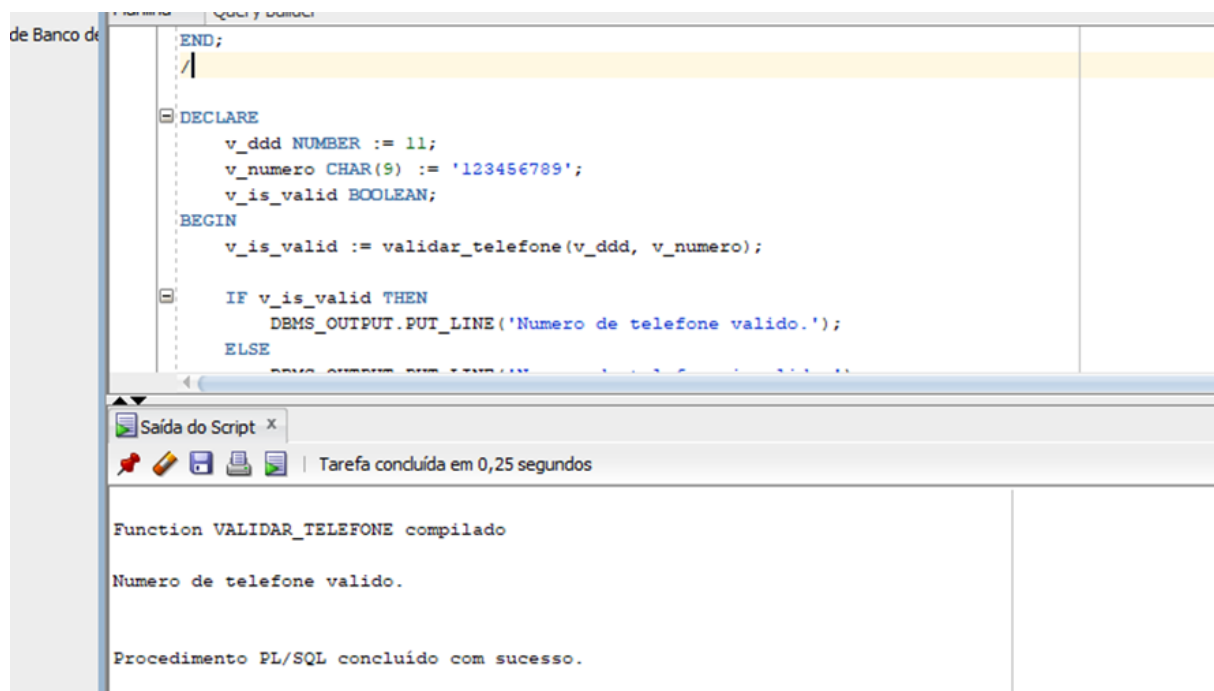


DataBase Design

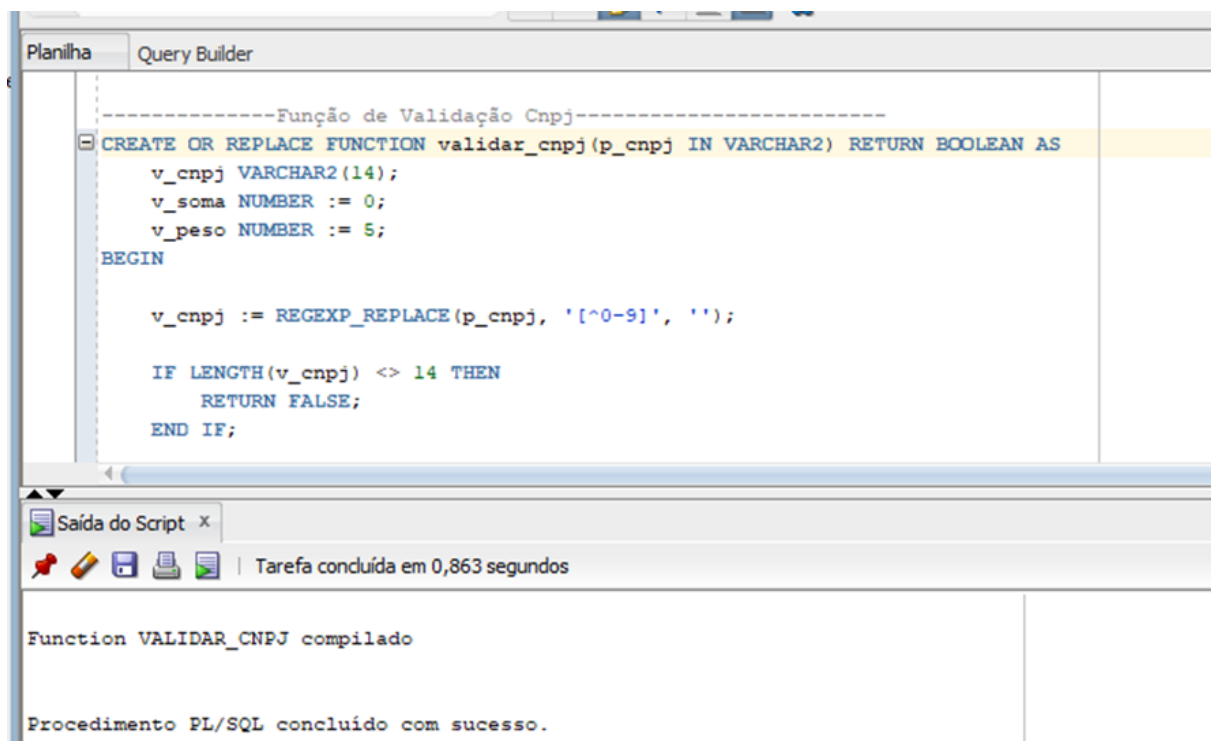
Documentação Print Comprovação

Função Validação de telefone

```
set serveroutput on;
CREATE OR REPLACE FUNCTION validar_telefone (
    p_ddd IN NUMBER,
    p_numero IN CHAR
) RETURN BOOLEAN
IS
    v_pattern VARCHAR2(30) := '^\\d{9}$';
BEGIN
    IF REGEXP_LIKE(p_numero, v_pattern) THEN
        RETURN TRUE;
    ELSE
        RETURN FALSE;
    END IF;
END;
/
```



Função Validação de CNPJ



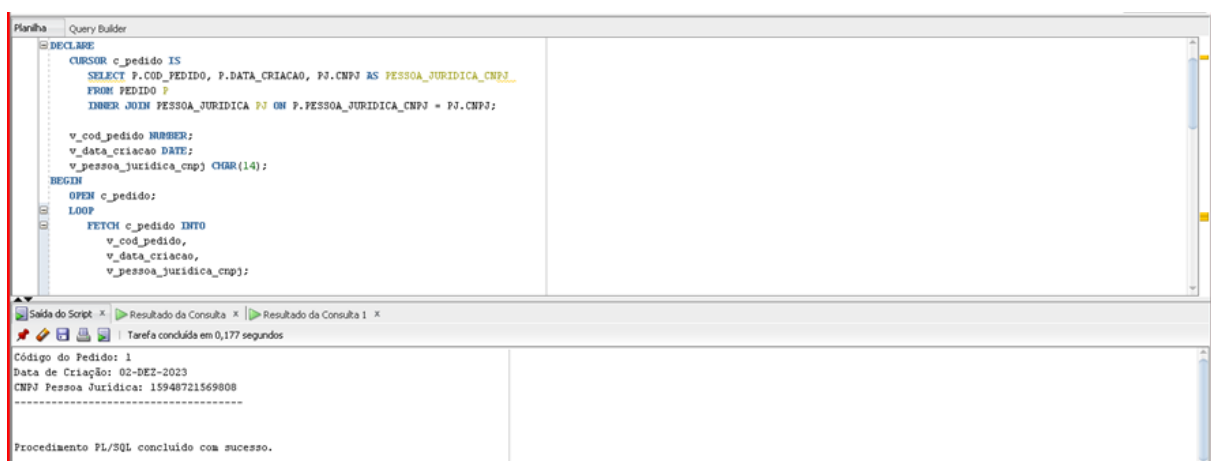
The screenshot shows the SQL Developer interface with the 'Query Builder' tab active. The main editor contains the following PL/SQL code for a function named 'validar_cnpj':

```
-----Função de Validação Cnpj-----  
CREATE OR REPLACE FUNCTION validar_cnpj(p_cnpj IN VARCHAR2) RETURN BOOLEAN AS  
    v_cnpj VARCHAR2(14);  
    v_soma NUMBER := 0;  
    v_peso NUMBER := 5;  
BEGIN  
    v_cnpj := REGEXP_REPLACE(p_cnpj, '[^0-9]', '');  
  
    IF LENGTH(v_cnpj) <> 14 THEN  
        RETURN FALSE;  
    END IF;
```

