" FLOAT(126),
    "SENHA_CARTAO" NUMBER,
    "SENHA_INTERNET" NUMBER,
    "TAXA_JUROS" FLOAT(126),
    "LIMITE_CREDITO" FLOAT(126),
    "TAXA_RENDIMENTO" FLOAT(126),
    "ID_AGENCIA_BANCARIA" NUMBER,
    "ID_CLIENTE" NUMBER,
    "ID_CONTA" NUMBER,
    "ID_TIPO_CONTA" NUMBER,
    CONSTRAINT "CONTA_PK" PRIMARY KEY ("ID")
    USING INDEX ENABLE,
    CONSTRAINT "CONTA_CK_01" CHECK (ID IS NOT NULL) ENABLE,
    CONSTRAINT "CONTA_CK_02" CHECK (DATA_ABERTURA IS NOT NULL)
ENABLE,
    CONSTRAINT "CONTA_CK_03" CHECK (NUMERO IS NOT NULL) ENABLE,
    CONSTRAINT "CONTA_CK_04" CHECK (SALDO IS NOT NULL) ENABLE,
    CONSTRAINT "CONTA_CK_05" CHECK (SENHA_CARTAO IS NOT NULL)
ENABLE,

```

```

        CONSTRAINT "CONTA_CK_06" CHECK (SENHA_INTERNET IS NOT NULL)
ENABLE,
        CONSTRAINT "CONTA_CK_07" CHECK (ID_AGENCIA_BANCARIA IS NOT NULL)
ENABLE,
        CONSTRAINT "CONTA_CK_08" CHECK (ID_CLIENTE IS NOT NULL) ENABLE,
        CONSTRAINT "CONTA_UK_01" UNIQUE ("ID_AGENCIA_BANCARIA",
"NUMERO")
    USING INDEX ENABLE,
        CONSTRAINT "CONTA_CK_09" CHECK (ID_TIPO_CONTA IS NOT NULL)
ENABLE
    )
/
ALTER TABLE "CONTA" ADD CONSTRAINT "CONTA_FK_AGENCIA_BANCARIA"
FOREIGN KEY ("ID_AGENCIA_BANCARIA")
    REFERENCES "AGENCIA_BANCARIA" ("ID") ENABLE
/
ALTER TABLE "CONTA" ADD CONSTRAINT "CONTA_FK_CLIENTE" FOREIGN KEY
("ID_CLIENTE")
    REFERENCES "CLIENTE" ("ID") ENABLE
/
ALTER TABLE "CONTA" ADD CONSTRAINT "CONTA_FK_CONTA" FOREIGN KEY
("ID_CONTA")
    REFERENCES "CONTA" ("ID") ENABLE
/
ALTER TABLE "CONTA" ADD CONSTRAINT "CONTA_FK_TIPO_CONTA" FOREIGN KEY
("ID_TIPO_CONTA")
    REFERENCES "TIPO_CONTA" ("ID") ENABLE
/

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_CONTA"
before insert on "CONTA"
for each row
begin
    if :NEW."ID" is null then
        select "CONTA_SEQ".nextval into :NEW."ID" from sys.dual;
    end if;
end;

/
ALTER TRIGGER "BI_CONTA" ENABLE
/

```

6.16. TIPO_CHAVE

```

CREATE TABLE "TIPO_CHAVE"
(
    "ID" NUMBER,
    "NOME" VARCHAR2(100),

```

```

        CONSTRAINT "TIPO_CHAVE_PK" PRIMARY KEY ("ID")
    USING INDEX ENABLE,
        CONSTRAINT "TIPO_CHAVE_CK_01" CHECK (ID IS NOT NULL) ENABLE,
        CONSTRAINT "TIPO_CHAVE_CK_02" CHECK (NOME IS NOT NULL) ENABLE,
        CONSTRAINT "TIPO_CHAVE_UK_01" UNIQUE ("NOME")
    USING INDEX ENABLE
)
/

```

```

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_TIPO_CHAVE"
before insert on "TIPO_CHAVE"
for each row
begin
    if :NEW."ID" is null then
        select "TIPO_CHAVE_SEQ".nextval into :NEW."ID" from sys.dual;
    end if;
end;

/
ALTER TRIGGER "BI_TIPO_CHAVE" ENABLE
/

```

6.17. CHAVE_PIX

```

CREATE TABLE "CHAVE_PIX"
(
    "ID" NUMBER,
    "CHAVE" VARCHAR2(100),
    "ID_CONTA" NUMBER,
    "ID_TIPO_CHAVE" NUMBER,
    CONSTRAINT "CHAVE_PIX_PK" PRIMARY KEY ("ID")
    USING INDEX ENABLE,
    CONSTRAINT "CHAVE_PIX_CK_01" CHECK (ID IS NOT NULL) ENABLE,
    CONSTRAINT "CHAVE_PIX_CK_02" CHECK (CHAVE IS NOT NULL) ENABLE,
    CONSTRAINT "CHAVE_PIX_CK_03" CHECK (ID_CONTA IS NOT NULL) ENABLE,
    CONSTRAINT "CHAVE_PIX_CK_04" CHECK (ID_TIPO_CHAVE IS NOT NULL)
    ENABLE,
    CONSTRAINT "CHAVE_PIX_UK_01" UNIQUE ("CHAVE")
    USING INDEX ENABLE
)
/
ALTER TABLE "CHAVE_PIX" ADD CONSTRAINT "CHAVE_PIX_FK_CONTA" FOREIGN
KEY ("ID_CONTA")
REFERENCES "CONTA" ("ID") ENABLE
/
ALTER TABLE "CHAVE_PIX" ADD CONSTRAINT "CHAVE_PIX_FK_TIPO_CHAVE"
FOREIGN KEY ("ID_TIPO_CHAVE")
REFERENCES "TIPO_CHAVE" ("ID") ENABLE

```

```

/

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_CHAVE_PIX"
before insert on "CHAVE_PIX"
for each row
begin
if :NEW."ID" is null then
select "CHAVE_PIX_SEQ".nextval into :NEW."ID" from sys.dual;
end if;
end;

/

ALTER TRIGGER "BI_CHAVE_PIX" ENABLE

/

```

6.18. TIPO_TRANSACAO

```

CREATE TABLE "TIPO_TRANSACAO"
(
  "ID" NUMBER,
  "NOME" VARCHAR2(100),
  CONSTRAINT "TIPO_TRANSACAO_PK" PRIMARY KEY ("ID")
USING INDEX ENABLE,
  CONSTRAINT "TIPO_TRANSACAO_CK_01" CHECK (ID IS NOT NULL) ENABLE,
  CONSTRAINT "TIPO_TRANSACAO_CK_02" CHECK (NOME IS NOT NULL)
ENABLE,
  CONSTRAINT "TIPO_TRANSACAO_UK_01" UNIQUE ("NOME")
USING INDEX ENABLE
)
/

```

```

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_TIPO_TRANSACAO"
before insert on "TIPO_TRANSACAO"
for each row
begin
if :NEW."ID" is null then
select "TIPO_TRANSACAO_SEQ".nextval into :NEW."ID" from sys.dual;
end if;
end;

/

ALTER TRIGGER "BI_TIPO_TRANSACAO" ENABLE

/

```

6.19. TRANSACAO

```

CREATE TABLE "TRANSACAO"

```

```

(  "ID" NUMBER,
    "NUMERO" NUMBER,
    "DATA" DATE,
    "VALOR" FLOAT(126),
    "ID_TERMINAL" NUMBER,
    "ID_TIPO_TRANSACAO" NUMBER,
    "ID_CLIENTE" NUMBER,
    "ID_CONTA_ORIGEM" NUMBER,
    "ID_CONTA_DESTINO" NUMBER,
    CONSTRAINT "TRANSACAO_PK" PRIMARY KEY ("ID")
USING INDEX ENABLE,
    CONSTRAINT "TRANSACAO_UK_01" UNIQUE ("NUMERO")
USING INDEX ENABLE,
    CONSTRAINT "TRANSACAO_CK_01" CHECK (ID IS NOT NULL) ENABLE,
    CONSTRAINT "TRANSACAO_CK_02" CHECK (NUMERO IS NOT NULL) ENABLE,
    CONSTRAINT "TRANSACAO_CK_03" CHECK (DATA IS NOT NULL) ENABLE,
    CONSTRAINT "TRANSACAO_CK_04" CHECK (VALOR IS NOT NULL) ENABLE,
    CONSTRAINT "TRANSACAO_CK_05" CHECK (ID_TIPO_TRANSACAO IS NOT
NULL) ENABLE,
    CONSTRAINT "TRANSACAO_CK_06" CHECK (ID_CONTA_DESTINO IS NOT
NULL) ENABLE
)
/
ALTER TABLE "TRANSACAO" ADD CONSTRAINT "TRANSACAO_FK_CLIENTE"
FOREIGN KEY ("ID_CLIENTE")
REFERENCES "CLIENTE" ("ID") ENABLE
/
ALTER TABLE "TRANSACAO" ADD CONSTRAINT
"TRANSACAO_FK_CONTA_DESTINO" FOREIGN KEY ("ID_CONTA_DESTINO")
REFERENCES "CONTA" ("ID") ENABLE
/
ALTER TABLE "TRANSACAO" ADD CONSTRAINT "TRANSACAO_FK_CONTA_ORIGEM"
FOREIGN KEY ("ID_CONTA_ORIGEM")
REFERENCES "CONTA" ("ID") ENABLE
/
ALTER TABLE "TRANSACAO" ADD CONSTRAINT
"TRANSACAO_FK_TIPO_TRANSACAO" FOREIGN KEY ("ID_TIPO_TRANSACAO")
REFERENCES "TIPO_TRANSACAO" ("ID") ENABLE
/

