**FACULDADE DE INFORMÁTICA E ADMINISTRAÇÃO PAULISTA**

CARLOS ALBERTO MACHARELLI JUNIOR - RM 551677

CARLOS EDUARDO MENDONÇA DA SILVA – RM 552164

EDUARDO TOSHIO ROCHA OKUBO – RM 551763

KAUÊ ALEXANDRE DE OLIVEIRA – RM 551812

VITOR MACHADO MIRANDA – RM 551451

**DOCUMENTAÇÃO DO PROJETO BEAUTY TECH**

CARLOS ALBERTO MACHARELLI JUNIOR - RM 551677

CARLOS EDUARDO MENDONÇA DA SILVA – RM 552164

EDUARDO TOSHIO ROCHA OKUBO – RM 551763

KAUÊ ALEXANDRE DE OLIVEIRA – RM 551812

VITOR MACHADO MIRANDA – RM 551451

**DOCUMENTAÇÃO DO PROJETO BEAUTY TECH**

Trabalho acadêmico apresentado à disciplina de Mastering Relational and Non-Relational Database do Curso Análise e Desenvolvimento de Sistemas da Faculdade de Informática e Administração Paulista como requisito de nota da Sprint 3 da Turma 2TDSPV. Requerido pelo prof. Renê de Ávila Mendes.

**SUMÁRIO**

**1 INTRODUÇÃO .....................................................................................................................4**