Below the editor, the 'Saída do Script' (Script Output) window shows the execution results:

```
Function VALIDAR_CNPJ compilado  
  
Procedimento PL/SQL concluído com sucesso.
```

Bloco anônimo com cursor para pelo uma consulta no banco de dados com um Join



The screenshot shows the SQL Developer interface with the 'Query Builder' tab active. The main editor contains the following PL/SQL code for an anonymous block:

```
DECLARE  
    CURSOR c_pedido IS  
        SELECT P.COD_PEDIDO, P.DATA_CRIACAO, PJ.CNPJ AS PESSOA_JURIDICA_CNPJ  
        FROM PEDIDO P  
        INNER JOIN PESSOA_JURIDICA PJ ON P.PESSOA_JURIDICA_CNPJ = PJ.CNPJ;  
  
    v_cod_pedido NUMBER;  
    v_data_criacao DATE;  
    v_pessoa_juridica_cnpj CHAR(14);  
BEGIN  
    OPEN c_pedido;  
    LOOP  
        FETCH c_pedido INTO  
            v_cod_pedido,  
            v_data_criacao,  
            v_pessoa_juridica_cnpj;
```

Below the editor, the 'Saída do Script' (Script Output) window shows the execution results:

```
Código do Pedido: 1  
Data de Criação: 02-DEZ-2023  
CNPJ Pessoa Jurídica: 15948721569808  
  
Procedimento PL/SQL concluído com sucesso.
```

Criar uma procedure que imprima um relatório com pelo menos uma regra de negócio, que contenha funções, inner Join, order by, sum ou Count

Planilha Query Builder

```
-- Criar uma procedure que imprima um relatório com pelo menos uma regra de negócio, que contenha função
SET SERVEROUTPUT ON;

CREATE OR REPLACE PROCEDURE gerar_relatorio IS
    v_valor_minimo NUMBER := 200;
BEGIN
    FOR categoria_rec IN (
        SELECT c.categoria, COUNT(p.cod_produto) AS num_produtos, SUM(p.valor_unitario) AS valor_total
        FROM categoria c
        INNER JOIN produto p ON c.cod_categoria = p.categoria_cod_categoria
        WHERE p.valor_unitario > v_valor_minimo
        GROUP BY c.categoria
        ORDER BY c.categoria
    )
```

Saída do Script x

Tarefa concluída em 0,226 segundos

```
Procedure GERAR_RELATORIO compilado

Categoria: squi
Número de Produtos: 1
Valor Total: $232.
-----
```

Procedures procedures para a realização de Insert, update e delete

```
-----Criar uma procedure que imprima um relatório com pelo m
CREATE OR REPLACE PROCEDURE gerenciar_pais (
    p_cod_pais IN NUMBER,
    p_nome IN VARCHAR2,
    p_operacao IN VARCHAR2
) AS
BEGIN
    IF p_operacao = 'INSERT' THEN
        INSERT INTO pais (cod_pais, nome)
        VALUES (p_cod_pais, p_nome);
    ELSIF p_operacao = 'UPDATE' THEN
        UPDATE pais
```

Saída do Script x

Tarefa concluída em 0,334 segundos

```
Procedure GERENCIAR_PAIS compilado

Procedimento PL/SQL concluído com sucesso.
```

```
EXEC gerenciar_pais(1,'Holanda','INSERT');

CREATE OR REPLACE PROCEDURE gerenciar_estado (
    p_cod_estado IN NUMBER,
    p_nome IN VARCHAR,
    p_pais_cod_pais IN NUMBER,
    p_operacao IN VARCHAR2
) AS
BEGIN
    IF p_operacao = 'INSERT' THEN
        INSERT INTO estado (cod_estado, nome, pais_cod_pais)
        VALUES (p_cod_estado, p_nome, p_pais_cod_pais);
    ELSIF p_operacao = 'UPDATE' THEN
        UPDATE estado
        SET nome = p_nome, pais_cod_pais = p_pais_cod_pais
        WHERE cod_estado = p_cod_estado;
    ELSIF p_operacao = 'DELETE' THEN
        DELETE FROM estado
        WHERE cod_estado = p_cod_estado;
END;
```

Saída do Script x

Tarefa concluída em 0,21 segundos

Procedure GERENCIAR_ESTADO compilado

Procedimento PL/SQL concluído com sucesso.

Planilha Query Builder

```
-----Procedures cidade-----
CREATE OR REPLACE PROCEDURE gerenciar_cidade (
    p_cod_cidade IN NUMBER,
    p_nome IN VARCHAR2,
    p_cod_ibge IN NUMBER,
    p_estado_cod_estado IN NUMBER,
    p_operacao IN VARCHAR2
) AS
BEGIN
    IF p_operacao = 'INSERT' THEN
        INSERT INTO cidade (cod_cidade, nome, cod_ibge, estado_cod_estado)
        VALUES (p_cod_cidade, p_nome, p_cod_ibge, p_estado_cod_estado);
    ELSIF p_operacao = 'UPDATE' THEN
        UPDATE cidade
        SET nome = p_nome, cod_ibge = p_cod_ibge, estado_cod_estado = p_estado_cod_estado
        WHERE cod_cidade = p_cod_cidade;
    ELSIF p_operacao = 'DELETE' THEN
        DELETE FROM cidade
        WHERE cod_cidade = p_cod_cidade;
END;
```

Saída do Script x

Tarefa concluída em 0,219 segundos

Procedure GERENCIAR_CIDADE compilado

Procedimento PL/SQL concluído com sucesso.

Platina Query builder

```
-----Procedures bairro-----
CREATE OR REPLACE PROCEDURE gerenciar_bairro (
    p_cod_bairro IN NUMBER,
    p_nome IN VARCHAR2,
    p_cidade_cod_cidade IN NUMBER,
    p_operacao IN VARCHAR2
) AS
BEGIN
    IF p_operacao = 'INSERT' THEN
        INSERT INTO bairro (cod_bairro, nome, cidade_cod_cidade)
        VALUES (p_cod_bairro, p_nome, p_cidade_cod_cidade);
    ELSIF p_operacao = 'UPDATE' THEN
        UPDATE bairro
        SET nome = p_nome, cidade_cod_cidade = p_cidade_cod_cidade
        WHERE cod_bairro = p_cod_bairro;
    ELSIF p_operacao = 'DELETE' THEN
        DELETE FROM bairro
        WHERE cod_bairro = p_cod_bairro;
    END IF;
END;
```

Saída do Script x

Tarefa concluída em 0,202 segundos

Procedure GERENCIAR_BAIRRO compilado

Procedimento PL/SQL concluído com sucesso.

