Trilha Prática 3

Views

-- View 1: Resumo de vendas por estabelecimento: exibe o total de vendas realizadas por cada estabelecimento, com a quantidade de produtos vendidos e a soma total do valor das vendas.

```
CREATE VIEW vw_resumo_vendas_estab AS
SELECT
	e.cp_cod_estab,
	e.nm_estab,
	COUNT(ep.cp_id_produto) AS qtd_produtos_vendidos,
	SUM(ep.preco_venda_estab) AS total_vendas
FROM tbl_estabelecimento e
JOIN estab_produto ep ON e.cp_cod_estab = ep.cp_cod_estab
GROUP BY e.cp_cod_estab, e.nm_estab;
```

-- View 2: Produtos e seus fornecedores: lista os produtos junto com os respectivos fornecedores, incluindo informações de preço e data de vencimento.

```
CREATE VIEW vw_produtos_fornecedores AS
SELECT
	p.cp_id_produto,
	p.nm_produto,
	f.cp_cod_forn,
	f.nm_forn,
	fp.data_venda_forn,
	fp.data_vencimento,
	fp.preco_venda_forn
FROM tbl_produto p
JOIN forn_produto fp ON p.cp_id_produto = fp.cp_id_produto
JOIN tbl_fornecedor f ON fp.cp_cod_forn = f.cp_cod_forn;
```

Procedures

-- Procedure 1 - Registrar nova venda: insere um novo pedido com seus produtos e calcula o valor total da compra.

```
CREATE OR REPLACE PROCEDURE sp_registrar_venda(
	IN p_cod_cliente INT,
	IN p_data_pedido DATE,
	IN p_hora_pedido TIME,
	IN p_produtos INT[],
	OUT p_id_pedido INT
)
LANGUAGE plpgsql AS $$
DECLARE
	v_valor_total REAL := 0;
```

```
    v_produto_id INT;
    v_preco REAL;
BEGIN
    -- Criar um novo pedido
    INSERT INTO tbl_pedido (ce_cod_cliente, data_pedido, hora_pedido, valor_total)
    VALUES (p_cod_cliente, p_data_pedido, p_hora_pedido, 0)
    RETURNING cp_id_pedido INTO p_id_pedido;

    -- Percorrer os produtos, adicionar ao pedido e calcular o valor total
    FOREACH v_produto_id IN ARRAY p_produtos LOOP
    SELECT preco_venda_estab INTO v_preco
    FROM estab_produto
    WHERE cp_id_produto = v_produto_id
    LIMIT 1;

    INSERT INTO pedido_produto (cp_id_pedido, cp_id_produto)
    VALUES (p_id_pedido, v_produto_id);

    v_valor_total := v_valor_total + v_preco;
    END LOOP;

    -- Atualizar o valor total do pedido
    UPDATE tbl_pedido
    SET valor_total = v_valor_total
    WHERE cp_id_pedido = p_id_pedido;
END;
$$;

-- Procedure 2 - Atualizar preço de venda de um produto em um estabelecimento: permite
atualizar o preço de um produto específico em um estabelecimento.
CREATE OR REPLACE PROCEDURE sp_atualizar_preco_produto(
    IN p_cod_estab INT,
    IN p_id_produto INT,
    IN p_novo_preco REAL
)
LANGUAGE plpgsql AS $$
BEGIN
    UPDATE estab_produto
    SET preco_venda_estab = p_novo_preco
    WHERE cp_cod_estab = p_cod_estab AND cp_id_produto = p_id_produto;

    IF NOT FOUND THEN
    RAISE EXCEPTION 'Produto ou estabelecimento não encontrados.';
    END IF;
END;
$$;
```

Transações

-- Transação 1 - Registrar um novo cliente e seu primeiro pedido: A transação insere um novo cliente e seu primeiro pedido, garantindo que ambas as operações sejam concluídas com sucesso ou revertidas em caso de erro.

```
DO $$
DECLARE
	v_cod_cliente INT;
	v_id_pedido INT;
BEGIN
	-- Início da transação
	BEGIN
	-- Inserir novo cliente
	INSERT INTO tbl_cliente (nm_cliente, cpf_cliente)
	VALUES ('João da Silva', '12345678901')
	RETURNING cp_cod_cliente INTO v_cod_cliente;

	-- Inserir pedido para o novo cliente
	INSERT INTO tbl_pedido (ce_cod_cliente, data_pedido, hora_pedido, valor_total)
	VALUES (v_cod_cliente, CURRENT_DATE, CURRENT_TIME, 150.00)
	RETURNING cp_id_pedido INTO v_id_pedido;

	-- Associar produtos ao pedido
	INSERT INTO pedido_produto (cp_id_pedido, cp_id_produto)
	VALUES (v_id_pedido, 1), (v_id_pedido, 2);

	-- Commit se tudo ocorreu bem
	COMMIT;
	EXCEPTION
	WHEN OTHERS THEN
	-- Rollback em caso de erro
	ROLLBACK;
	RAISE NOTICE 'Erro ao registrar cliente e pedido. Transação revertida.';
	END;
END $$;
```

-- Transação 2 - Registrar uma venda com múltiplos produtos de forma segura: garante que um pedido e seus produtos só sejam inseridos se todas as operações forem bem-sucedidas. Caso ocorra algum erro, a transação é revertida.

```
DO $$
DECLARE
	v_id_pedido INT;
	v_valor_total REAL := 0;
	v_produto_id INT;
	v_preco REAL;
BEGIN
	BEGIN -- Início da transação
	-- Criar novo pedido
```

```
        INSERT INTO tbl_pedido (ce_cod_cliente, data_pedido, hora_pedido, valor_total)
        VALUES (1, '2025-01-20', '14:30:00', 0)
        RETURNING cp_id_pedido INTO v_id_pedido;

        -- Adicionar produtos ao pedido e calcular valor total
        FOR v_produto_id IN (SELECT cp_id_produto FROM tbl_produto LIMIT 3) LOOP
        SELECT preco_venda_estab INTO v_preco
        FROM estab_produto
        WHERE cp_id_produto = v_produto_id
        LIMIT 1;

        INSERT INTO pedido_produto (cp_id_pedido, cp_id_produto)
        VALUES (v_id_pedido, v_produto_id);

        v_valor_total := v_valor_total + v_preco;
        END LOOP;

        -- Atualizar valor total do pedido
        UPDATE tbl_pedido
        SET valor_total = v_valor_total
        WHERE cp_id_pedido = v_id_pedido;

        COMMIT; -- Confirmação da transação
        RAISE NOTICE 'Pedido registrado com sucesso: ID %', v_id_pedido;
        EXCEPTION
        WHEN OTHERS THEN
        ROLLBACK; -- Desfazer transação em caso de erro
        RAISE NOTICE 'Erro ao registrar pedido: %', SQLERRM;
        END;
END $$;
```