**2 Projeto de banco de dados modelo relacional Beauty Tech.................................................5**

**3 Projeto de banco de dados modelo físico Beauty Tech........................................................6**

**4 Criamos procedures de INSERT/ UPDATE / DELETE**

**4.1 CRUD TABELA CATEGORIA**

**4.1.1 INSERT BT\_CATEGORIA………………………………………………………………………8**

**4.1.2 UPDATE BT\_CATEGORIA…………………………………………………………………….8**

**4.1.3 DELETE BT\_CATEGORIA……………………………………………………………………..8**

**4.2 CRUD TABELA CLIENTE**

**4.2.1 INSERT BT\_CLIENTE………………………………………………………………………………9**

**4.2.2 UPDATE BT\_CLIENTE…………………………………………………………………………..10**

**4.2.3 DELETE BT\_CLIENTE……………………………………………………………………………10**

**4.3 CRUD TABELA EMPRESA**

**4.3.1 INSERT BT\_EMPRESA…………………………………………………………………………11**

**4.3.2 UPDATE BT\_EMPRESA……………………………………………………………………….11**

**4.3.3 DELETE BT\_EMPRESA…………………………………………………………………………12**

**4.4 CRUD TABELA ENDERECO**

**4.4.1 INSERT BT\_ENDERECO………………………………………………………………………12**

**4.4.2 UPDATE BT\_ENDERECO………………………………………………………………………13**

**4.4.2 DELETE BT\_ENDERECO……………………………………………………………………….14**

**4.5 CRUD TABELA PRODUTO**

**4.5.1 INSERT BT\_PRODUTO………………………………………………………………………..14**

**4.5.2 UPDATE BT\_PRODUTO………………………………………………………………………15**

**4.5.3 DELETE BT\_PRODUTO ………………………………………………………………………15**

**4.6 CRUD BT\_TELEFONE**

**4.6.1 INSERT BT\_PRODUTO…………………………………………………………………….…16**

**4.6.2 UPDATE BT\_PRODUTO………………………………………………………………………16**

**4.6.3 DELETE BT\_PRODUTO……………………………………………………………….………17**

**5 PROCEDURE GET\_CLIENTES COM TELEFONE…………………………………………………………………18**

**6 PROCEDURE CR\_RELATORIO\_ACESSOS\_CLIENTE……………………………………………………………18**

**7 ARQUIVO PYTHON PARA CHAMADA DAS PROCEDURES…………………………………………………19**

**8 FUNÇÃO VALIDAR CPF……………………………………………………………………………………………………21**

**8.1 TESTE CORRETO FUNÇÃO VALIDAR CPF**

**8.2 TESTE COM ERROS FUNÇÃO VALIDAR E-MAIL**

**9 FUNÇÃO VALIDAR EMAIL……………………………………………………………………………………………….23**

**9.1 TESTE CORRETO FUNÇÃO VALIDAR EMAIL**

**9.2 TESTE COM ERROS FUNÇÃO VALIDAR EMAIL**

**10 FUNÇÃO VERIFICAR MAIORIDADE………………………………………………………………………………24**

**10.1 SELECT TESTANDO FUNÇÃO VERIFICAR MAIORIDADE**

**11 FUNÇÃO TRANSFORMANDO EM JSON…………………………………………………………………………25**

**12 PROCEDURE GERAR JSON CLIENTE………………………………………………………………………………26**

**12.1 TESTE CORRETO PROCEDURE GERAR JSON CLIENTE**

**12.2 TESTE COM ERROS PROCEDURE GERAR JSON CLIENTE**

**13 PROCEDURE IMPRIMINDO VALORES……………………………………………………………………………27**

**13.1 TESTE PROCEDURE IMPRIMINDO VALORES**

**14 CRIAÇÃO TABELA AUDITORIA……………………………………………………………………………………….27**

**15 TRIGGER AUDITORIA BT CLIENTE………………………………………………………………………………… 28**

**15.1 TESTE TRIGGER COM INSERT**

**15.2 TESTE TRIGGER COM UPDATE**

**15.3 TESTE TRIGGER COM DELETE**

**1 INTRODUÇÃO**

Nosso projeto, Beauty Tech, surge para abordar os desafios enfrentados no setor de cosméticos, onde a abundância de opções pode sobrecarregar os consumidores. O primeiro desafio é o desengajamento dos clientes na busca por produtos específicos devido à vasta oferta disponível. Em seguida, a gestão de estoque se torna um desafio, com excesso de produtos menos populares e escassez dos mais procurados. Por último, a intensa competição no mercado exige estratégias inovadoras para se destacar.

Nossa solução será um sistema de consulta e busca, que auxiliará o público-alvo na busca de produtos cosméticos que melhor se identifiquem com as categorias desejadas por ele (foco, cor, tipo de pele, clima, etc.), listando os produtos desejados de forma precisa e rápida. Essas buscas abrangerão clientes híbridos e digitais, já que o sistema estará disponível tanto no site quanto no aplicativo para vendas online, além de estar integrado em totens nas lojas físicas. Isso ajudará as lojas físicas a oferecerem assistência e venda dos produtos de forma mais eficiente, mostrando quantidade, especificações e detalhes, o que pode reduzir a necessidade de funcionários para assistência e aumentar a precisão das buscas para os clientes.

**2 Projeto de banco de dados modelo relacional Beauty Tech:**

**Diagrama

Descrição gerada automaticamente**

**3 Projeto de banco de dados modelo físico Beauty Tech**

**Diagrama

Descrição gerada automaticamente**

**4 Criamos procedures de INSERT/ UPDATE / DELETE:**

**4.1** **CRUD TABELA BT\_CATEGORIA**

**4.1.1 INSERT BT\_CATEGORIA**  
**CREATE** OR REPLACE PROCEDURE insert\_bt\_categoria (  
    p\_id\_categoria   IN bt\_categoria.id\_categoria%TYPE,  
    p\_nm\_categoria   IN bt\_categoria.nm\_categoria%TYPE,  
    p\_desc\_categoria IN bt\_categoria.desc\_categoria%TYPE  
) IS  
**BEGIN**  
    **INSERT** **INTO** bt\_categoria (id\_categoria, nm\_categoria, desc\_categoria)  
    **VALUES** (p\_id\_categoria, p\_nm\_categoria, p\_desc\_categoria);  
EXCEPTION  
    WHEN DUP\_VAL\_ON\_INDEX THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: Não podemos ter duplicidade de ID na tabela, faça o insert com outro ID.');  
COMMIT;  
END;  
/  
 **4.1.2 UPDATE BT\_CATEGORIA**  
**CREATE** OR REPLACE PROCEDURE update\_bt\_categoria (  
    p\_id\_categoria   IN bt\_categoria.id\_categoria%TYPE,  
    p\_nm\_categoria   IN bt\_categoria.nm\_categoria%TYPE,  
    p\_desc\_categoria IN bt\_categoria.desc\_categoria%TYPE  
) IS  
**BEGIN**  
    **UPDATE** bt\_categoria  
    **SET** nm\_categoria = p\_nm\_categoria,  
        desc\_categoria = p\_desc\_categoria  
    **WHERE** id\_categoria = p\_id\_categoria;  
  
    -- Verifica se alguma linha foi atualizada   
    **IF** SQL%ROWCOUNT = 0 THEN  
        DBMS\_OUTPUT.PUT\_LINE( 'Erro: ID da Categoria a ser atualizada não encontrada.');  
    END **IF**;  
COMMIT;  
END;  
  
/  
 **4.1.3 DELETE BT\_CATEGORIA**  
**CREATE** OR REPLACE PROCEDURE delete\_bt\_categoria (  
    p\_id\_categoria IN bt\_categoria.id\_categoria%TYPE  
) IS  
**BEGIN**  
    **DELETE** **FROM** bt\_categoria  
    **WHERE** id\_categoria = p\_id\_categoria;  
  