Planilha Query Builder

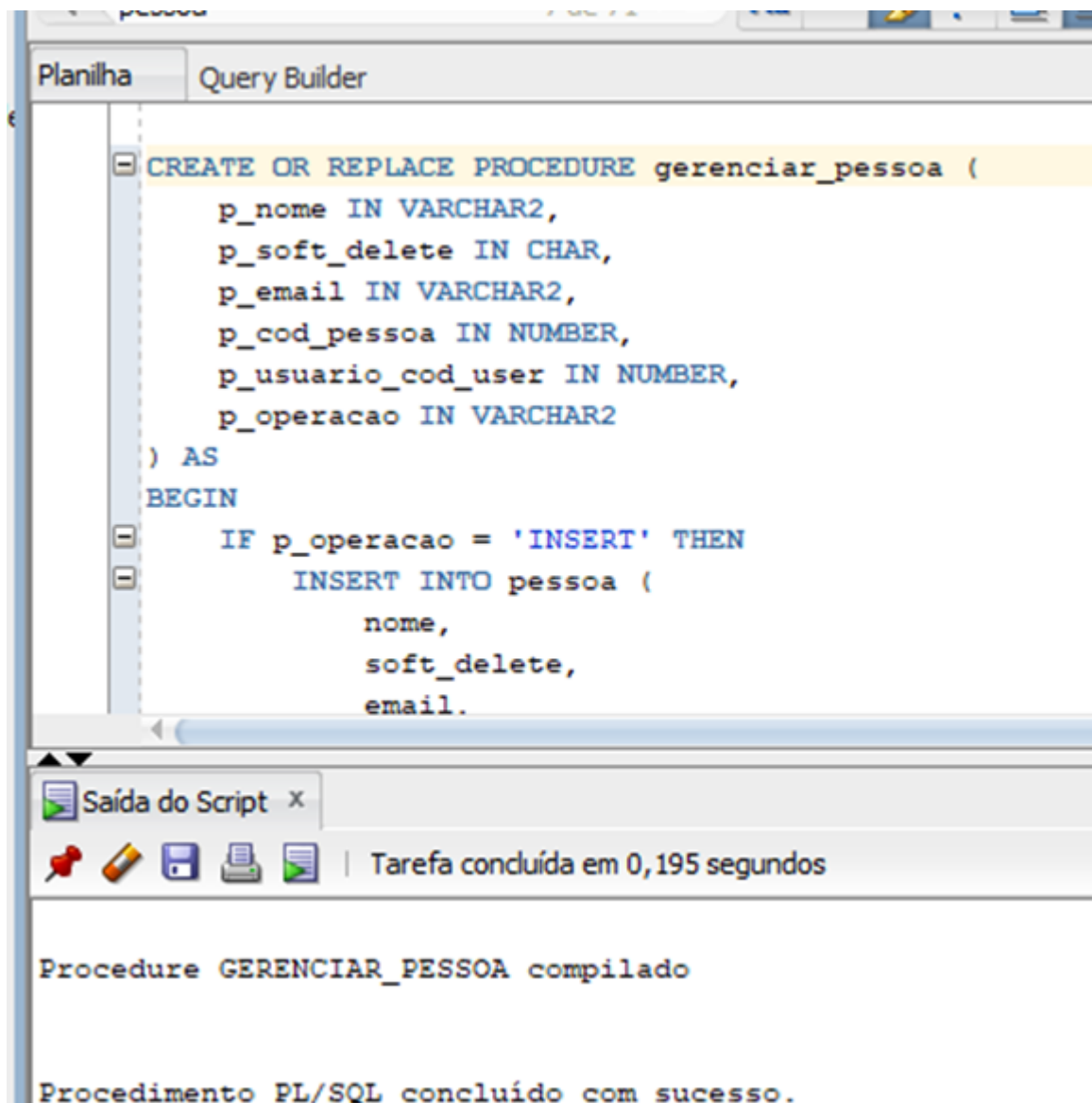
```
CREATE OR REPLACE PROCEDURE gerenciar_usuario (  
    p_nome IN VARCHAR2,  
    p_senha IN VARCHAR2,  
    p_cod_user IN NUMBER,  
    p_operacao IN VARCHAR2  
) AS  
BEGIN  
    IF p_operacao = 'INSERT' THEN  
        INSERT INTO usuario (  
            nome,  
            senha,  
            cod_user  
        ) VALUES (  
            p_nome,  
            p_senha,  
            p_cod_user  
        );  
    ELSIF p_operacao = 'UPDATE' THEN  
        UPDATE usuario
```

Saída do Script x

Tarefa concluída em 0,214 segundos

Procedure GERENCIAR_USUARIO compilado

Procedimento PL/SQL concluído com sucesso.



pessoa 7 de 71

Planilha Query Builder

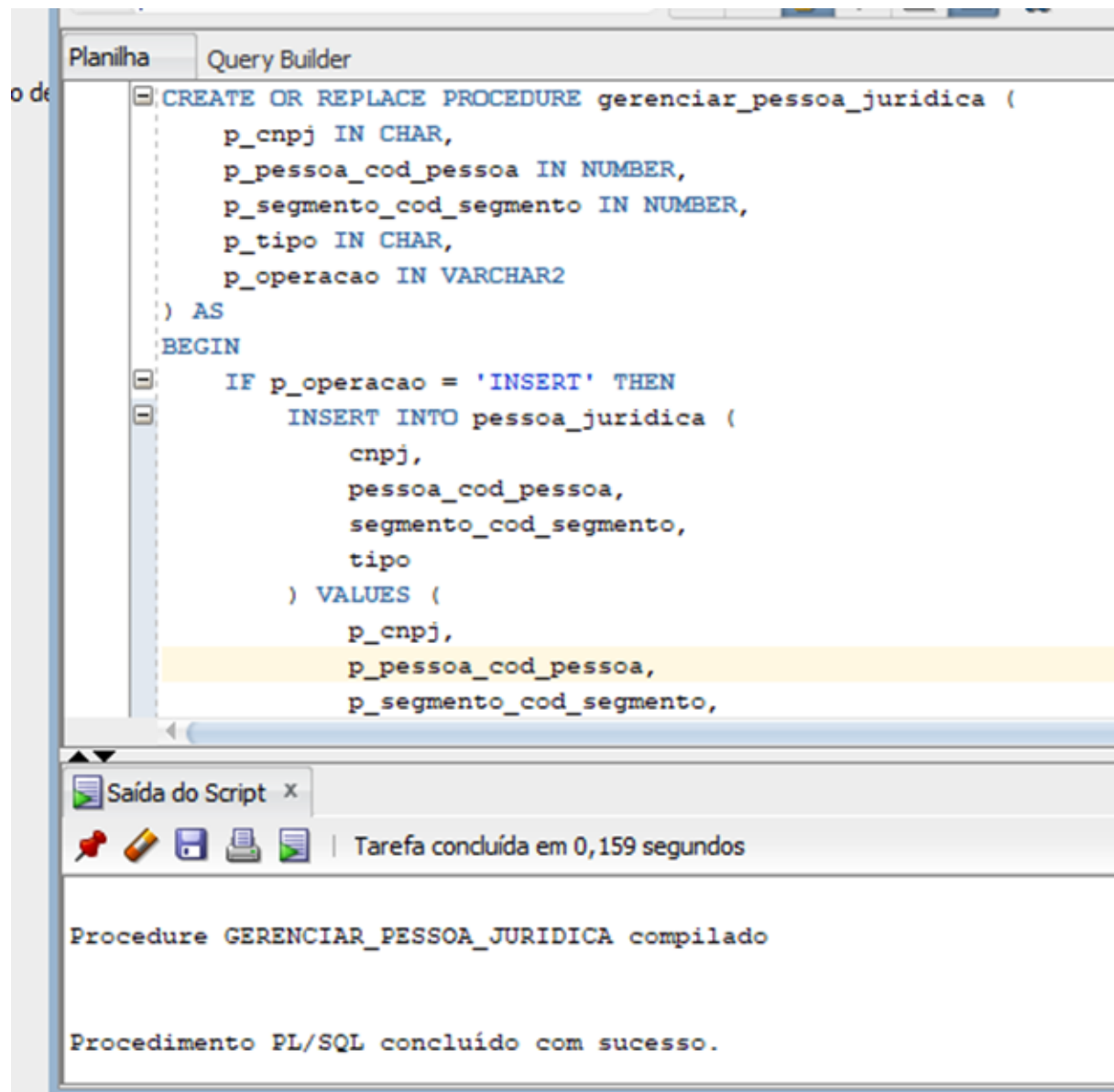
```
CREATE OR REPLACE PROCEDURE gerenciar_segmento (  
    p_cod_segmento IN NUMBER,  
    p_nome IN VARCHAR2,  
    p_operacao IN VARCHAR2  
) AS  
BEGIN  
    IF p_operacao = 'INSERT' THEN  
        INSERT INTO segmento (  
            cod_segmento,  
            nome  
        ) VALUES (  
            p_cod_segmento,  
            p_nome  
        );  
    ELSIF p_operacao = 'UPDATE' THEN
```