CREATE OR REPLACE EDITIONABLE TRIGGER "BI_TRANSACAO"
before insert on "TRANSACAO"
for each row
begin
if :NEW."ID" is null then
select "TRANSACAO_SEQ".nextval into :NEW."ID" from sys.dual;
end if;
end;

```

```
/  
ALTER TRIGGER "BI_TRANSACAO" ENABLE  
/
```

7. PACKAGES

- 7.1. ESTADO
- 7.2. CIDADE
- 7.3. BAIRRO
- 7.4. LOGRADOURO
- 7.5. TIPO LOGRADOURO
- 7.6. ENDEREÇO
- 7.7. BANCO
- 7.8. AGÊNCIA BANCÁRIA
- 7.9. TELEFONE BANCO
- 7.10. CLIENTE

```
create or replace PACKAGE PKG_CLIENTE AS
/* TODO enter package declarations (types, exceptions, methods etc) here */
vr_cliente cliente%ROWTYPE;
PROCEDURE PR_INCLUIR      ( pr_cliente IN cliente%ROWTYPE);
PROCEDURE PR_ALTERAR      ( pr_cliente IN cliente%ROWTYPE );
PROCEDURE PR_EXCLUIR      ( pn_id      IN number );
FUNCTION FN_BUSCAR        ( pn_id      IN number
                           ) return CLIENTE%ROWTYPE;
END PKG_CLIENTE;
```

7.11. CLIENTE PESSOA FÍSICA

```
create or replace PACKAGE PKG_CLIENTE_PESSOA_FISICA AS
/* TODO enter package declarations (types, exceptions, methods etc) here */
vr_cliente_Pessoa_Fisica CLIENTE_PESSOA_FISICA%ROWTYPE;
PROCEDURE PR_INCLUIR      ( pr_cliente_Pessoa_Fisica IN
CLIENTE_PESSOA_FISICA%ROWTYPE,
                           pv_erro      OUT varchar2
                           );
PROCEDURE PR_ALTERAR      ( pr_cliente_Pessoa_Fisica IN
CLIENTE_PESSOA_FISICA%ROWTYPE );
PROCEDURE PR_EXCLUIR      ( pn_id      IN number );
FUNCTION FN_BUSCAR        ( pv_conteudo IN NUMBER
                           ) return CLIENTE_PESSOA_FISICA%ROWTYPE;
FUNCTION FN_BUSCAR        ( pn_id      IN NUMBER
                           ) return CLIENTE_PESSOA_FISICA%ROWTYPE;
FUNCTION FN_VERIFICAREXISTE ( pn_conteudo IN NUMBER
                           ) return boolean;
END PKG_CLIENTE_PESSOA_FISICA;
```

7.12. CLIENTE PESSOA JURÍDICA

```
create or replace PACKAGE PKG_CLIENTE_PESSOA_JURIDICA AS
```

```

/* TODO enter package declarations (types, exceptions, methods etc) here */
vr_CLIENTE_PESSOA_JURIDICA CLIENTE_PESSOA_JURIDICA%ROWTYPE;
PROCEDURE PR_INCLUIR      ( pr_CLIENTE_PESSOA_JURIDICA IN
CLIENTE_PESSOA_JURIDICA%ROWTYPE,
                        pv_erro      OUT varchar2
                        );
PROCEDURE PR_ALTERAR      ( pr_CLIENTE_PESSOA_JURIDICA IN
CLIENTE_PESSOA_JURIDICA%ROWTYPE );
PROCEDURE PR_EXCLUIR      ( pn_id      IN number );
FUNCTION FN_BUSCAR        ( pv_conteudo IN VARCHAR2
                        ) return CLIENTE_PESSOA_JURIDICA%ROWTYPE;
FUNCTION FN_BUSCAR        ( pn_id IN NUMBER
                        ) return CLIENTE_PESSOA_JURIDICA%ROWTYPE;
FUNCTION FN_VERIFICAREXISTE ( pv_conteudo IN VARCHAR2
                        ) return boolean;
END PKG_CLIENTE_PESSOA_JURIDICA;

```

7.13. TELEFONE CLIENTE

7.14. CONTA

```

create or replace PACKAGE PKG_CONTA AS
/* TODO enter package declarations (types, exceptions, methods etc) here */
vr_CONTA CONTA%ROWTYPE;
PROCEDURE PR_INCLUIR      ( pr_CONTA IN CONTA%ROWTYPE,
                        pv_erro      OUT varchar2
                        );
PROCEDURE PR_ALTERAR      ( pr_CONTA IN CONTA%ROWTYPE );
PROCEDURE PR_EXCLUIR      ( pn_id      IN number );
FUNCTION FN_BUSCAR        ( pn_conteudo IN NUMBER
                        ) return CONTA%ROWTYPE;
FUNCTION FN_BUSCAR        ( pn_id IN NUMBER
                        ) return CONTA%ROWTYPE;
FUNCTION FN_VERIFICAREXISTE ( pn_numero IN NUMBER,
                        pn_agencia IN NUMBER
                        ) return boolean;
END PKG_CONTA;

```

7.15. TIPO CONTA

7.16. CHAVE PIX

7.17. TIPO CHAVE

7.18. TRANSAÇÃO

```

create or replace PACKAGE PKG_TRANSACAO AS
/* TODO enter package declarations (types, exceptions, methods etc) here */
vr_TRANSACAO TRANSACAO%ROWTYPE;
PROCEDURE PR_INCLUIR      ( pr_TRANSACAO IN TRANSACAO%ROWTYPE
                        );

```



```

FUNCTION FN_BUSCAR      ( pn_numero  IN NUMBER
                        ) return TRANSACAO%ROWTYPE;
FUNCTION FN_BUSCAR      ( pn_id    IN NUMBER
                        ) return TRANSACAO%ROWTYPE;
END PKG_TRANSACAO;

```

7.19. TIPO TRANSAÇÃO

8. PROCEDURES

8.1. RENDIMENTO DA CONTA POUPANÇA

```

-- PROCEDIMENTO PARA RENDER O SALDO DA CONTA POUPANÇA EM (0.5%)
create or replace procedure PR_RENDER_CONTA_POUPANCA
is
begin
    UPDATE CONTA SET SALDO = (SALDO * 1.005) WHERE CONTA.ID > 0;
end PR_RENDER_CONTA_POUPANCA;

```

8.2. CADASTRO PESSOA FÍSICA

```

create or replace PROCEDURE PR_CAD_CLIENTE_PESSOA_FISICA (
pr_CLIENTE_PESSOA_FISICA IN CLIENTE_PESSOA_FISICA%ROWTYPE,
                                pr_CLIENTE IN CLIENTE%ROWTYPE,
                                pv_erro OUT VARCHAR2) IS
vb_existe BOOLEAN;
begin
    declare
    begin
        vb_existe := PKG_CLIENTE_PESSOA_FISICA.FN_VERIFICAREXISTE(pn_conteudo
=> pr_CLIENTE_PESSOA_FISICA.CPF);
        IF vb_existe = FALSE THEN
            PKG_CLIENTE.PR_INCLUIR(pr_cliente => pr_CLIENTE);
            PKG_CLIENTE_PESSOA_FISICA.PR_INCLUIR(pr_CLIENTE_PESSOA_FISICA =>
pr_CLIENTE_PESSOA_FISICA, pv_erro => pv_erro);
        END IF;
    END;
end PR_CAD_CLIENTE_PESSOA_FISICA;

```

8.3. CADASTRO PESSOA JURÍDICA

