

UNIVERSIDADE anhembi morumbi

DISCIPLINA: ENGENHARIA DE DADOS

2 º SEMESTRE DE 2024

AUTORES DO TRABALHO

Gustavo Henrique da Silva, RA: 12523161925

Pedro Henrique Flores da Rocha, RA: 12522213610

Lucas Oliveira Andrade, RA: 12522195771

Professor

ELIAS PEREIRA DA SILVA

São Paulo

2024

**Índice**

1. Introdução ....................................................... Página 3  
   • Descrição do Problema que o Programa Resolve  
   • Objetivo do Programa  
   • Ferramentas e Tecnologias Utilizadas
2. Arquivos do Programa ......................................... Página 3  
   • BRModelo 1: Finalidade do Modelo  
   • BRModelo 2: Finalidade do Modelo  
   • Script: Linguagem Utilizada e Função Principal
3. Descrição do Funcionamento .................................... Página 4  
   • Fluxo de Dados entre os Modelos e o Script  
   • Principais Funções e Algoritmos
4. Descrição do Modelo Conceitual ................................. Página 4 e 5  
   • Entidades e Atributos  
   • Relacionamentos
5. Dicionário de Dados .................................................. Página 5  
   • Tabelas e Campos
6. Especificação das Stored Procedures ............................ Página 5 e 6  
   • GetCashFlowBalance  
   • GetIncomeExpenseReport  
   • GetCashFlowProjection
7. Manual de Utilização do Banco de Dados ....................... Página 6 e 7  
   • Como Inserir Dados  
   • Como Consultar Dados  
   • Como Atualizar Dados  
   • Como Excluir Dados
8. Conclusão ........................................................... Página 7  
   • Resultados Obtidos

**1. Introdução**

* **Descrição do Problema que o Programa Resolve**:  
  O programa desenvolvido tem como objetivo desenvolver um banco de dados relacional para o cálculo e análise de fluxo de caixa de uma empresa.
* **Objetivo do Programa**:  
  O objetivo principal é criar e documentar a modelagem de dados utilizando a ferramenta BRModelo, além de gerar um script SQL de implementação, a partir da modelagem do banco de dados. Isso visa fornecer uma estrutura sólida para o desenvolvimento de sistemas complexos e eficientes.
* **Ferramentas e Tecnologias Utilizadas**:

O programa foi desenvolvido utilizando:

* + **BRModelo**: Ferramenta para modelagem de dados.
  + **SQL (Structured Query Language)**: Linguagem de consulta utilizada para manipulação de dados no banco.
  + **Ambiente de Desenvolvimento**: Sql Server

**2. Arquivos do Programa**

* **BRModelo 1: Finalidade do Modelo:**  
  O arquivo "**MER\_BrModelo.brm**" representa o Modelo Entidade-Relacionamento (MER) que descreve a estrutura das entidades e seus relacionamentos dentro do sistema. Ele define os elementos do banco de dados, suas propriedades, e como eles interagem.
* **BRModelo 2: Finalidade do Modelo:**  
  O arquivo "**DER\_BrModelo.brm**" contém o Diagrama de Entidade-Relacionamento (DER), que detalha as tabelas do banco de dados, suas chaves primárias, estrangeiras, e outras restrições. Ele descreve como o MER é traduzido para a estrutura relacional.
* **Script: Linguagem Utilizada e Função Principal**  
  O arquivo "A3\_EngenhariaDeDados.sql" é um script SQL que contém as instruções para criar as tabelas no banco de dados com base no modelo relacional. Ele inclui comandos como CREATE TABLE, INSERT, ALTER, e outros necessários para a criação e manipulação do banco de dados.

**3. Descrição do Funcionamento**

* **Fluxo de Dados entre os Modelos e o Script**  
  O fluxo de dados começa com a modelagem no BrModelo, onde são definidas as entidades e seus relacionamentos. O modelo MER é então traduzido para um modelo relacional no DER, que é finalmente convertido em um script SQL. Este script é utilizado para criar as tabelas e relacionamentos no banco de dados, que armazenam e manipulam as informações conforme o sistema exige.
* **Principais Funções e Algoritmos**  
  O algoritmo principal do sistema é a criação das tabelas no banco de dados, seguindo o modelo relacional. O script SQL foi projetado para garantir a integridade referencial entre as tabelas, usando chaves primárias e estrangeiras, além de garantir a conformidade com as regras de normalização.

**4. Descrição do Modelo Conceitual**

O modelo conceitual do banco de dados **CashFlowDB** tem como objetivo gerenciar as contas e movimentações financeiras de uma empresa, incluindo receitas, despesas, centros de custo e formas de pagamento. O modelo foi projetado para ser simples e eficiente, permitindo que as transações financeiras sejam registradas e consultadas de forma clara.