Saída do Script x

Tarefa concluída em 0,174 segundos

Procedure GERENCIAR_SEGMENTO compilado

Procedimento PL/SQL concluído com sucesso.



plano de

7 de 71

Planilha Query Builder

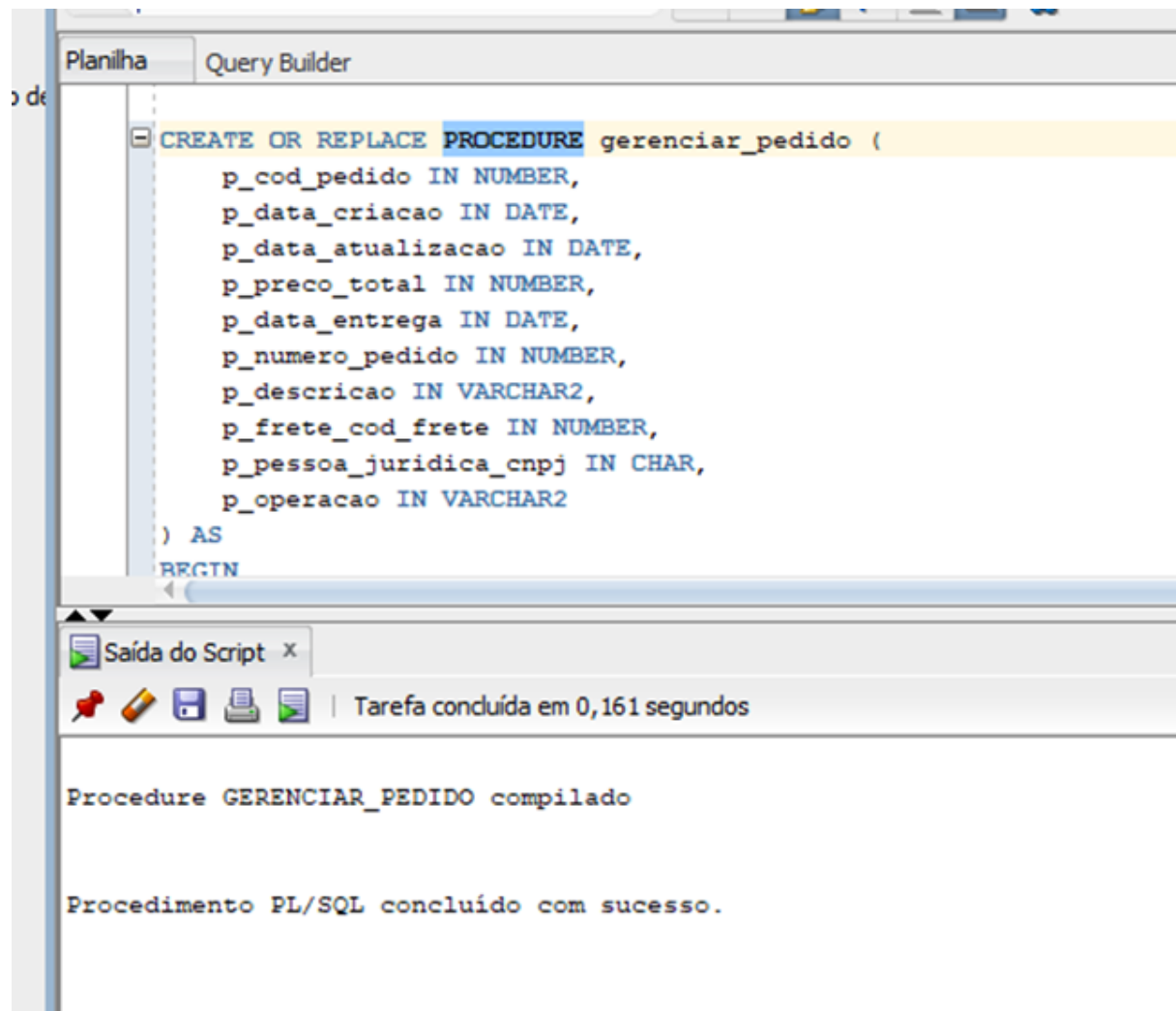
```
CREATE OR REPLACE PROCEDURE gerenciar_frete (  
    p_cod_frete IN NUMBER,  
    p_custo IN NUMBER,  
    p_operacao IN VARCHAR2  
) AS  
BEGIN  
    IF p_operacao = 'INSERT' THEN  
        INSERT INTO frete (cod_frete, custo)  
        VALUES (p_cod_frete, p_custo);  
    ELSIF p_operacao = 'UPDATE' THEN  
        UPDATE frete  
        SET custo = p_custo  
        WHERE cod_frete = p_cod_frete;  
    ELSIF p_operacao = 'DELETE' THEN  
        DELETE FROM frete  
        WHERE cod_frete = p_cod_frete;  
    END IF;
```

Saída do Script x

Tarefa concluída em 0,211 segundos

Procedure GERENCIAR_FRETE compilado

Procedimento PL/SQL concluído com sucesso.



Planilha Query Builder

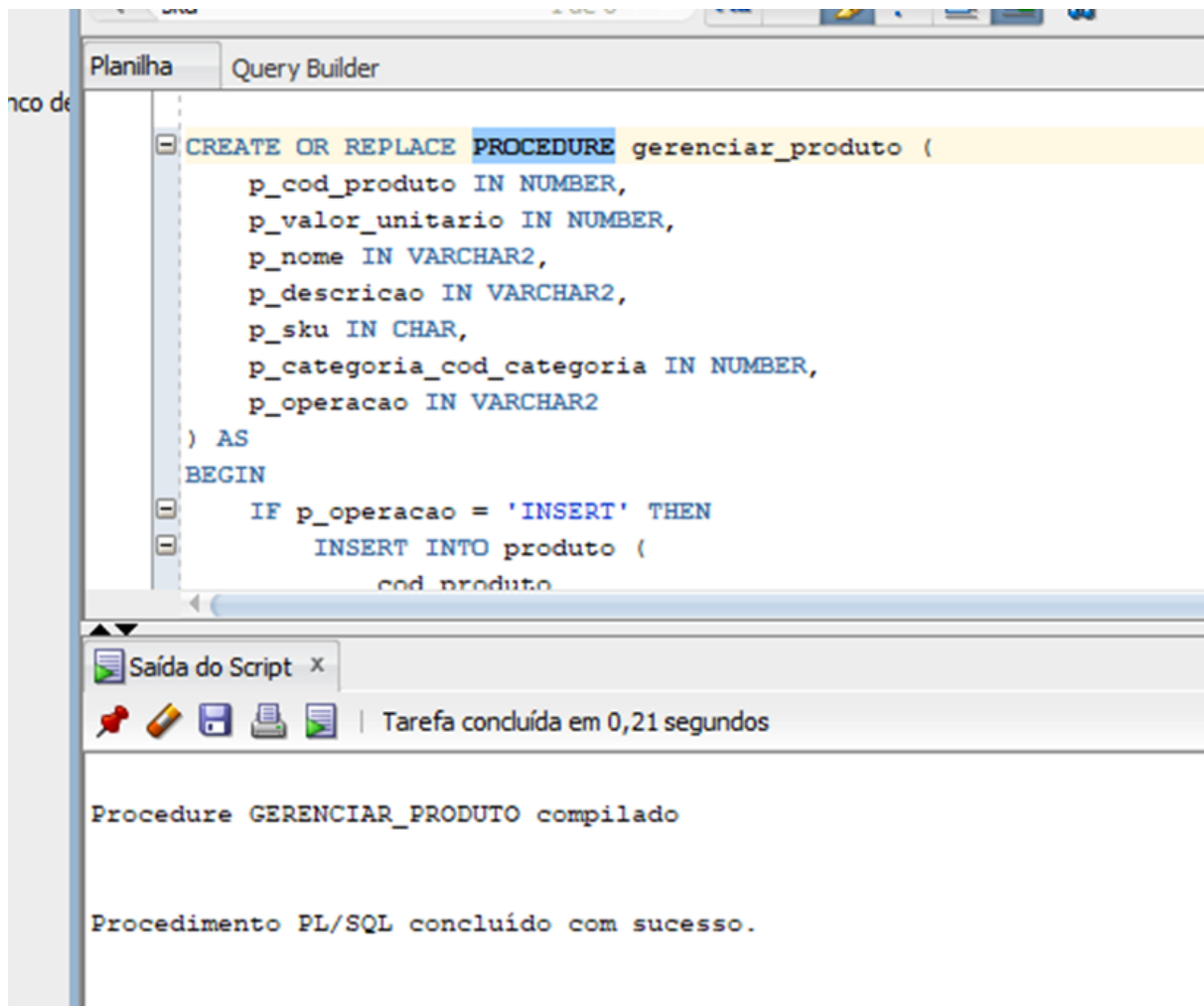
```
CREATE OR REPLACE PROCEDURE gerenciar_categoria (  
    p_cod_categoria IN NUMBER,  
    p_categoria IN VARCHAR2,  
    p_operacao IN VARCHAR2  
) AS  
BEGIN  
    IF p_operacao = 'INSERT' THEN  
        INSERT INTO categoria (cod_categoria, categoria)  
        VALUES (p_cod_categoria, p_categoria);  
    ELSIF p_operacao = 'UPDATE' THEN  
        UPDATE categoria  
        SET categoria = p_categoria  
        WHERE cod_categoria = p_cod_categoria;
```