```

create or replace PROCEDURE PR_CAD_CLIENTE_PESSOA_JURIDICA (
pr_CLIENTE_PESSOA_JURIDICA IN CLIENTE_PESSOA_JURIDICA%ROWTYPE,
                                pr_CLIENTE IN CLIENTE%ROWTYPE,
                                pv_erro OUT VARCHAR2) IS
vb_existe BOOLEAN;
begin
    declare
    begin
        vb_existe :=
PKG_CLIENTE_PESSOA_JURIDICA.FN_VERIFICAREXISTE(pv_conteudo =>
pr_CLIENTE_PESSOA_JURIDICA.CNPJ);
        IF vb_existe = FALSE THEN
            PKG_CLIENTE.PR_INCLUIR(pr_cliente => pr_CLIENTE);

PKG_CLIENTE_PESSOA_JURIDICA.PR_INCLUIR(pr_CLIENTE_PESSOA_JURIDICA =>
pr_CLIENTE_PESSOA_JURIDICA, pv_erro => pv_erro);
        END IF;
    END;
end PR_CAD_CLIENTE_PESSOA_JURIDICA;

```

8.4. DEPOSITO

```

create or replace PROCEDURE PR_DEPOSITO ( pn_NUMERO IN varchar2,pd_DATE IN
DATE,pn_Valor IN number,
                                pn_ID_TERMINAL IN number,pn_ID_CLIENTE IN number
,pn_ID_CONTA_ORIGEM IN number,pn_ID_CONTA_DESTINO IN number
) IS

```

```

begin
insert into TRANSACAO
(NUMERO,DATA,VALOR,ID_TERMINAL,ID_TIPO_TRANSACAO,ID_CLIENTE,ID_CONTA_
ORIGEM,ID_CONTA_DESTINO) values
(pn_NUMERO,pd_DATE,pn_Valor,pn_ID_TERMINAL,2,pn_ID_CLIENTE,pn_ID_CONTA_O
RIGEM,NULL);
update CONTA set SALDO = SALDO + pn_Valor where ID = pn_ID_CONTA_ORIGEM;
end PR_DEPOSITO;

```

8.5. ENVIAR EMAIL AUTOMATICO

```

create or replace PROCEDURE PR_ENVIAREMAILAUT ( pv_smtp IN varchar2,pv_usuario
IN varchar2,
                                pv_senha IN varchar2,pv_remetente IN varchar2,
                                pv_destinatario IN varchar2, pv_assunto IN varchar2,
                                pc_mensagem IN clob) IS
pv_conn utl_smtp.connection;
begin
pv_conn := utl_smtp.open_connection (pv_smtp,587);
utl_smtp.ehlo(pv_conn,pv_smtp);
utl_smtp.command (pv_conn, 'AUTH LOGIN');

utl_smtp.command
(pv_conn,utl_raw.cast_to_varchar2(utl_encode.base64_encode(utl_raw.cast_to_raw(pv_usu
ario))));
utl_smtp.command
(pv_conn,utl_raw.cast_to_varchar2(utl_encode.base64_encode(utl_raw.cast_to_raw(pv_sen
ha))));

utl_smtp.mail (pv_conn,('<' ||pv_remetente||'>'));
utl_smtp.rcpt (pv_conn,('<' ||pv_destinatario||'>'));

utl_smtp.open_data(pv_conn);
utl_smtp.write_raw_data(pv_conn, utl_raw.cast_to_raw('Content-Type:
Text/Html;charset=UTF=8'||utl_tcp.crlf));
utl_smtp.write_raw_data(pv_conn, utl_raw.cast_to_raw('From:' ||pv_remetente||utl_tcp.crlf));
utl_smtp.write_raw_data(pv_conn, utl_raw.cast_to_raw('To:' ||pv_destinatario||utl_tcp.crlf));
utl_smtp.write_raw_data(pv_conn, utl_raw.cast_to_raw('Subject:' ||pv_assunto||utl_tcp.crlf));
utl_smtp.write_raw_data(pv_conn, utl_raw.cast_to_raw(''||utl_tcp.crlf));
utl_smtp.write_raw_data(pv_conn, utl_raw.cast_to_raw(utl_tcp.crlf||pc_mensagem));

utl_smtp.CLOSE_DATA(pv_conn);
utl_smtp.QUIT(pv_conn);
EXCEPTION
WHEN OTHERS THEN
utl_smtp.quit (pv_conn);

```

```
RAISE_APPLICATION_ERROR(-20011,'Não foi Possivel enviar o Email devido ao seguinte
erro: '||sqlerrm);
end;
```

8.6. MOVIMENTA SALDO

```
create or replace PROCEDURE PR_MOVIMENTA_SALDO (pr_TRANSACAO IN
TRANSACAO%ROWTYPE) IS
```

```
vc_conta          CONTA%ROWTYPE;
vc_conta_origem   CONTA%ROWTYPE;
vc_conta_destino   CONTA%ROWTYPE;
begin
    IF pr_TRANSACAO.ID_TIPO_TRANSACAO = 2 THEN

        PKG_TRANSACAO.PR_INCLUIR( pr_TRANSACAO => pr_TRANSACAO);

        vc_conta := PKG_CONTA.FN_BUSCAR(pn_id =>
pr_TRANSACAO.ID_CONTA_ORIGEM);
        vc_conta.SALDO := vc_conta.SALDO + pr_TRANSACAO.VALOR;

        PKG_CONTA.PR_ALTERAR(pr_CONTA => vc_conta);
        --UPDATE CONTA SET SALDO = SALDO + pr_TRANSACAO.VALOR where ID =
pr_TRANSACAO.ID_CONTA_ORIGEM;
        ELSIF pr_TRANSACAO.ID_TIPO_TRANSACAO = 1 THEN

            PKG_TRANSACAO.PR_INCLUIR(pr_TRANSACAO => pr_TRANSACAO);

            vc_conta := PKG_CONTA.FN_BUSCAR(pn_id =>
pr_TRANSACAO.ID_CONTA_DESTINO);
            vc_conta.SALDO := vc_conta.SALDO - pr_TRANSACAO.VALOR;

            PKG_CONTA.PR_ALTERAR(pr_CONTA => vc_conta);
            --UPDATE CONTA SET SALDO = SALDO - pr_TRANSACAO.VALOR where ID =
pr_TRANSACAO.ID_CONTA_DESTINO;
            ELSIF pr_TRANSACAO.ID_TIPO_TRANSACAO = 3 THEN

                PKG_TRANSACAO.PR_INCLUIR(pr_TRANSACAO => pr_TRANSACAO);

                vc_conta_origem := PKG_CONTA.FN_BUSCAR(pn_id =>
pr_TRANSACAO.ID_CONTA_DESTINO);
                vc_conta_origem.SALDO := vc_conta_origem.SALDO - pr_TRANSACAO.VALOR;

                PKG_CONTA.PR_ALTERAR(pr_CONTA => vc_conta_origem);
                --UPDATE CONTA SET SALDO = SALDO - pr_TRANSACAO.VALOR where ID =
pr_TRANSACAO.ID_CONTA_ORIGEM;
                vc_conta_destino := PKG_CONTA.FN_BUSCAR(pn_id =>
pr_TRANSACAO.ID_CONTA_DESTINO);
                vc_conta_destino.SALDO := vc_conta_destino.SALDO + pr_TRANSACAO.VALOR;
```

```

        PKG_CONTA.PR_ALTERAR(pr_CONTA => vc_conta_destino);
        --UPDATE CONTA SET SALDO = SALDO + pr_TRANSACAO.VALOR where ID =
pr_TRANSACAO.ID_CONTA_DESTINO;
    END IF;
end PR_MOVIMENTA_SALDO;

```

8.7. SAQUE

```

create or replace PROCEDURE PR_SAQUE ( pn_NUMERO IN varchar2,pd_DATE IN
DATE,pn_Valor IN number,
                                pn_ID_TERMINAL IN number,pn_ID_CLIENTE IN number
,pn_ID_CONTA_ORIGEM IN number,pn_ID_CONTA_DESTINO IN number
) IS

begin
insert into TRANSACAO
(NUMERO,DATA,VALOR,ID_TERMINAL,ID_TIPO_TRANSACAO,ID_CLIENTE,ID_CONTA_
ORIGEM,ID_CONTA_DESTINO) values
(pn_NUMERO,pd_DATE,pn_Valor,pn_ID_TERMINAL,1,pn_ID_CLIENTE,pn_ID_CONTA_O
RIGEM,NULL);
update CONTA set SALDO = SALDO - pn_Valor where ID = pn_ID_CONTA_DESTINO;
end PR_SAQUE;

```

8.8. TRANSFERÊNCIA