    **IF** SQL%NOTFOUND THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: Categoria a ser excluída não encontrada.');  
    END **IF**;  
COMMIT;  
END;  
/  
  
  
  
**4.2 CRUD TABELA CLIENTE**

**4.2.1 INSERT BT\_CLIENTE**  
**CREATE** OR REPLACE PROCEDURE insert\_bt\_cliente (  
    p\_id\_cliente             IN bt\_cliente.id\_cliente%TYPE,  
    p\_id\_telefone            IN bt\_cliente.id\_telefone%TYPE,  
    p\_id\_genero              IN bt\_cliente.id\_genero%TYPE,  
    p\_cpf\_cliente            IN bt\_cliente.cpf\_cliente%TYPE,  
    p\_nm\_cliente             IN bt\_cliente.nm\_cliente%TYPE,  
    p\_email\_cliente          IN bt\_cliente.email\_cliente%TYPE,  
    p\_dt\_nascimento\_cliente  IN bt\_cliente.dt\_nascimento\_cliente%TYPE,  
    p\_estado\_civil\_cliente   IN bt\_cliente.estado\_civil\_cliente%TYPE,  
    p\_dt\_cadastro            IN bt\_cliente.dt\_cadastro%TYPE,  
    p\_dt\_exclusao            IN bt\_cliente.dt\_exclusao%TYPE DEFAULT NULL  
) IS  
**BEGIN**  
    **INSERT** **INTO** bt\_cliente (  
        id\_cliente,  
        id\_telefone,  
        id\_genero,  
        cpf\_cliente,  
        nm\_cliente,  
        email\_cliente,  
        dt\_nascimento\_cliente,  
        estado\_civil\_cliente,  
        dt\_cadastro,  
        dt\_exclusao  
    ) **VALUES** (  
        p\_id\_cliente,  
        p\_id\_telefone,  
        p\_id\_genero,  
        p\_cpf\_cliente,  
        p\_nm\_cliente,  
        p\_email\_cliente,  
        p\_dt\_nascimento\_cliente,  
        p\_estado\_civil\_cliente,  
        p\_dt\_cadastro,  
        p\_dt\_exclusao  
    );  
    COMMIT;  
EXCEPTION  
    WHEN DUP\_VAL\_ON\_INDEX THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: Não podemos ter duplicidade de ID na tabela, faça o insert com outro ID.');  
COMMIT;  
  
END;  
/  
 **4.2.2 UPDATE BT\_CLIENTE**  
**CREATE** OR REPLACE PROCEDURE update\_bt\_cliente (  
    p\_id\_cliente            IN bt\_cliente.id\_cliente%TYPE,  
    p\_id\_telefone           IN bt\_cliente.id\_telefone%TYPE,  
    p\_id\_genero             IN bt\_cliente.id\_genero%TYPE,  
    p\_cpf\_cliente           IN bt\_cliente.cpf\_cliente%TYPE,  
    p\_nm\_cliente            IN bt\_cliente.nm\_cliente%TYPE,  
    p\_email\_cliente         IN bt\_cliente.email\_cliente%TYPE,  
    p\_dt\_nascimento\_cliente IN bt\_cliente.dt\_nascimento\_cliente%TYPE,  
    p\_estado\_civil\_cliente  IN bt\_cliente.estado\_civil\_cliente%TYPE,  
    p\_dt\_exclusao           IN bt\_cliente.dt\_exclusao%TYPE  
) IS  
**BEGIN**  
    **UPDATE** bt\_cliente  
    **SET**  
        id\_telefone = p\_id\_telefone,  
        id\_genero = p\_id\_genero,  
        cpf\_cliente = p\_cpf\_cliente,  
        nm\_cliente = p\_nm\_cliente,  
        email\_cliente = p\_email\_cliente,  
        dt\_nascimento\_cliente = p\_dt\_nascimento\_cliente,  
        estado\_civil\_cliente = p\_estado\_civil\_cliente,  
        dt\_exclusao = p\_dt\_exclusao  
    **WHERE**  
        id\_cliente = p\_id\_cliente;  
  
    **IF** SQL%NOTFOUND THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: ID do Cliente a ser atualizado não encontrado.');  
    END **IF**;  
    COMMIT;  
  
END;  
/  
 **4.2.3 DELETE BT\_CLIENTE**  
**CREATE** OR REPLACE PROCEDURE delete\_bt\_cliente (  
    p\_id\_cliente IN bt\_cliente.id\_cliente%TYPE  
) IS  
**BEGIN**  
    **DELETE** **FROM** bt\_cliente  
    **WHERE** id\_cliente = p\_id\_cliente;  
  