Saída do Script x

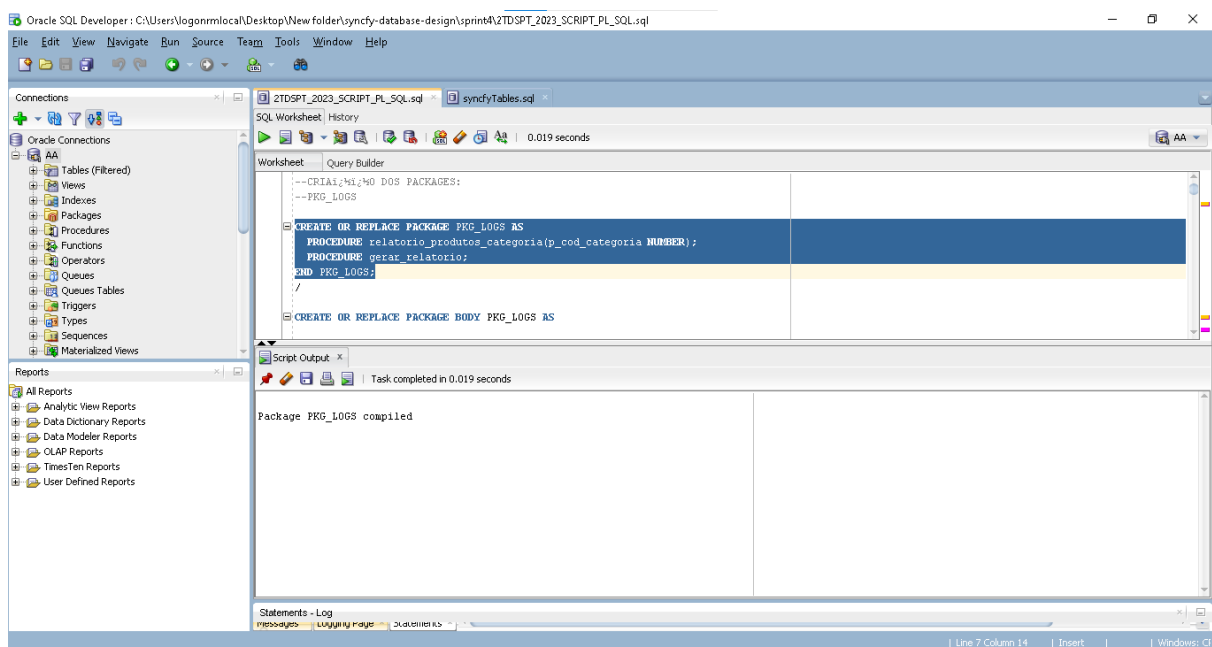
Tarefa concluída em 0,251 segundos

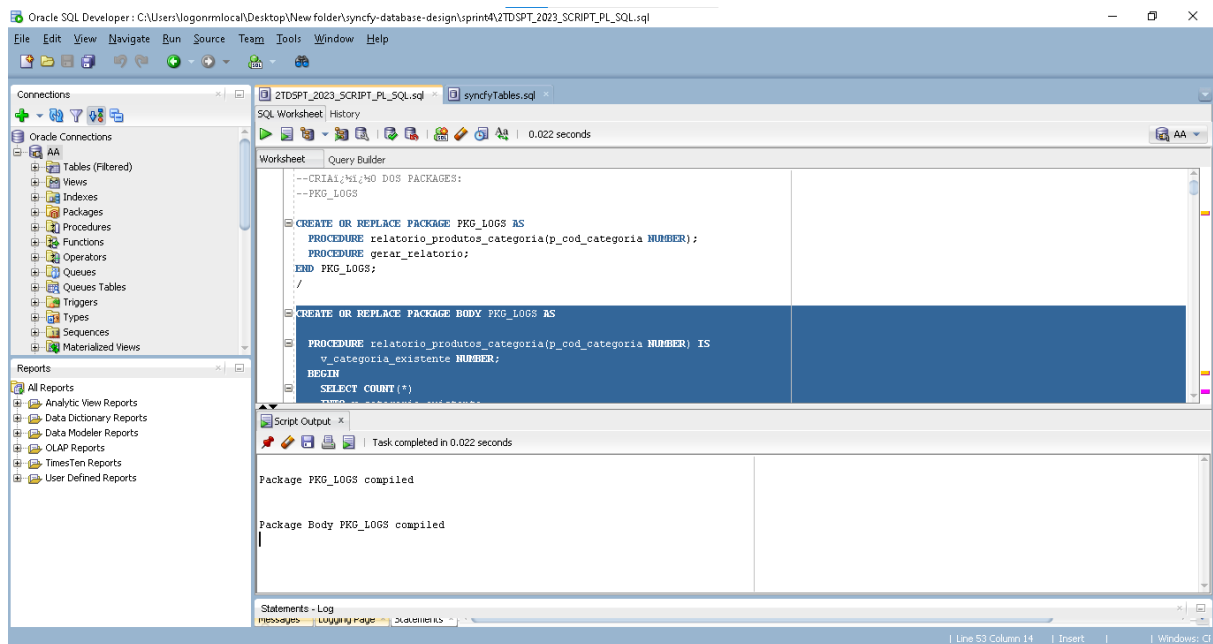
Procedure GERENCIAR_CATEGORIA compilado

Procedimento PL/SQL concluído com sucesso.