```

create or replace PROCEDURE PR_TRANSFERENCIA ( pn_NUMERO IN
varchar2,pd_DATE IN DATE,pn_Valor IN number,
                                pn_ID_TERMINAL IN number,pn_ID_CLIENTE IN number
,pn_ID_CONTA_ORIGEM IN number,
                                pn_ID_CONTA_DESTINO IN number
) IS

begin
insert into TRANSACAO
(NUMERO,DATA,VALOR,ID_TERMINAL,ID_TIPO_TRANSACAO,ID_CLIENTE,ID_CONTA_
ORIGEM,ID_CONTA_DESTINO) values
(pn_NUMERO,pd_DATE,pn_Valor,pn_ID_TERMINAL,3,pn_ID_CLIENTE,pn_ID_CONTA_O
RIGEM,pn_ID_CONTA_DESTINO);
update CONTA set SALDO = SALDO - pn_Valor where ID = pn_ID_CONTA_ORIGEM;
update CONTA set SALDO = SALDO + pn_Valor where ID = pn_ID_CONTA_DESTINO;

end PR_TRANSFERENCIA;

```

9. JOBS

9.1. RENDIMENTO DA CONTA POUPANÇA

```
BEGIN
DBMS_SCHEDULER.CREATE_JOB
(
    JOB_NAME => 'RENDIMENTO_POUPANÇA',
    JOB_TYPE => 'STORED_PROCEDURE',
    JOB_ACTION => 'PR_RENDER_CONTA_POUPANCA',
    START_DATE => SYSDATE,
    REPEAT_INTERVAL => 'FREQ=MONTHLY;INTERVAL=1',
    ENABLED => TRUE,
    COMMENTS => 'JOB PARA RENDER AS CONTAS POUPANÇAS '
);
END;
```

10. FUNÇÕES

10.1. VALIDAÇÃO DO CPF

```
create or replace FUNCTION FN_VALIDACPF(CPF VARCHAR2 )
RETURN VARCHAR2 IS
cpf_c number := lpad(cpf,11,'0');
tam number(2):= length(substr(cpf,-2,2));
tot1 number(4) := 0;
tot2 number(4) := 0;
multiplicador number(2) := 9;
digito number(1) := 0;
resto1 number(2) := 0;
resto2 number(2) := 0;
dc1_cpf number(1) := 0;
dc2_cpf number(1) := 0;
digito_dig char(2) := substr(cpf,-2,2);
cpf_x number := substr(lpad(to_char(cpf_c),11,'0'),1,9);
tamanho number(2) := length(to_char(cpf_x));
dc_cpf char(2);
BEGIN
    for i in reverse 1..tamanho loop
        tot1 := tot1 + multiplicador *
            (substr(to_char(cpf_x),i,1));
        multiplicador := multiplicador - 1;
    end loop;
    resto1 := mod(tot1,11);
    if resto1 = 10 then
        dc1_cpf := 0;
    else
        dc1_cpf := resto1;
    end if;
```

```

end if;
digito := dc1_cpf;
tot2 := tot2 + 9 * digito;
multiplicador := 8;
for i in reverse 1..tamanho loop
    tot2 := tot2 + multiplicador *
        (substr(to_char(cpf_x),i,1));
    multiplicador := multiplicador - 1;
end loop;
resto2 := mod(tot2,11);
if resto2 = 10 then
    dc2_cpf := 0;
else
    dc2_cpf := resto2;
end if;
dc_cpf := to_char(dc1_cpf) || to_char(dc2_cpf);
if dc_cpf = digito_dig then
    return 'CPF VALIDO';
else
    return 'CPF INVALIDO';
end if;
END FN_VALIDACPF;

```

10.2. GERAR CHAVE PIX ALEATÓRIA

create or replace function "FN_CHAVEPIX_ALEATORIA"
 (pv_valor in VARCHAR2)
 return VARCHAR2 is

```

pv_hexkey varchar2(32) :=null;

```

```

begin
pv_hexkey := rawtohex(DBMS_OBFUSCATION_TOOLKIT.md5(input =>
    UTL_RAW.cast_to_raw(pv_valor)));
return nvl(pv_hexkey,"");
end;

```

11. SELECTS

11.1. EXTRATO PF 7 DIAS

```

SELECT CONTA.NUMERO AS "CONTA",
    AGENCIA_BANCARIA.NUMERO AS "AGÊNCIA BANCÁRIA",

```

```

    CLIENTE_PESSOA_FISICA.PRIMEIRO_NOME || ' ' ||
    CLIENTE_PESSOA_FISICA.SEGUNDO_NOME || ' ' ||
    CLIENTE_PESSOA_FISICA.ULTIMO_NOME AS "NOME COMPLETO",
    TRANSACAO.NUMERO AS "NUMERO TRANSAÇÃO",
    TRANSACAO.DATA,
    TRANSACAO.VALOR,
    TRANSACAO.ID_TERMINAL AS "TERMINAL",
    TIPO_TRANSACAO.NOME AS "TIPO DE TRANSAÇÃO",
    TRANSACAO.ID_CONTA_ORIGEM AS "CONTA DE ORIGEM",
    TRANSACAO.ID_CONTA_DESTINO AS "CONTA DE DESTINO",
    CONTA.SALDO
FROM CONTA,
    AGENCIA_BANCARIA,
    CLIENTE_PESSOA_FISICA,
    TRANSACAO,
    TIPO_TRANSACAO
WHERE CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID
    AND CONTA.ID_CLIENTE = CLIENTE_PESSOA_FISICA.ID
    AND TRANSACAO.ID_CLIENTE = CLIENTE_PESSOA_FISICA.ID
    AND TIPO_TRANSACAO.ID = TRANSACAO.ID_TIPO_TRANSACAO
    AND TRANSACAO.DATA > '07-12-2021'

```

11.2. EXTRATO PF 15 DIAS

```

SELECT CONTA.NUMERO AS "CONTA",
    AGENCIA_BANCARIA.NUMERO AS "AGÊNCIA BANCÁRIA",
    CLIENTE_PESSOA_FISICA.PRIMEIRO_NOME || ' ' ||
    CLIENTE_PESSOA_FISICA.SEGUNDO_NOME || ' ' ||
    CLIENTE_PESSOA_FISICA.ULTIMO_NOME AS "NOME COMPLETO",
    TRANSACAO.NUMERO AS "NUMERO TRANSAÇÃO",
    TRANSACAO.DATA,
    TRANSACAO.VALOR,
    TRANSACAO.ID_TERMINAL AS "TERMINAL",
    TIPO_TRANSACAO.NOME AS "TIPO DE TRANSAÇÃO",
    TRANSACAO.ID_CONTA_ORIGEM AS "CONTA DE ORIGEM",
    TRANSACAO.ID_CONTA_DESTINO AS "CONTA DE DESTINO",
    CONTA.SALDO
FROM CONTA,
    AGENCIA_BANCARIA,
    CLIENTE_PESSOA_FISICA,
    TRANSACAO,
    TIPO_TRANSACAO
WHERE CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID
    AND CONTA.ID_CLIENTE = CLIENTE_PESSOA_FISICA.ID
    AND TRANSACAO.ID_CLIENTE = CLIENTE_PESSOA_FISICA.ID
    AND TIPO_TRANSACAO.ID = TRANSACAO.ID_TIPO_TRANSACAO
    AND TRANSACAO.DATA > '29-11-2021'

```


11.3. EXTRATO PF 30 DIAS