    **IF** SQL%NOTFOUND THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: Cliente a ser excluído não encontrado.');  
    END **IF**;  
    COMMIT;  
  
END;  
/  
  
 **4.3 CRUD TABELA BT\_EMPRESA**

**4.3.1 INSERT BT\_EMPRESA**  
**CREATE** OR REPLACE PROCEDURE insert\_bt\_empresa (  
    p\_id\_empresa    IN bt\_empresa.id\_empresa%TYPE,  
    p\_nm\_empresa    IN bt\_empresa.nm\_empresa%TYPE,  
    p\_cnpj\_empresa  IN bt\_empresa.cnpj\_empresa%TYPE,  
    p\_desc\_empresa  IN bt\_empresa.desc\_empresa%TYPE  
) IS  
**BEGIN**  
    **INSERT** **INTO** bt\_empresa (  
        id\_empresa,  
        nm\_empresa,  
        cnpj\_empresa,  
        desc\_empresa  
    ) **VALUES** (  
        p\_id\_empresa,  
        p\_nm\_empresa,  
        p\_cnpj\_empresa,  
        p\_desc\_empresa  
    );  
    COMMIT;  
EXCEPTION  
    WHEN DUP\_VAL\_ON\_INDEX THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: Não podemos ter duplicidade de ID na tabela, faça o insert com outro ID.');  
COMMIT;  
END;  
/  
 **4.3.2 UPDATE  BT\_EMPRESA**  
**CREATE** OR REPLACE PROCEDURE update\_bt\_empresa (  
    p\_id\_empresa    IN bt\_empresa.id\_empresa%TYPE,  
    p\_nm\_empresa    IN bt\_empresa.nm\_empresa%TYPE,  
    p\_cnpj\_empresa  IN bt\_empresa.cnpj\_empresa%TYPE,  
    p\_desc\_empresa  IN bt\_empresa.desc\_empresa%TYPE  
) IS  
**BEGIN**  
    **UPDATE** bt\_empresa  
    **SET**  
        nm\_empresa = p\_nm\_empresa,  
        cnpj\_empresa = p\_cnpj\_empresa,  
        desc\_empresa = p\_desc\_empresa  
    **WHERE**  
        id\_empresa = p\_id\_empresa;  
  
    **IF** SQL%NOTFOUND THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: ID da Empresa a ser atualizada não encontrado.');  
    END **IF**;  
    COMMIT;  
END;  
/  
 **4.3.3 DELETE  BT\_EMPRESA**  
**CREATE** OR REPLACE PROCEDURE delete\_bt\_empresa (  
    p\_id\_empresa IN bt\_empresa.id\_empresa%TYPE  
) IS  
**BEGIN**  
    **DELETE** **FROM** bt\_empresa  
    **WHERE** id\_empresa = p\_id\_empresa;  
  
    **IF** SQL%NOTFOUND THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: Empresa a ser excluída não encontrada.');  
    END **IF**;  
    COMMIT;  
END;  
/  
 **4.4 CRUD BT\_ENDERECO**