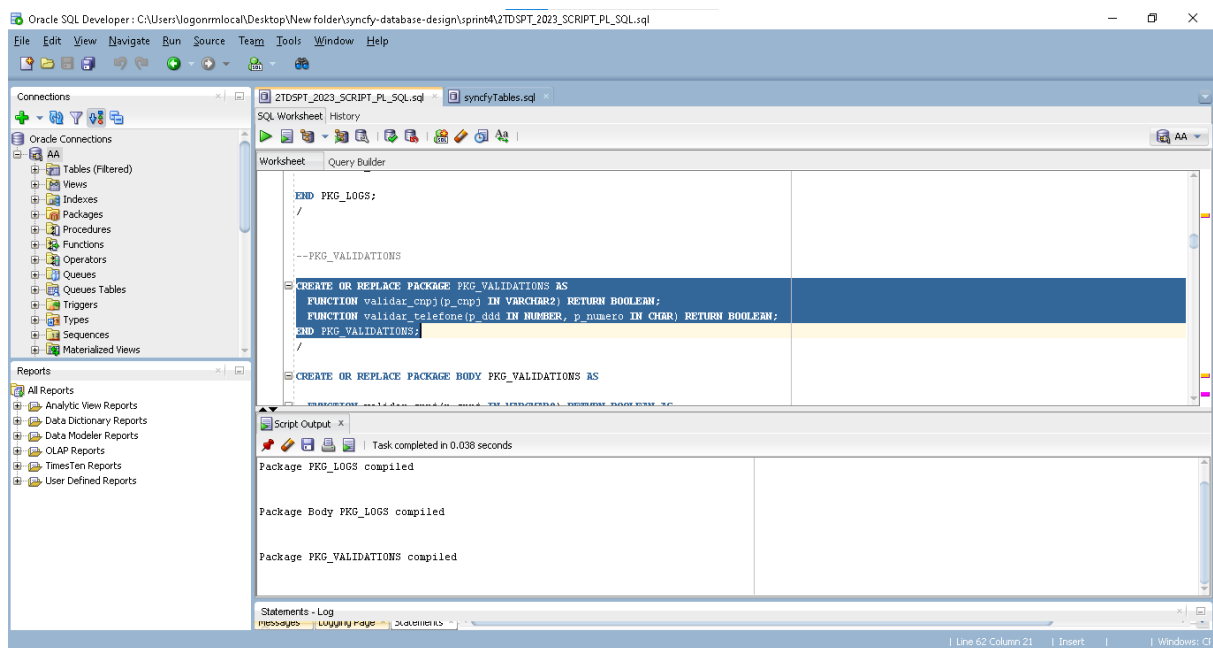


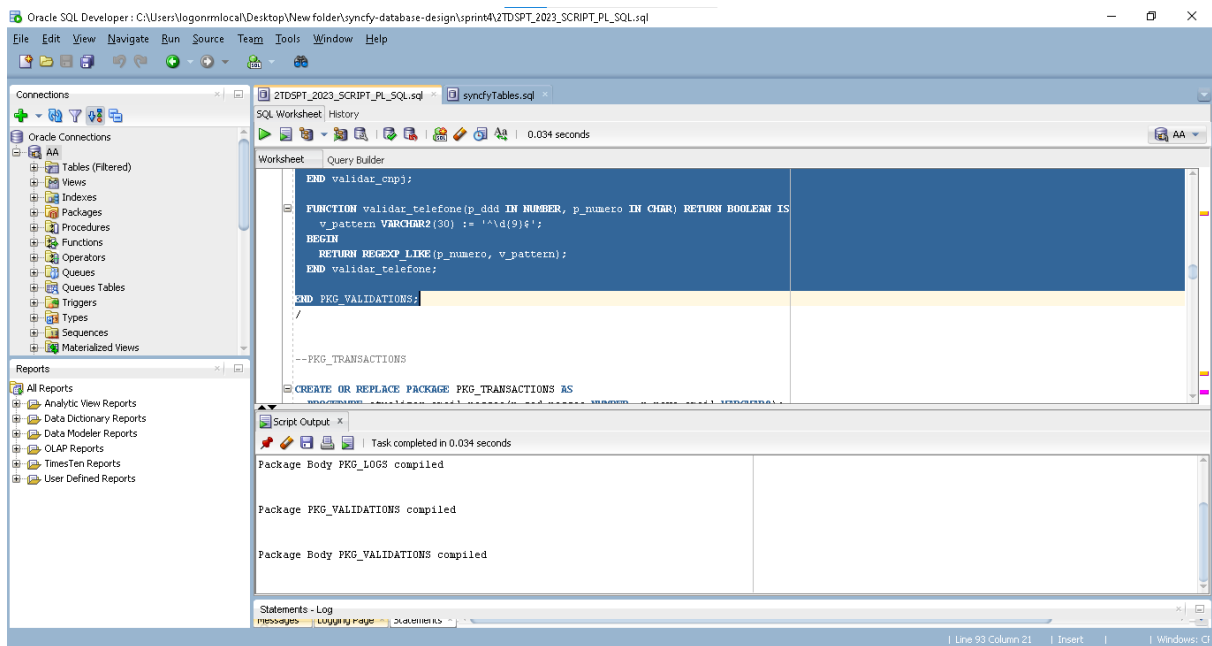
Criação do package e package body do pacote de logs