```
SELECT CONTA.NUMERO AS "CONTA",
       AGENCIA_BANCARIA.NUMERO AS "AGÊNCIA BANCÁRIA",
       CLIENTE_PESSOA_FISICA.PRIMEIRO_NOME || ' ' ||
CLIENTE_PESSOA_FISICA.SEGUNDO_NOME || ' ' ||
CLIENTE_PESSOA_FISICA.ULTIMO_NOME AS "NOME COMPLETO",
       TRANSACAO.NUMERO AS "NUMERO TRANSAÇÃO",
       TRANSACAO.DATA,
       TRANSACAO.VALOR,
       TRANSACAO.ID_TERMINAL AS "TERMINAL",
       TIPO_TRANSACAO.NOME AS "TIPO DE TRANSAÇÃO",
       TRANSACAO.ID_CONTA_ORIGEM AS "CONTA DE ORIGEM",
       TRANSACAO.ID_CONTA_DESTINO AS "CONTA DE DESTINO",
       CONTA.SALDO
FROM CONTA,
     AGENCIA_BANCARIA,
     CLIENTE_PESSOA_FISICA,
     TRANSACAO,
     TIPO_TRANSACAO
WHERE CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID
      AND CONTA.ID_CLIENTE = CLIENTE_PESSOA_FISICA.ID
      AND TRANSACAO.ID_CLIENTE = CLIENTE_PESSOA_FISICA.ID
      AND TIPO_TRANSACAO.ID = TRANSACAO.ID_TIPO_TRANSACAO
      AND TRANSACAO.DATA > '14-11-2021'
```

11.4. EXTRATO PJ 7 DIAS

```
SELECT CONTA.NUMERO AS "CONTA",
       AGENCIA_BANCARIA.NUMERO AS "AGÊNCIA BANCÁRIA",
       CLIENTE_PESSOA_JURIDICA.NOME_FANTASIA AS "NOME FANTASIA",
       TRANSACAO.NUMERO AS "NUMERO TRANSAÇÃO",
       TRANSACAO.DATA,
       TRANSACAO.VALOR,
       TRANSACAO.ID_TERMINAL AS "TERMINAL",
       TIPO_TRANSACAO.NOME AS "TIPO DE TRANSAÇÃO",
       TRANSACAO.ID_CONTA_ORIGEM AS "CONTA DE ORIGEM",
       TRANSACAO.ID_CONTA_DESTINO AS "CONTA DE DESTINO",
       CONTA.SALDO
FROM CONTA,
     AGENCIA_BANCARIA,
     CLIENTE_PESSOA_JURIDICA,
     TRANSACAO,
     TIPO_TRANSACAO
WHERE CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID
      AND CONTA.ID_CLIENTE = CLIENTE_PESSOA_JURIDICA.ID_CLIENTE
      AND TRANSACAO.ID_CLIENTE = CLIENTE_PESSOA_JURIDICA.ID_CLIENTE
```

```
AND TIPO_TRANSACAO.ID = TRANSACAO.ID_TIPO_TRANSACAO
AND TRANSACAO.DATA > '07-12-2021'
```

11.5. EXTRATO PJ 15 DIAS

```
SELECT CONTA.NUMERO AS "CONTA",
       AGENCIA_BANCARIA.NUMERO AS "AGÊNCIA BANCÁRIA",
       CLIENTE_PESSOA_JURIDICA.NOME_FANTASIA AS "NOME FANTASIA",
       TRANSACAO.NUMERO AS "NUMERO TRANSAÇÃO",
       TRANSACAO.DATA,
       TRANSACAO.VALOR,
       TRANSACAO.ID_TERMINAL AS "TERMINAL",
       TIPO_TRANSACAO.NOME AS "TIPO DE TRANSAÇÃO",
       TRANSACAO.ID_CONTA_ORIGEM AS "CONTA DE ORIGEM",
       TRANSACAO.ID_CONTA_DESTINO AS "CONTA DE DESTINO",
       CONTA.SALDO
FROM CONTA,
     AGENCIA_BANCARIA,
     CLIENTE_PESSOA_JURIDICA,
     TRANSACAO,
     TIPO_TRANSACAO
WHERE CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID
      AND CONTA.ID_CLIENTE = CLIENTE_PESSOA_JURIDICA.ID_CLIENTE
      AND TRANSACAO.ID_CLIENTE = CLIENTE_PESSOA_JURIDICA.ID_CLIENTE
      AND TIPO_TRANSACAO.ID = TRANSACAO.ID_TIPO_TRANSACAO
      AND TRANSACAO.DATA > '29-11-2021'
```

11.6. EXTRATO PJ 30 DIAS

```
SELECT CONTA.NUMERO AS "CONTA",
       AGENCIA_BANCARIA.NUMERO AS "AGÊNCIA BANCÁRIA",
       CLIENTE_PESSOA_JURIDICA.NOME_FANTASIA AS "NOME FANTASIA",
       TRANSACAO.NUMERO AS "NUMERO TRANSAÇÃO",
       TRANSACAO.DATA,
       TRANSACAO.VALOR,
       TRANSACAO.ID_TERMINAL AS "TERMINAL",
       TIPO_TRANSACAO.NOME AS "TIPO DE TRANSAÇÃO",
       TRANSACAO.ID_CONTA_ORIGEM AS "CONTA DE ORIGEM",
       TRANSACAO.ID_CONTA_DESTINO AS "CONTA DE DESTINO",
       CONTA.SALDO
FROM CONTA,
     AGENCIA_BANCARIA,
     CLIENTE_PESSOA_JURIDICA,
     TRANSACAO,
     TIPO_TRANSACAO
WHERE CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID
      AND CONTA.ID_CLIENTE = CLIENTE_PESSOA_JURIDICA.ID_CLIENTE
```

```

AND TRANSACAO.ID_CLIENTE = CLIENTE_PESSOA_JURIDICA.ID_CLIENTE
AND TIPO_TRANSACAO.ID = TRANSACAO.ID_TIPO_TRANSACAO
AND TRANSACAO.DATA > '14-11-2021'

```

11.7. SALDO PF

```

SELECT CONTA.NUMERO AS "NÚMERO DA CONTA",
       AGENCIA_BANCARIA.NUMERO "NÚMERO DA AGÊNCIA",
       CLIENTE_PESSOA_FISICA.PRIMEIRO_NOME || ' ' ||
CLIENTE_PESSOA_FISICA.SEGUNDO_NOME || ' ' ||
CLIENTE_PESSOA_FISICA.ULTIMO_NOME AS "NOME COMPLETO",
CONTA.SALDO
FROM CONTA,
     AGENCIA_BANCARIA,
     CLIENTE_PESSOA_FISICA
WHERE CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID
      AND CONTA.ID_CLIENTE = CLIENTE_PESSOA_FISICA.ID

```

11.8. SALDO PJ

```

SELECT CONTA.NUMERO AS "NÚMERO DA CONTA",
       AGENCIA_BANCARIA.NUMERO "NÚMERO DA AGÊNCIA",
       CLIENTE_PESSOA_JURIDICA.NOME_FANTASIA AS "NOME FANTASIA",
CONTA.SALDO
FROM CONTA,
     AGENCIA_BANCARIA,
     CLIENTE_PESSOA_JURIDICA
WHERE CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID
      AND CONTA.ID_CLIENTE = CLIENTE_PESSOA_JURIDICA.ID_CLIENTE

```

11.9. LISTA DE CLIENTES PF EM ORDEM ALFABÉTICA

```

SELECT CLIENTE_PESSOA_FISICA.PRIMEIRO_NOME || ' ' ||
CLIENTE_PESSOA_FISICA.SEGUNDO_NOME || ' ' ||
CLIENTE_PESSOA_FISICA.ULTIMO_NOME AS "NOME COMPLETO",
CONTA.NUMERO AS "NÚMERO DA CONTA"
FROM CONTA,
     CLIENTE_PESSOA_FISICA
WHERE CONTA.ID_CLIENTE = CLIENTE_PESSOA_FISICA.ID
ORDER BY CLIENTE_PESSOA_FISICA.PRIMEIRO_NOME

```

11.10. LISTA CLIENTES PJ EM ORDEM ALFABÉTICA

```

SELECT CLIENTE_PESSOA_JURIDICA.NOME_FANTASIA AS "NOME COMPLETO",
CONTA.NUMERO AS "NÚMERO DA CONTA"
FROM CONTA,

```

```

    CLIENTE_PESSOA_JURIDICA
WHERE  CONTA.ID_CLIENTE = CLIENTE_PESSOA_JURIDICA.ID_CLIENTE
ORDER BY CLIENTE_PESSOA_JURIDICA.NOME_FANTASIA

```

11.11. LISTA DE CLIENTES PF COM CONTA POUPANÇA EM ORDEM ALFABÉTICA

```

SELECT CLIENTE_PESSOA_FISICA.PRIMEIRO_NOME || ' ' ||
CLIENTE_PESSOA_FISICA.SEGUNDO_NOME || ' ' ||
CLIENTE_PESSOA_FISICA.ULTIMO_NOME AS "NOME COMPLETO",
CONTA.NUMERO AS "NÚMERO DA CONTA",
TIPO_CONTA.NOME AS "TIPO DE CONTA"