**4.4.1 INSERT BT\_ENDERECO**  
**CREATE** OR REPLACE PROCEDURE insert\_bt\_endereco (  
    p\_id\_endereco          IN bt\_endereco.id\_endereco%TYPE,  
    p\_id\_empresa           IN bt\_endereco.id\_empresa%TYPE,  
    p\_uf\_endereco          IN bt\_endereco.uf\_endereco%TYPE,  
    p\_cidade\_endereco      IN bt\_endereco.cidade\_endereco%TYPE,  
    p\_bairro\_endereco      IN bt\_endereco.bairro\_endereco%TYPE,  
    p\_logradouro\_endereco  IN bt\_endereco.logradouro\_endereco%TYPE,  
    p\_nr\_endereco          IN bt\_endereco.nr\_endereco%TYPE,  
    p\_complemento\_endereco IN bt\_endereco.complemento\_endereco%TYPE DEFAULT NULL  
) IS  
**BEGIN**  
    **INSERT** **INTO** bt\_endereco (  
        id\_endereco,  
        id\_empresa,  
        uf\_endereco,  
        cidade\_endereco,  
        bairro\_endereco,  
        logradouro\_endereco,  
        nr\_endereco,  
        complemento\_endereco  
    ) **VALUES** (  
        p\_id\_endereco,  
        p\_id\_empresa,  
        p\_uf\_endereco,  
        p\_cidade\_endereco,  
        p\_bairro\_endereco,  
        p\_logradouro\_endereco,  
        p\_nr\_endereco,  
        p\_complemento\_endereco  
    );  
    COMMIT;  
EXCEPTION  
    WHEN DUP\_VAL\_ON\_INDEX THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: Não podemos ter duplicidade de ID na tabela, faça o insert com outro ID.');  
COMMIT;          
END;  
/  
 **4.4.2 ATUALIZAR BT\_ENDERECO**  
**CREATE** OR REPLACE PROCEDURE update\_bt\_endereco (  
    p\_id\_endereco          IN bt\_endereco.id\_endereco%TYPE,  
    p\_id\_empresa           IN bt\_endereco.id\_empresa%TYPE,  
    p\_uf\_endereco          IN bt\_endereco.uf\_endereco%TYPE,  
    p\_cidade\_endereco      IN bt\_endereco.cidade\_endereco%TYPE,  
    p\_bairro\_endereco      IN bt\_endereco.bairro\_endereco%TYPE,  
    p\_logradouro\_endereco  IN bt\_endereco.logradouro\_endereco%TYPE,  
    p\_nr\_endereco          IN bt\_endereco.nr\_endereco%TYPE,  
    p\_complemento\_endereco IN bt\_endereco.complemento\_endereco%TYPE  
) IS  
**BEGIN**  
    **UPDATE** bt\_endereco  
    **SET**  
        id\_empresa = p\_id\_empresa,  
        uf\_endereco = p\_uf\_endereco,  
        cidade\_endereco = p\_cidade\_endereco,  
        bairro\_endereco = p\_bairro\_endereco,  
        logradouro\_endereco = p\_logradouro\_endereco,  
        nr\_endereco = p\_nr\_endereco,  
        complemento\_endereco = p\_complemento\_endereco  
    **WHERE**  
        id\_endereco = p\_id\_endereco;  
  
    **IF** SQL%NOTFOUND THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: ID do Endereço a ser atualizado não encontrado.');  
    END **IF**;  
COMMIT;  
END;  
/  
 **4.4.3 DELETE  BT\_ENDERECO**  
**CREATE** OR REPLACE PROCEDURE delete\_bt\_endereco (  
    p\_id\_endereco IN bt\_endereco.id\_endereco%TYPE  
) IS  
**BEGIN**  
    **DELETE** **FROM** bt\_endereco  
    **WHERE** id\_endereco = p\_id\_endereco;  
  
    **IF** SQL%NOTFOUND THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: Endereço a ser excluído não encontrado.');  
    END **IF**;  
COMMIT;  
END;  
/  
 **4.5 CRUD BT\_PRODUTO**

**4.5.1 INSERT BT\_PRODUTO**  
**CREATE** OR REPLACE PROCEDURE insert\_bt\_produto (  
    p\_id\_produto     IN bt\_produto.id\_produto%TYPE,  
    p\_nm\_produto     IN bt\_produto.nm\_produto%TYPE,  
    p\_desc\_produto   IN bt\_produto.desc\_produto%TYPE,  
    p\_vl\_produto     IN bt\_produto.vl\_produto%TYPE,  
    p\_dt\_fab\_produto IN bt\_produto.dt\_fab\_produto%TYPE,  
    p\_dt\_validade    IN bt\_produto.dt\_validade%TYPE,  
    p\_dt\_cadastro    IN bt\_produto.dt\_cadastro%TYPE,  
    p\_img\_produto    IN bt\_produto.img\_produto%TYPE  
) IS  
**BEGIN**  
    **INSERT** **INTO** bt\_produto (  
        id\_produto,  
        nm\_produto,  
        desc\_produto,  
        vl\_produto,  
        dt\_fab\_produto,  
        dt\_validade,  
        dt\_cadastro,  
        img\_produto  
    ) **VALUES** (  
        p\_id\_produto,  
        p\_nm\_produto,  
        p\_desc\_produto,  
        p\_vl\_produto,  
        p\_dt\_fab\_produto,  
        p\_dt\_validade,  
        p\_dt\_cadastro,  
        p\_img\_produto  
    );  
EXCEPTION  
    WHEN DUP\_VAL\_ON\_INDEX THEN          
        DBMS\_OUTPUT.PUT\_LINE('Erro: Não podemos ter duplicidade de ID na tabela, faça o insert com outro ID.');  
  
COMMIT;     
END;  
  