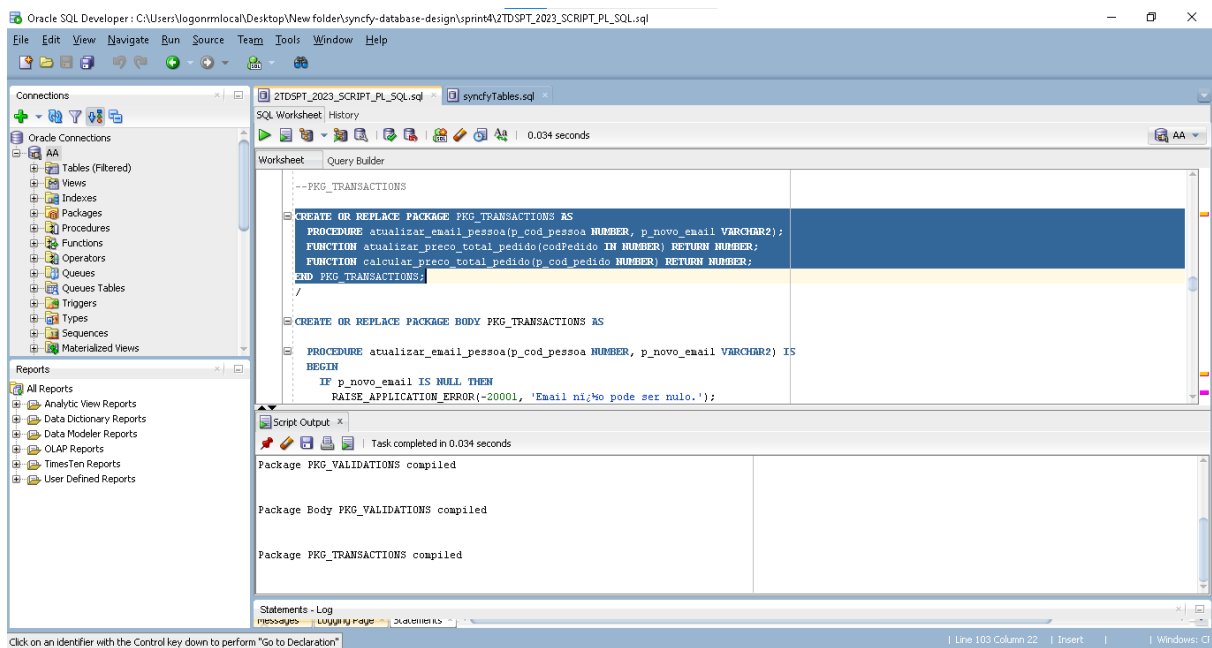


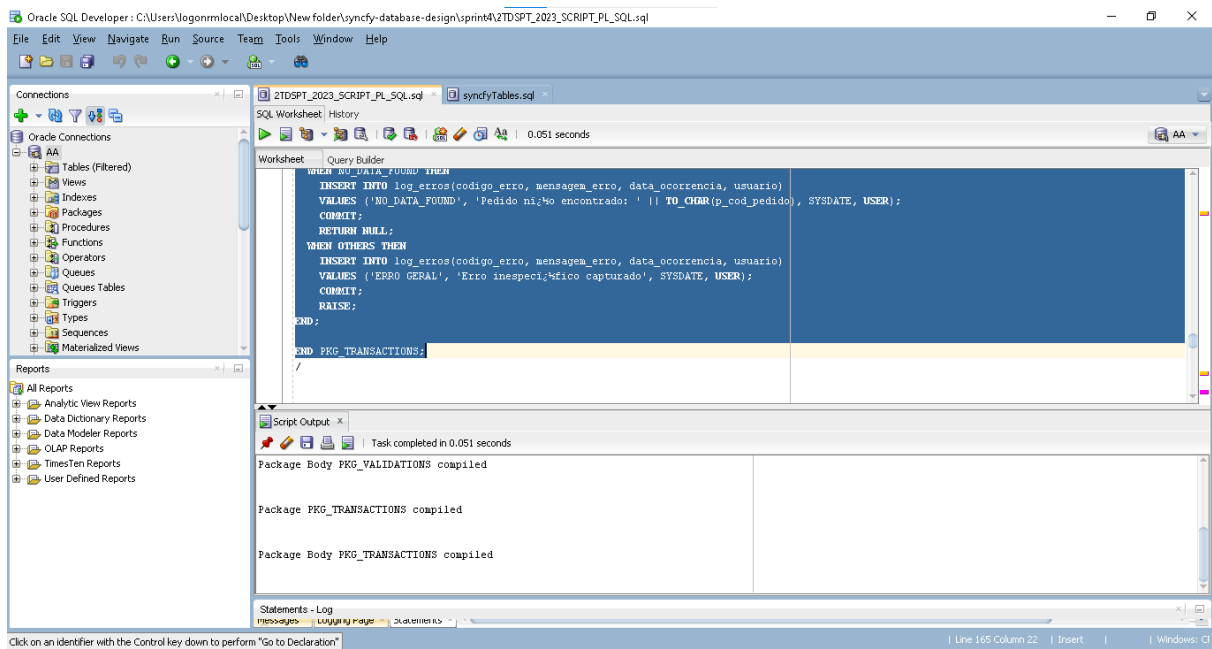
Criação do package e package body do pacote de validação