FROM CONTA,
     AGENCIA_BANCARIA,
     CLIENTE_PESSOA_FISICA,
     TIPO_CONTA
WHERE CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID
      AND CONTA.ID_CLIENTE = CLIENTE_PESSOA_FISICA.ID
      AND CONTA.ID_TIPO_CONTA = TIPO_CONTA.ID
      AND TIPO_CONTA.ID = 2
ORDER BY CLIENTE_PESSOA_FISICA.PRIMEIRO_NOME

```

11.12. LISTA DE CLIENTES PJ COM CONTA POUPANÇA

```

SELECT CLIENTE_PESSOA_JURIDICA.NOME_FANTASIA AS "NOME FANTASIA",
CONTA.NUMERO AS "NÚMERO DA CONTA",
TIPO_CONTA.NOME AS "TIPO DE CONTA"

FROM CONTA,
     AGENCIA_BANCARIA,
     CLIENTE_PESSOA_JURIDICA,
     TIPO_CONTA
WHERE CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID
      AND CONTA.ID_CLIENTE = CLIENTE_PESSOA_JURIDICA.ID_CLIENTE
      AND CONTA.ID_TIPO_CONTA = TIPO_CONTA.ID
      AND TIPO_CONTA.ID = 2
ORDER BY CLIENTE_PESSOA_JURIDICA.NOME_FANTASIA

```

11.13. LISTA DE CLIENTES PF COM CONTA CORRENTE EM ORDEM ALFABÉTICA

```

SELECT CLIENTE_PESSOA_FISICA.PRIMEIRO_NOME || ' ' ||
CLIENTE_PESSOA_FISICA.SEGUNDO_NOME || ' ' ||
CLIENTE_PESSOA_FISICA.ULTIMO_NOME AS "NOME COMPLETO",
CONTA.NUMERO AS "NÚMERO DA CONTA",
TIPO_CONTA.NOME AS "TIPO DE CONTA"

```

```

FROM CONTA,
     AGENCIA_BANCARIA,
     CLIENTE_PESSOA_FISICA,
     TIPO_CONTA
WHERE CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID
      AND CONTA.ID_CLIENTE = CLIENTE_PESSOA_FISICA.ID
      AND CONTA.ID_TIPO_CONTA = TIPO_CONTA.ID
      AND TIPO_CONTA.ID = 1
ORDER BY CLIENTE_PESSOA_FISICA.PRIMEIRO_NOME

```

11.14. LISTA DE CLIENTES PJ COM CONTA CORRENTE

```

SELECT CLIENTE_PESSOA_JURIDICA.NOME_FANTASIA AS "NOME FANTASIA",
       CONTA.NUMERO AS "NÚMERO DA CONTA",
       TIPO_CONTA.NOME AS "TIPO DE CONTA"

```

```

FROM CONTA,
     AGENCIA_BANCARIA,
     CLIENTE_PESSOA_JURIDICA,
     TIPO_CONTA
WHERE CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID
      AND CONTA.ID_CLIENTE = CLIENTE_PESSOA_JURIDICA.ID_CLIENTE
      AND CONTA.ID_TIPO_CONTA = TIPO_CONTA.ID
      AND TIPO_CONTA.ID = 1
ORDER BY CLIENTE_PESSOA_JURIDICA.NOME_FANTASIA

```

11.15. TOTAL DE CONTAS POR AGÊNCIA

```

SELECT AGENCIA_BANCARIA.NOME AS "AGÊNCIA BANCÁRIA",
       COUNT(CONTA.ID) AS "TOTAL DE CONTAS ABERTAS"
FROM CONTA,
     AGENCIA_BANCARIA
WHERE AGENCIA_BANCARIA.ID = CONTA.ID_AGENCIA_BANCARIA
GROUP BY AGENCIA_BANCARIA.NOME

```

11.16. TOTAL DE CONTAS POUPANÇA POR AGÊNCIA

```

SELECT AGENCIA_BANCARIA.NOME AS "AGÊNCIA BANCÁRIA",
       COUNT(CONTA.ID_TIPO_CONTA) AS "TOTAL CONTAS ABERTAS"
FROM CONTA,
     AGENCIA_BANCARIA
WHERE CONTA.ID_TIPO_CONTA=2
      AND AGENCIA_BANCARIA.ID = CONTA.ID_AGENCIA_BANCARIA
GROUP BY AGENCIA_BANCARIA.NOME

```

11.17. TOTAL DE CONTAS CORRENTE POR AGÊNCIA

```
SELECT AGENCIA_BANCARIA.NOME AS "AGÊNCIA BANCÁRIA",  
       COUNT(CONTA.ID_TIPO_CONTA) AS "TOTAL CONTAS ABERTAS"  
FROM CONTA,  
     AGENCIA_BANCARIA  
WHERE CONTA.ID_TIPO_CONTA=1  
      AND AGENCIA_BANCARIA.ID = CONTA.ID_AGENCIA_BANCARIA  
GROUP BY AGENCIA_BANCARIA.NOME
```

11.18. TOTAL DE CONTAS PF ABERTAS NO MÊS POR AGÊNCIA

```
SELECT CONTA.NUMERO AS "NÚMERO DA CONTA",  
       TIPO_CONTA.NOME AS "TIPO DE CONTA",  
       CLIENTE_PESSOA_FISICA.PRIMEIRO_NOME || ' ' ||  
CLIENTE_PESSOA_FISICA.SEGUNDO_NOME || ' ' ||  
CLIENTE_PESSOA_FISICA.ULTIMO_NOME AS "NOME COMPLETO",  
       AGENCIA_BANCARIA.NOME AS "AGÊNCIA BANCÁRIA"  
FROM CONTA,  
     CLIENTE_PESSOA_FISICA,  
     AGENCIA_BANCARIA,  
     TIPO_CONTA  
WHERE CONTA.DATA_ABERTURA > '01-12-2021'  
      AND CONTA.DATA_ABERTURA < '31-12-2021'  
      AND CONTA.ID_CLIENTE = CLIENTE_PESSOA_FISICA.ID_CLIENTE  
      AND CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID  
      AND CONTA.ID_TIPO_CONTA = TIPO_CONTA.ID
```

11.19. MOVIMENTAÇÃO DAS AGÊNCIAS

```
SELECT  
     AGENCIA_BANCARIA.NOME AS "AGÊNCIA BANCÁRIA",  
     SUM(TRANSACAO.VALOR) AS "DINHEIRO MOVIMENTADO (TOTAL)"  
FROM TRANSACAO,  
     AGENCIA_BANCARIA,  
     CONTA  
WHERE TRANSACAO.ID_CONTA_ORIGEM = CONTA.ID  
      AND CONTA.ID_AGENCIA_BANCARIA = AGENCIA_BANCARIA.ID  
GROUP BY AGENCIA_BANCARIA.NOME
```

11.20. CONTAS PJ ABERTAS NO ÚLTIMO MÊS, POR AGÊNCIA

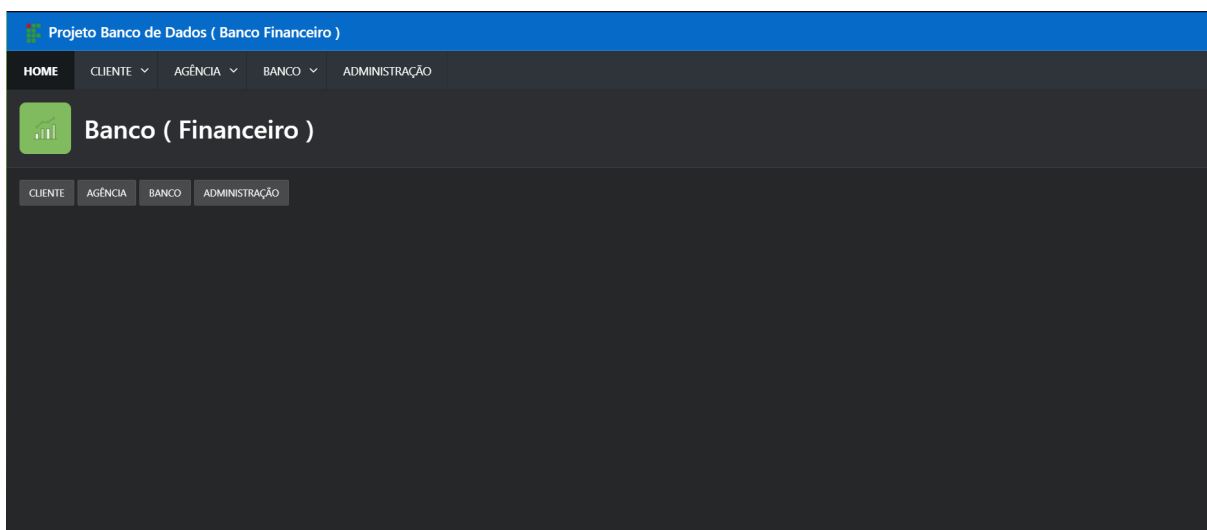
```
SELECT CONTA.NUMERO AS "NÚMERO DA CONTA",  
       TIPO_CONTA.NOME AS "TIPO DE CONTA",
```