/

**4.5.2 UPDATE BT\_PRODUTO**

**CREATE** OR REPLACE PROCEDURE update\_bt\_produto (  
    p\_id\_produto     IN bt\_produto.id\_produto%TYPE,  
    p\_nm\_produto     IN bt\_produto.nm\_produto%TYPE,  
    p\_desc\_produto   IN bt\_produto.desc\_produto%TYPE,  
    p\_vl\_produto     IN bt\_produto.vl\_produto%TYPE,  
    p\_dt\_fab\_produto IN bt\_produto.dt\_fab\_produto%TYPE,  
    p\_dt\_validade    IN bt\_produto.dt\_validade%TYPE,  
    p\_dt\_cadastro    IN bt\_produto.dt\_cadastro%TYPE,  
    p\_img\_produto    IN bt\_produto.img\_produto%TYPE  
) IS  
**BEGIN**  
    **UPDATE** bt\_produto  
    **SET**  
        nm\_produto = p\_nm\_produto,  
        desc\_produto = p\_desc\_produto,  
        vl\_produto = p\_vl\_produto,  
        dt\_fab\_produto = p\_dt\_fab\_produto,  
        dt\_validade = p\_dt\_validade,  
        dt\_cadastro = p\_dt\_cadastro,  
        img\_produto = p\_img\_produto  
    **WHERE**  
        id\_produto = p\_id\_produto;  
  
    **IF** SQL%NOTFOUND THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: ID do Produto a ser atualizado não encontrado.');  
    END **IF**;  
  
    COMMIT;  
END;  
  
  
/

**4.5.3 DELETE BT\_PRODUTO**  
**CREATE** OR REPLACE PROCEDURE delete\_bt\_produto (  
    p\_id\_produto IN bt\_produto.id\_produto%TYPE,  
    p\_nm\_produto OUT bt\_produto.nm\_produto%TYPE  
) IS  
**BEGIN**  
  
    **SELECT** nm\_produto **INTO** p\_nm\_produto  
    **FROM** bt\_produto  
    **WHERE** id\_produto = p\_id\_produto;  
  
      
    **DELETE** **FROM** bt\_produto  
    **WHERE** id\_produto = p\_id\_produto;  
  
    **IF** SQL%NOTFOUND THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: Produto a ser excluído não encontrado.');  
    END **IF**;  
  
    COMMIT;  
  
END;  
/  
 **4.6 CRUD BT\_TELEFONE**

**4.6.1 INSERT BT\_TELEFONE**  
**CREATE** OR REPLACE PROCEDURE insert\_bt\_telefone (  
    p\_id\_telefone  IN bt\_telefone.id\_telefone%TYPE,  
    p\_ddi\_telefone IN bt\_telefone.ddi\_telefone%TYPE,  
    p\_ddd\_telefone IN bt\_telefone.ddd\_telefone%TYPE,  
    p\_nr\_telefone  IN bt\_telefone.nr\_telefone%TYPE  
) IS  
**BEGIN**  
    **INSERT** **INTO** bt\_telefone (  
        id\_telefone,  
        ddi\_telefone,  
        ddd\_telefone,  
        nr\_telefone  
    ) **VALUES** (  
        p\_id\_telefone,  
        p\_ddi\_telefone,  
        p\_ddd\_telefone,  
        p\_nr\_telefone  
    );  
EXCEPTION  
    WHEN DUP\_VAL\_ON\_INDEX THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: Não podemos ter duplicidade de ID na tabela, faça o insert com outro ID.');  
COMMIT;      
END;  
  
/

**4.6.2 UPDATE BT\_TELEFONE**   
**CREATE** OR REPLACE PROCEDURE update\_bt\_telefone (  
    p\_id\_telefone  IN bt\_telefone.id\_telefone%TYPE,  
    p\_ddi\_telefone IN bt\_telefone.ddi\_telefone%TYPE,  
    p\_ddd\_telefone IN bt\_telefone.ddd\_telefone%TYPE,  
    p\_nr\_telefone  IN bt\_telefone.nr\_telefone%TYPE  
) IS  
**BEGIN**  
    **UPDATE** bt\_telefone  
    **SET**  
        ddi\_telefone = p\_ddi\_telefone,  
        ddd\_telefone = p\_ddd\_telefone,  
        nr\_telefone = p\_nr\_telefone  
    **WHERE**  
        id\_telefone = p\_id\_telefone;  
  
    **IF** SQL%NOTFOUND THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: ID do telefone a ser atualizado não encontrado.');  
    END **IF**;  
      
COMMIT;  
END;  
  
/  
 **4.6.3 DELETE BT\_TELEFONE**  
**CREATE** OR REPLACE PROCEDURE delete\_bt\_telefone (  
    p\_id\_telefone IN bt\_telefone.id\_telefone%TYPE,  
    p\_nr\_telefone OUT bt\_telefone.nr\_telefone%TYPE  
) IS  
**BEGIN**  
    