**Entidades e Atributos**

1. **Contas (Accounts)**
   * **account\_id**: Identificador único da conta (PK)
   * **account\_name**: Nome da conta (ex: Conta Corrente)
   * **account\_type**: Tipo da conta (ex: Corrente, Poupança)
   * **initial\_balance**: Saldo inicial da conta
2. **Categorias (Categories)**
   * **category\_id**: Identificador único da categoria (PK)
   * **category\_name**: Nome da categoria (ex: Salário, Aluguel)
   * **category\_type**: Tipo da categoria (Receita ou Despesa)
3. **Centros de Custo (CostCenters)**
   * **cost\_center\_id**: Identificador único do centro de custo (PK)
   * **cost\_center\_name**: Nome do centro de custo (ex: Administração, TI)
   * **description**: Descrição do centro de custo

**4. Formas de Pagamento (PaymentMethods)**

* + **payment\_method\_id**: Identificador único da forma de pagamento (PK)
  + **payment\_type**: Tipo de pagamento (ex: Dinheiro, Cartão)

**5. Movimentações Financeiras (FinancialMovements)**

* + **movement\_id**: Identificador único da movimentação (PK)
  + **movement\_date**: Data da movimentação
  + **amount**: Valor da movimentação
  + **description**: Descrição da movimentação
  + **account\_id**: Referência à tabela Accounts (FK)
  + **category\_id**: Referência à tabela Categories (FK)
  + **cost\_center\_id**: Referência opcional à tabela CostCenters (FK)
  + **payment\_method\_id**: Referência à tabela PaymentMethods (FK)

**Relacionamentos**

1. **Accounts -> FinancialMovements** (1:N)
2. **Categories -> FinancialMovements** (1:N)
3. **CostCenters -> FinancialMovements** (0:N)
4. **PaymentMethods -> FinancialMovements** (1:N)

**5. Dicionário de Dados**

Abaixo está a descrição das tabelas com seus respectivos campos e tipos de dados.

**Tabelas e Campos**

1. **Accounts**
   * **account\_id**: INT (PK, Auto Increment)
   * **account\_name**: VARCHAR(50)
   * **account\_type**: VARCHAR(20)
   * **initial\_balance**: FLOAT
2. **Categories**
   * **category\_id**: INT (PK, Auto Increment)
   * **category\_name**: VARCHAR(50)
   * **category\_type**: VARCHAR(10)
3. **CostCenters**
   * **cost\_center\_id**: INT (PK, Auto Increment)
   * **cost\_center\_name**: VARCHAR(50)
   * **description**: VARCHAR(200)
4. **PaymentMethods**
   * **payment\_method\_id**: INT (PK, Auto Increment)
   * **payment\_type**: VARCHAR(20)
5. **FinancialMovements**
   * **movement\_id**: INT (PK, Auto Increment)
   * **movement\_date**: DATE
   * **amount**: DECIMAL(12, 2)
   * **description**: VARCHAR(200)
   * **account\_id**: INT (FK to Accounts)
   * **category\_id**: INT (FK to Categories)
   * **cost\_center\_id**: INT (FK to CostCenters, nullable)
   * **payment\_method\_id**: INT (FK to PaymentMethods)

**6. Especificação das Stored Procedures**

1. **GetCashFlowBalance**
   * **Objetivo**: Calcular o saldo atual de cada conta, somando as receitas e subtraindo as despesas.
   * **Exemplo de uso**:
   * EXEC GetCashFlowBalance;
2. **GetIncomeExpenseReport**
   * **Objetivo**: Gerar um relatório com o total acumulado de receitas e despesas por categoria.
   * **Exemplo de uso**:
   * EXEC GetIncomeExpenseReport;
3. **GetCashFlowProjection**
   * **Objetivo**: Gerar uma projeção do fluxo de caixa para os próximos meses, com base nas movimentações financeiras passadas.
   * **Parâmetro**: @months define o número de meses para a projeção.
   * **Exemplo de uso**:
   * EXEC GetCashFlowProjection @months = 6;

**7. Manual de Utilização do Banco de Dados**

**Como Inserir Dados**

1. **Contas**
   * Para adicionar uma nova conta:
   * INSERT INTO Accounts (account\_name, account\_type, initial\_balance)
   * VALUES ('Conta Corrente', 'C', 5000.00);
2. **Categorias**
   * Para adicionar uma nova categoria:
   * INSERT INTO Categories (category\_name, category\_type)
   * VALUES ('Salário', 'R');
3. **Movimentações Financeiras**
   * Para registrar uma movimentação financeira:
   * INSERT INTO FinancialMovements (movement\_date, amount, description, account\_id, category\_id, payment\_method\_id)
   * VALUES ('2024-12-01', 5000.00, 'Salário recebido', 1, 1, 1);

**Como Consultar Dados**

1. **Saldo das Contas**
   * Para ver o saldo de todas as contas:
   * EXEC GetCashFlowBalance;
2. **Relatório de Receitas e Despesas**
   * Para gerar o relatório de receitas e despesas:
   * EXEC GetIncomeExpenseReport;
3. **Projeção de Fluxo de Caixa**
   * Para ver a projeção do fluxo de caixa para os próximos 6 meses:
   * EXEC GetCashFlowProjection @months = 6;

**Como Atualizar Dados**

1. **Atualizar saldo de conta**
   * Para atualizar o saldo de uma conta:
   * UPDATE Accounts
   * SET initial\_balance = 6000.00
   * WHERE account\_id = 1;

**Como Excluir Dados**

1. **Excluir uma movimentação financeira**
   * Para excluir uma movimentação financeira:
   * DELETE FROM FinancialMovements
   * WHERE movement\_id = 1;

**5. Conclusão**

* **Resultados Obtidos**  
  O programa foi capaz de criar um banco de dados funcional a partir dos modelos de dados definidos, proporcionando uma estrutura relacional eficiente e bem documentada para o sistema.