```

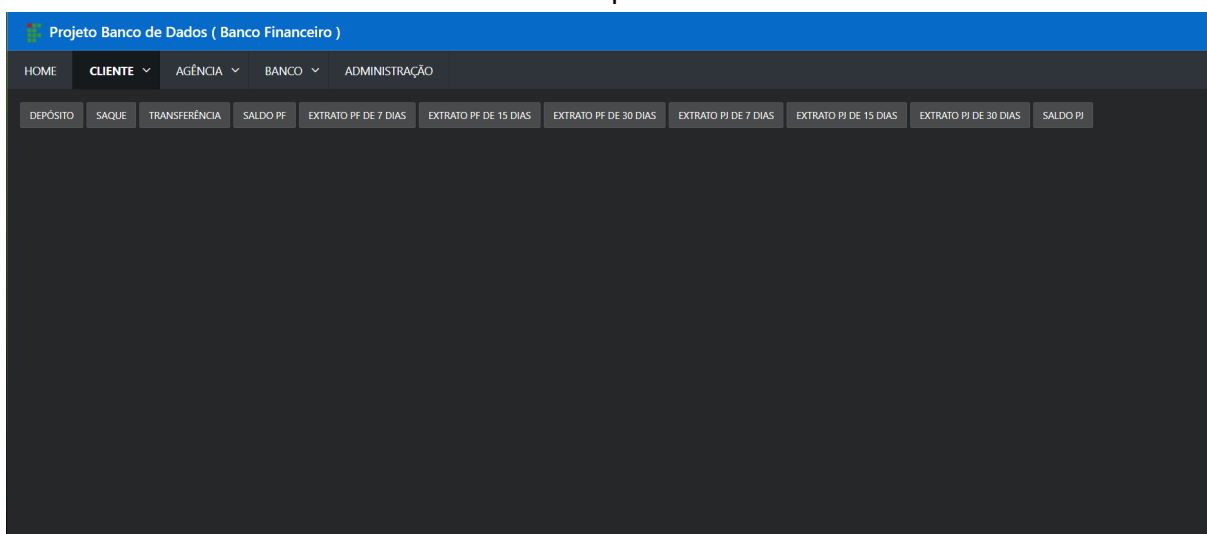
    CLIENTE_PESSOA_JURIDICA.NOME_FANTASIA AS "NOME
    FANTASIA",
    AGENCIA_BANCARIA.NOME AS "AGÊNCIA BANCÁRIA"
FROM CONTA,
    CLIENTE_PESSOA_JURIDICA,
    AGENCIA_BANCARIA,
    TIPO_CONTA
WHERE CONTA.DATA_ABERTURA > '01-12-2021'
    AND CONTA.DATA_ABERTURA < '31-12-2021'
    AND CONTA.ID_CLIENTE =
    CLIENTE_PESSOA_JURIDICA.ID_CLIENTE
    AND CONTA.ID_AGENCIA_BANCARIA =
    AGENCIA_BANCARIA.ID
    AND CONTA.ID_TIPO_CONTA = TIPO_CONTA.ID

```

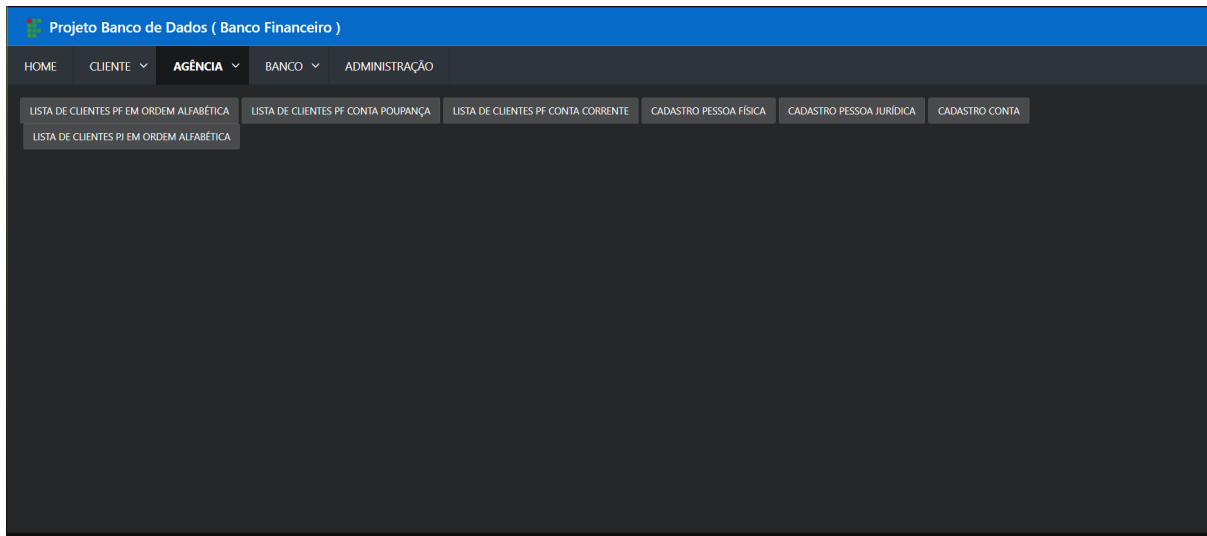
12. APLICAÇÃO



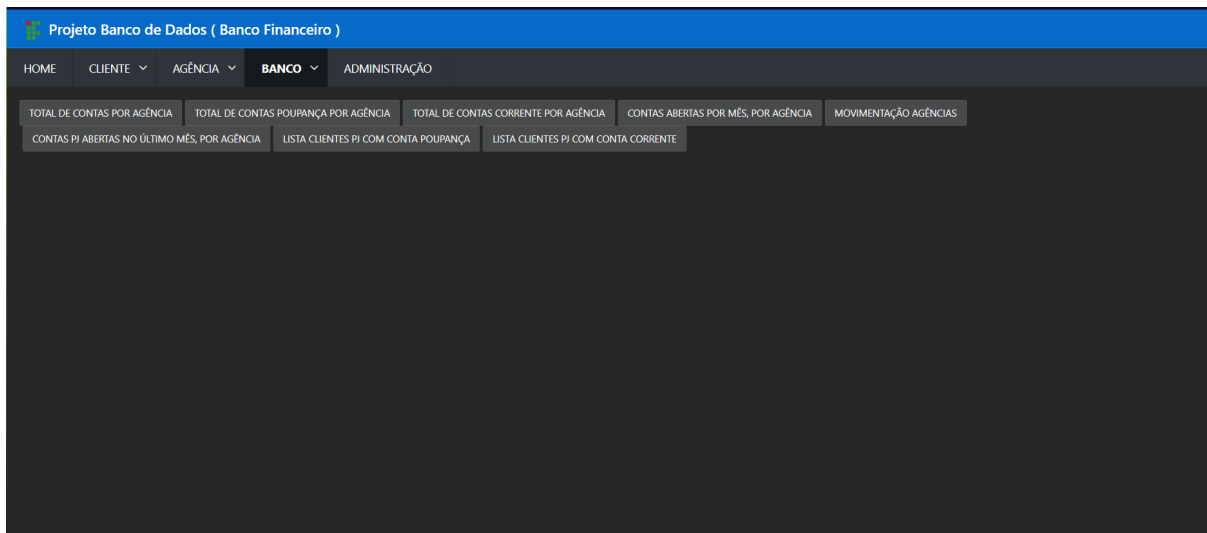
Fonte: Próprio Autor




Fonte: Próprio Autor



Fonte: Próprio Autor





Fonte: Próprio Autor




DEPÓSITO

DATA DA TRANSACAO







NUM_TRANSACAO




VALOR



ID_TERMINAL



ID_CLIENTE



ID_CONTA

DEPOSITAR

Cancelar

Fonte: Próprio Autor



Saque

Data da Transação



Num_Transação



Valor



ID_Terminal



ID_Cliente



ID_Conta Origem

SACAR

Cancelar

Fonte: Próprio Autor



TRANSFERÊNCIA



Num Transação



Data da Transação



Valor



ID Terminal



ID Cliente



ID Conta Origem



ID Conta Destino

Transferir

Cancelar

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)				
HOME	CLIENTE ▾	AGÊNCIA ▾	BANCO ▾	ADMINISTRAÇÃO
Saldo				
Q ▾		Ir	Ações ▾	
Saldo	Número Da Conta	Número Da Agência	Nome Completo	
5550,89	558723	611	Leonardo Pavan cunha	
175,54	757485	3502	Vinicius Souza Santos	
500,5	7894521	3502	JOVANDERSON MATIAS SOUZA	
10000	558969	574	Emily Silva Costa	
				1 - 4

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)

HOMECLIENTE ▾AGÊNCIA ▾BANCO ▾ADMINISTRAÇÃO

Extrato de 30 Dias

Q ▾

Ir

Ações ▾

Conta	Data	Valor	Saldo	Agência Bancária	Numero Transação	Terminal	Tipo De Transação	Conta De Origem	Conta De Destino	Nome Completo
757485	7/12/2021	50	175,54	3502	487878544	8524	TRANSFERÊNCIA	13	14	Vinicius Souza Santos
558723	13/12/2021	50	5550,89	611	15986	4564	TRANSFERÊNCIA	14	13	Leonardo Pavan cunha
558723	14/12/2021	250	5550,89	611	14896	18945	TRANSFERÊNCIA	14	13	Leonardo Pavan cunha
757485	7/12/2021	50	175,54	3502	487878541	8524	TRANSFERÊNCIA	13	14	Vinicius Souza Santos
757485	2/12/2021	50	175,54	3502	487878555	8524	TRANSFERÊNCIA	13	14	Vinicius Souza Santos
757485	9/12/2021	50	175,54	3502	48787897	8524	TRANSFERÊNCIA	12	13	Vinicius Souza Santos
757485	12/12/2021	200	175,54	3502	564156456	6895	DEPÓSITO	12		Vinicius Souza Santos
558723	12/12/2021	500	5550,89	611	189576	1665	DEPÓSITO	14		Leonardo Pavan cunha

1 - 8

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)

HOME

CLIENTE ▾

AGÊNCIA ▾

BANCO ▾

ADMINISTRAÇÃO

EXTRATO PJ 30 DIAS

Q ▾

Ir

Ações ▾

Conta	Agência Bancária	Nome Fantasia	Numero Transação	Data	Valor	Terminal	Tipo De Transação	Conta De Origem	Conta De Destino	Saldo
7878541	3502	Vinicius S/A	9877845	13/12/2021	950	55871	DEPÓSITO	23		7950
789652	2112	Emily Shows LTDA	98789456	13/12/2021	890	54512	DEPÓSITO	21		5890

1 - 2

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)	
HOME	<div> <div>CLIENTE</div> <div>AGÊNCIA</div> <div>BANCO</div> <div>ADMINISTRAÇÃO</div> </div>
Lista Clientes Ordem Alfabetica	
<div> <div>Q</div> <div></div> <div>Ir</div> <div>Ações</div> </div>	
Nome Completo	Número Da Conta
Emily Silva Costa	558969
JOVIANDERSON MATIAS SOUZA	7894521
Leonardo Pavan cunha	558723
Vinicius Souza Santos	757485
1 - 4	

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)

HOME

CLIENTE

AGÊNCIA

BANCO

ADMINISTRAÇÃO

Lista Clientes Conta Poupança

Q

Ir

Ações

Tipo De Conta	Nome Completo	Número Da Conta