    **SELECT** nr\_telefone **INTO** p\_nr\_telefone  
    **FROM** bt\_telefone  
    **WHERE** id\_telefone = p\_id\_telefone;  
  
    
    **DELETE** **FROM** bt\_telefone  
    **WHERE** id\_telefone = p\_id\_telefone;  
  
    **IF** SQL%ROWCOUNT = 0 THEN  
        DBMS\_OUTPUT.PUT\_LINE('Erro: Telefone a ser excluído não encontrado.');  
    END **IF**;  
COMMIT;  
END;  
  
/

**5 Criamos a procedure GET\_CLIENTES\_COM\_TELEFONE que imprime o nome e número de telefone do cliente.**

**Texto

Descrição gerada automaticamente**

**6 Criamos a procedure CR\_RELATORIO\_ACESSOS\_CLIENTE, que nos mostra a quantidade de acessos de cada cliente, último produto acessado e seu nome para identificação.** Texto

Descrição gerada automaticamente

**7 ARQUIVO PYTHON PARA CHAMADA DAS PROCEDURES**

Texto

Descrição gerada automaticamente

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Texto

Descrição gerada automaticamente Texto

Descrição gerada automaticamente com confiança média

Interface gráfica do usuário

Descrição gerada automaticamente

**8- FUNÇÃO VALIDAR CPF**

Texto

Descrição gerada automaticamente

**8.1 TESTE CORRETO FUNÇÃO VALIDAR CPF**

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**8.2 TESTE COM ERROS FUNÇÃO VALIDAR CPF**

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**9 FUNÇÃO VALIDAR EMAIL**

**Texto

Descrição gerada automaticamente**

**9.1 TESTE CORRETO FUNÇÃO VALIDAR E-MAIL**

Interface gráfica do usuário

Descrição gerada automaticamente

**9.2 TESTE COM ERROS FUNÇÃO VALIDAR E-MAIL**

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**10 FUNÇÃO VERIFICAR MAIORIDADE**

**Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente**

**10.1 SELECT TESTANDO FUNÇÃO VERIFICAR MAIORIDADE**

**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente**

**11 FUNÇÃO TRANSFORMANDO EM JSON**

**Texto

Descrição gerada automaticamente**

**12 PROCEDURE GERAR JSON CLIENTE**

**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente**

**12.1 TESTE CORRETO PROCEDURE GERAR JSON CLIENTE**

**Uma imagem contendo Aplicativo

Descrição gerada automaticamente**

**12.2 TESTE COM ERROS PROCEDURE GERAR JSON CLIENTE**

**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente com confiança média**

**13 PROCEDURE IMPRIMINDO VALORES**

**Texto

Descrição gerada automaticamente**

**13.1 TESTE PROCEDURE IMPRIMINDO VALORES**

**Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente**

**14 CRIANDO TABELA AUDITORIA**

**Interface gráfica do usuário

Descrição gerada automaticamente com confiança baixa**

**15 TRIGGER AUDITORIA TABELA CLIENTE**

*CREATE OR REPLACE TRIGGER trg\_auditoria\_bt\_cliente*

*AFTER INSERT OR UPDATE OR DELETE ON bt\_cliente*

*FOR EACH ROW*

*DECLARE*

*v\_dados\_anteriores CLOB;*

*v\_dados\_novos CLOB;*

*BEGIN*

*IF INSERTING THEN*

*v\_dados\_novos := '{' ||*

*'"id\_cliente": "' || :NEW.id\_cliente || '", ' ||*

*'"id\_telefone": "' || :NEW.id\_telefone || '", ' ||*

*'"id\_genero": "' || :NEW.id\_genero || '", ' ||*

*'"cpf\_cliente": "' || :NEW.cpf\_cliente || '", ' ||*

*'"nm\_cliente": "' || :NEW.nm\_cliente || '", ' ||*

*'"email\_cliente": "' || :NEW.email\_cliente || '", ' ||*

*'"dt\_nascimento\_cliente": "' || :NEW.dt\_nascimento\_cliente || '", ' ||*

*'"estado\_civil\_cliente": "' || :NEW.estado\_civil\_cliente || '", ' ||*

*'"dt\_cadastro": "' || :NEW.dt\_cadastro || '", ' ||*

*'"dt\_exclusao": "' || :NEW.dt\_exclusao || '"}' ;*

*v\_dados\_anteriores := NULL;*

*INSERT INTO BT\_AUDITORIA (*

*id\_auditoria, tabela\_nome, operacao\_tipo, dados\_anteriores, dados\_novos, usuario\_nome, data\_operacao*