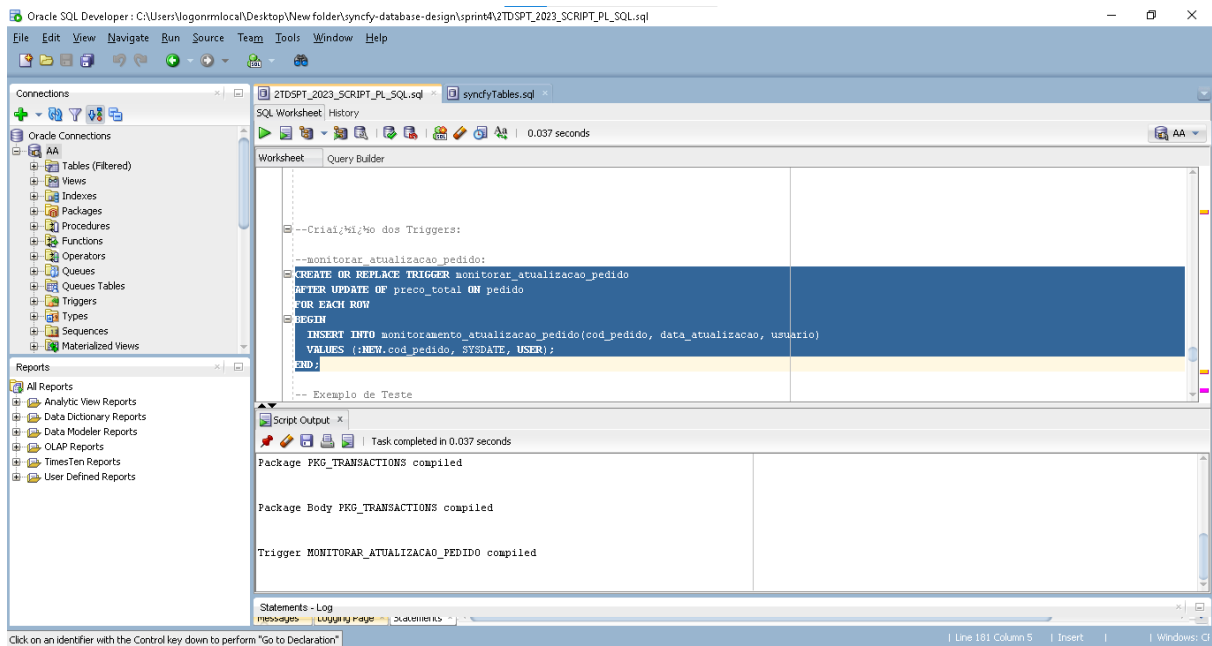


Criação do package e package body do pacote de transações

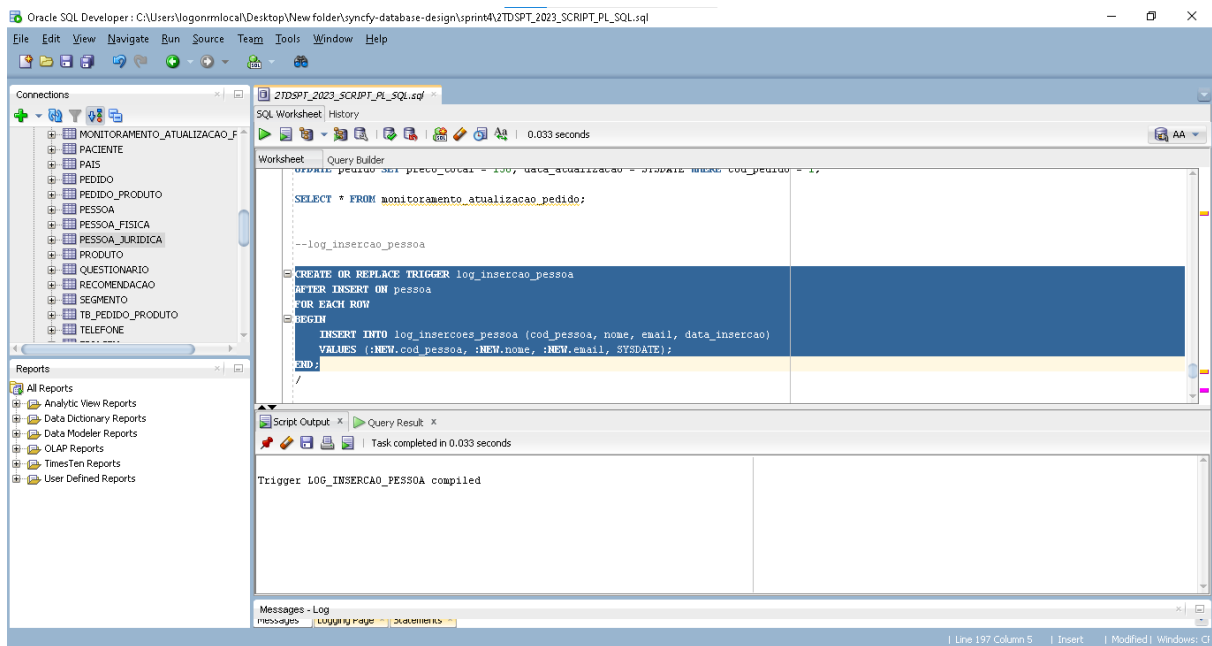




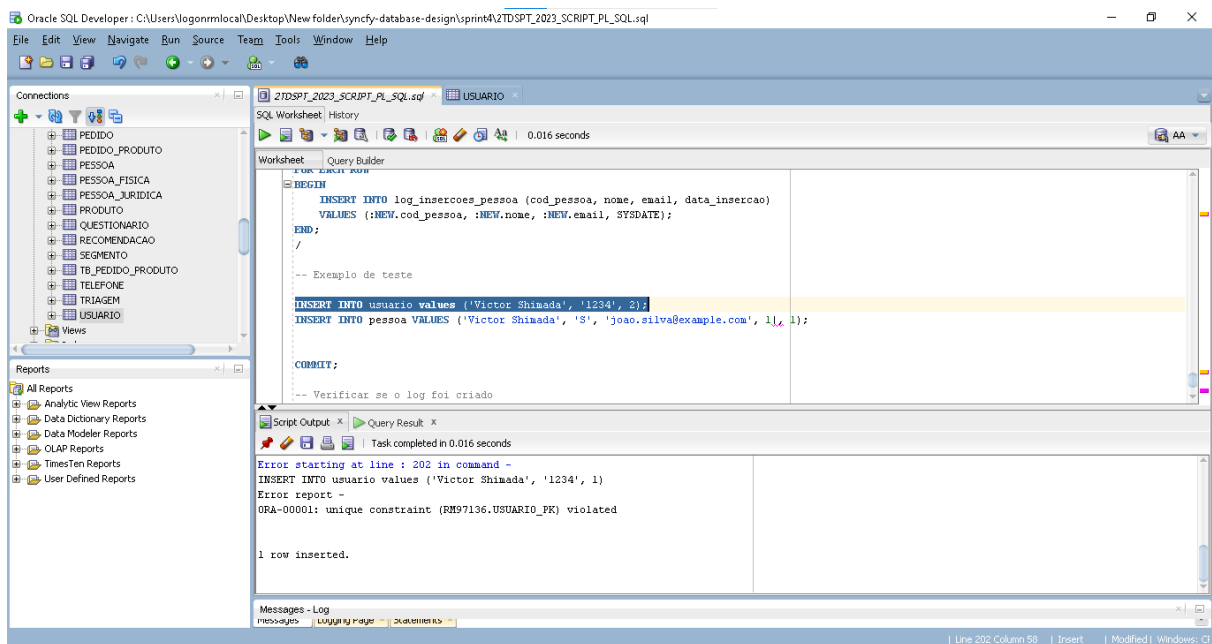
Trigger para monitorar atualização do pedido

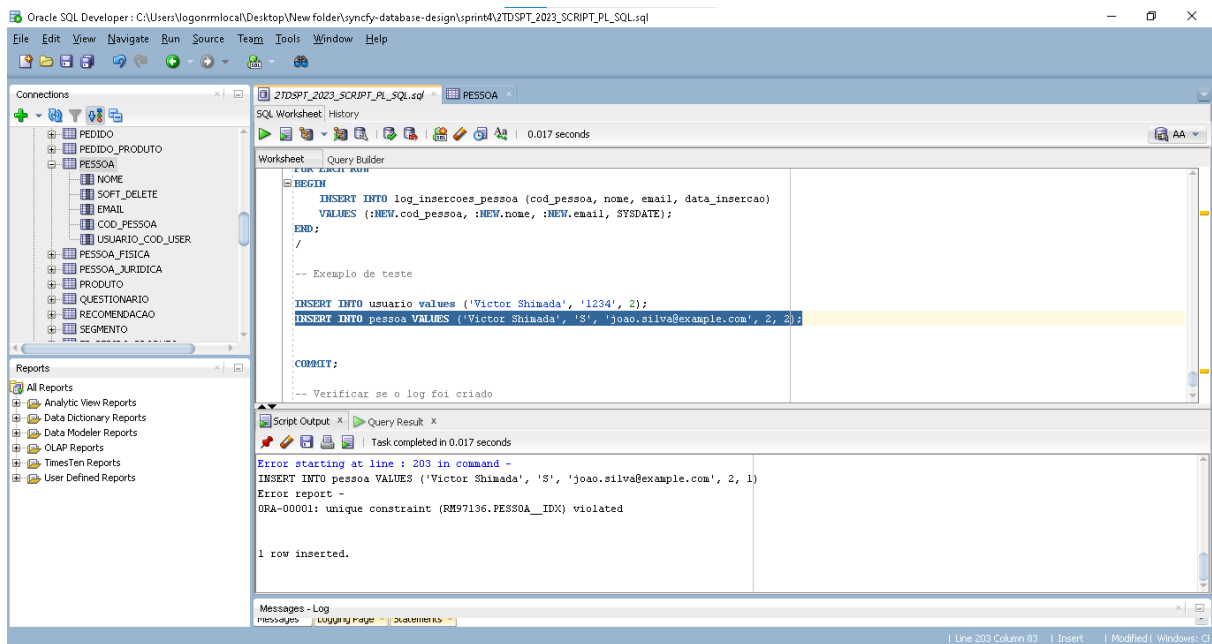


Trigger de log inserção de pessoa

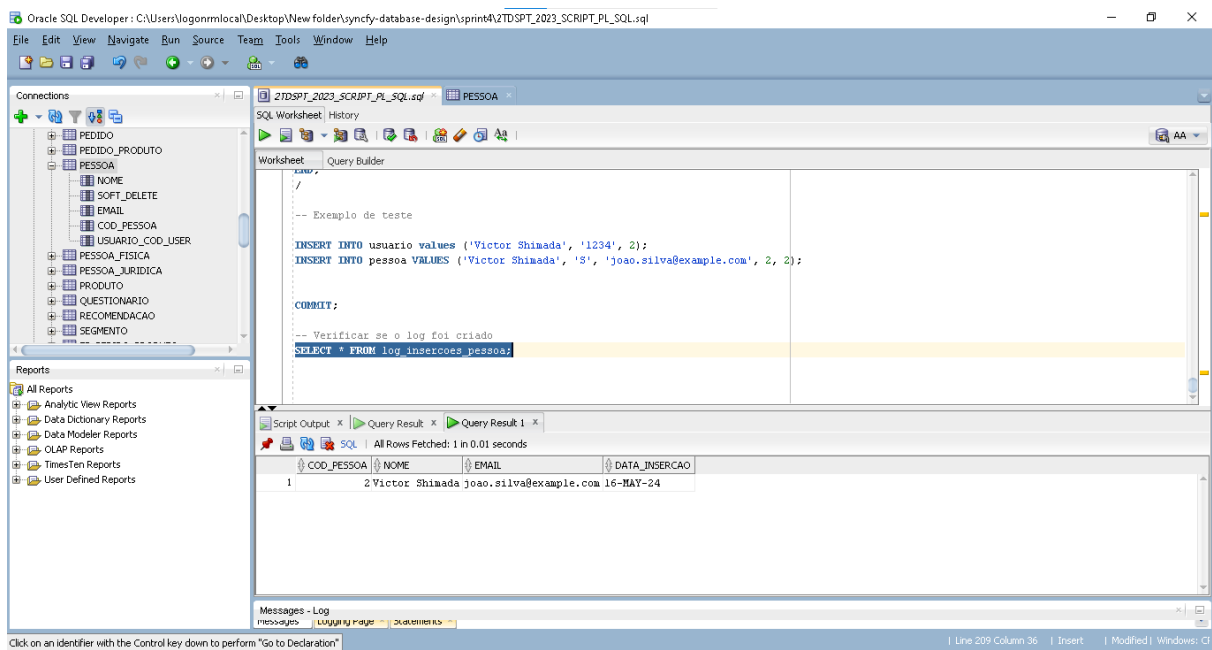


Exemplo de teste trigger de log inserção de pessoa:





Verificação de log criado



Exemplo de teste do trigger de monitorar atualização do pedido

Oracle SQL Developer: C:\Users\logonmlocal\Desktop\New folder\syncfy-database-design\spring4\2TDSPT_2023_SCRIPT_PL_SQL.sql

File Edit View Navigate Run Source Team Tools Window Help

Connections

- 2TDSPT_2023_SCRIPT_PL_SQL
- syncfyProceduresCrud.sql

SQL Worksheet History

0.183 seconds

Worksheet Query Builder

```

INSERT INTO monitoramento_atualizacao_pedido(cod_pedido, data_atualizacao, usuario)
VALUES (:NEW.cod_pedido, SYSDATE, USER);
END;

-- Exemplo de Teste
UPDATE pedido SET preco_total = 150, data_atualizacao = SYSDATE WHERE cod_pedido = 1;
SELECT * FROM monitoramento_atualizacao_pedido;

--log_insercao_pessoa
CREATE OR REPLACE TRIGGER log_insercao_pessoa
AFTER INSERT ON pessoa
FOR EACH ROW
BEGIN

```

Script Output x Query Result x Query Result 1 x

SQL All Rows Fetched: 1 in 0.008 seconds

COD_PEDIDO	DATA_ATUALIZACAO	USUARIO
1	16-MAY-24	RM97136

Compiler - Log

Messages Loggging messages Statements Loggging

Line 186 Column 48 | Insert | Modified | Windows: C