POUPANÇA	Emily Silva Costa	558969

1 - 1

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)

HOME

CLIENTE ▾

AGÊNCIA ▾

BANCO ▾

ADMINISTRAÇÃO

Lista Clientes Conta Corrente

Q ▾


Ir

Ações ▾

Tipo De Conta	Nome Completo	Número Da Conta
CORRENTE	JOVIANDERSON MATIAS SOUZA	7894521
CORRENTE	Leonardo Pavan cunha	558723
CORRENTE	Vinicius Souza Santos	757485

1 - 3

Fonte: Próprio Autor



CADASTRO PESSOA FÍSICA

①

Primeiro Nome

②

Nome do Meio

③

Último Nome

E-mail

Número do endereço

ID Endereço

RG

CPF

Data de Nascimento

Gênero

Cadastrar

Cancelar

Fonte: Próprio Autor



CADASTRO PESSOA JURÍDICA



E-mail



Número do Endereço



ID Endereço



Nome Fantasia



Razão Social




CNPJ


Cadastrar


Cancelar


Fonte: Próprio Autor





CADASTRO DE CONTA
































Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)

HOME

CLIENTE

AGÊNCIA

BANCO

ADMINISTRAÇÃO

LISTA CLIENTES PJ EM ORDEM ALFABÉTICA

Q

Ir

Ações

Nome Completo	Número Da Conta
Emily Shows LTDA	789652
Vinicius S/A	7878541

1 - 2

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)

HOME

CLIENTE

AGÊNCIA

BANCO

ADMINISTRAÇÃO

Total de contas por agência

Q

Ir

Ações

Agência Bancária	Total De Contas Abertas
AGÊNCIA 2112 - BRADESCO	1
AGÊNCIA 4557 - SANTANDER	1
Agência 0611 - ITAÚ UNIBANCO S.A	1
Agência Birigui da CEF	1
Agência Cidade Perola	3

1 - 5

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)

HOME

CLIENTE

AGÊNCIA

BANCO

ADMINISTRAÇÃO

Total de contas poupança por agência

Q

Ir

Ações

Agência Bancária	Total Contas Abertas
Agência Birigui da CEF	1
Agência Cidade Perola	1

1 - 2

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)

HOME

CLIENTE

AGÊNCIA

BANCO

ADMINISTRAÇÃO

Total de contas corrente por agência

Q

Ir

Ações

Agência Bancária	Total Contas Abertas
AGÊNCIA 2112 - BRADESCO	1
AGÊNCIA 4557 - SANTANDER	1
Agência 0611 - ITAÚ UNIBANCO S.A	1
Agência Cidade Perola	2
1 - 4	

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)

HOME

CLIENTE

AGÊNCIA

BANCO

ADMINISTRAÇÃO

Contas abertas por mês, por agência

Q

Ir

Ações

Agência Bancária	Número Da Conta	Nome Completo	Tipo De Conta
Agência Cidade Perola	757485	Vinicius Souza Santos	CORRENTE
AGÊNCIA 4557 - SANTANDER	45892	Julia Alves Santos	CORRENTE
1 - 2			

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)

HOME

CLIENTE

AGÊNCIA

BANCO

ADMINISTRAÇÃO

Movimentação Agências

Q

Ir

Ações

Agência Bancária	Dinheiro Movimentado (total)
AGÊNCIA 2112 - BRADESCO	890
Agência 0611 - ITAÚ UNIBANCO S.A	800
Agência Birigui da CEF	150
Agência Cidade Perola	1200
1 - 4	

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)			
HOME	CLIENTE ▾	AGÊNCIA ▾	BANCO ▾
ADMINISTRAÇÃO			
CONTAS PESSOA JURÍDICA ABERTAS NO ÚLTIMO MÊS, POR AGÊNCIA			
Q ▾	Ir	Ações ▾	
Número Da Conta	Tipo De Conta	Nome Fantasia	Agência Bancária
7878541	POUPANÇA	Vinicius S/A	Agência Cidade Perola
789652	CORRENTE	Emily Shows LTDA	AGÊNCIA 2112 - BRADESCO
			1 - 2

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)			
HOME	CLIENTE ▾	AGÊNCIA ▾	BANCO ▾
ADMINISTRAÇÃO			
LISTA CLIENTES PJ CONTA POUPANÇA			
Q ▾	Ir	Ações ▾	
Nome Fantasia	Número Da Conta	Tipo De Conta	
Vinicius S/A	7878541	POUPANÇA	
			1 - 1

Fonte: Próprio Autor

Projeto Banco de Dados (Banco Financeiro)			
HOME	CLIENTE ▾	AGÊNCIA ▾	BANCO ▾
ADMINISTRAÇÃO			
LISTA CLIENTES PJ CONTA CORRENTE			
Q ▾	Ir	Ações ▾	
Nome Fantasia	Número Da Conta	Tipo De Conta	
Emily Shows LTDA	789652	CORRENTE	
			1 - 1

Fonte: Próprio Autor