*) VALUES (*

*BT\_AUDITORIA\_SEQ.NEXTVAL, 'bt\_cliente', 'INSERT', v\_dados\_anteriores, v\_dados\_novos, SYS\_CONTEXT('USERENV', 'SESSION\_USER'), SYSDATE*

*);*

*ELSIF UPDATING THEN*

*v\_dados\_anteriores := '{' ||*

*'"id\_cliente": "' || :OLD.id\_cliente || '", ' ||*

*'"id\_telefone": "' || :OLD.id\_telefone || '", ' ||*

*'"id\_genero": "' || :OLD.id\_genero || '", ' ||*

*'"cpf\_cliente": "' || :OLD.cpf\_cliente || '", ' ||*

*'"nm\_cliente": "' || :OLD.nm\_cliente || '", ' ||*

*'"email\_cliente": "' || :OLD.email\_cliente || '", ' ||*

*'"dt\_nascimento\_cliente": "' || :OLD.dt\_nascimento\_cliente || '", ' ||*

*'"estado\_civil\_cliente": "' || :OLD.estado\_civil\_cliente || '", ' ||*

*'"dt\_cadastro": "' || :OLD.dt\_cadastro || '", ' ||*

*'"dt\_exclusao": "' || :OLD.dt\_exclusao || '"}';*

*v\_dados\_novos := '{' ||*

*'"id\_cliente": "' || :NEW.id\_cliente || '", ' ||*

*'"id\_telefone": "' || :NEW.id\_telefone || '", ' ||*

*'"id\_genero": "' || :NEW.id\_genero || '", ' ||*

*'"cpf\_cliente": "' || :NEW.cpf\_cliente || '", ' ||*

*'"nm\_cliente": "' || :NEW.nm\_cliente || '", ' ||*

*'"email\_cliente": "' || :NEW.email\_cliente || '", ' ||*

*'"dt\_nascimento\_cliente": "' || :NEW.dt\_nascimento\_cliente || '", ' ||*

*'"estado\_civil\_cliente": "' || :NEW.estado\_civil\_cliente || '", ' ||*

*'"dt\_cadastro": "' || :NEW.dt\_cadastro || '", ' ||*

*'"dt\_exclusao": "' || :NEW.dt\_exclusao || '"}' ;*

*INSERT INTO BT\_AUDITORIA (*

*id\_auditoria, tabela\_nome, operacao\_tipo, dados\_anteriores, dados\_novos, usuario\_nome, data\_operacao*

*) VALUES (*

*BT\_AUDITORIA\_SEQ.NEXTVAL, 'bt\_cliente', 'UPDATE', v\_dados\_anteriores, v\_dados\_novos, SYS\_CONTEXT('USERENV', 'SESSION\_USER'), SYSDATE*

*);*

*ELSIF DELETING THEN*

*v\_dados\_anteriores := '{' ||*

*'"id\_cliente": "' || :OLD.id\_cliente || '", ' ||*

*'"id\_telefone": "' || :OLD.id\_telefone || '", ' ||*

*'"id\_genero": "' || :OLD.id\_genero || '", ' ||*

*'"cpf\_cliente": "' || :OLD.cpf\_cliente || '", ' ||*

*'"nm\_cliente": "' || :OLD.nm\_cliente || '", ' ||*

*'"email\_cliente": "' || :OLD.email\_cliente || '", ' ||*

*'"dt\_nascimento\_cliente": "' || :OLD.dt\_nascimento\_cliente || '", ' ||*

*'"estado\_civil\_cliente": "' || :OLD.estado\_civil\_cliente || '", ' ||*

*'"dt\_cadastro": "' || :OLD.dt\_cadastro || '", ' ||*

*'"dt\_exclusao": "' || :OLD.dt\_exclusao || '"}' ;*

*v\_dados\_novos := NULL;*

*INSERT INTO BT\_AUDITORIA (*

*id\_auditoria, tabela\_nome, operacao\_tipo, dados\_anteriores, dados\_novos, usuario\_nome, data\_operacao*

*) VALUES (*

*BT\_AUDITORIA\_SEQ.NEXTVAL, 'bt\_cliente', 'DELETE', v\_dados\_anteriores, v\_dados\_novos, SYS\_CONTEXT('USERENV', 'SESSION\_USER'), SYSDATE*

*);*

*END IF;*

*END;*

*/*

**15.1 TESTE TRIGGER COM INSERT**

**Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente**

**Aplicativo

Descrição gerada automaticamente com confiança média**

**15.2 TESTE TRIGGER COM UPDATE**

**Interface gráfica do usuário, Texto, Aplicativo, Word

Descrição gerada automaticamente**

**15.3 TESTE TRIGGER COM DELETE**

**Uma imagem contendo Tabela

Descrição gerada automaticamente**