* Imagem gerada por IA

Common Business Oriented Language

Este artigo introdutório de Linguagem Cobol explora os fundamentos da programação nessa linguagem clássica, utilizada principalmente em sistemas financeiros e empresariais.

Introdução à Linguagem COBOL: História e Evolução

01 | Introdução à Linguagem COBOL: História e Evolução

A Linguagem COBOL (Common Business Oriented Language) é uma das linguagens de programação mais antigas ainda em uso hoje, fundamental para o desenvolvimento de sistemas empresariais. Sua criação remonta à década de 1950, em um contexto onde a computação estava emergindo como uma ferramenta vital para empresas e instituições financeiras.

Origem e Criação

O desenvolvimento do COBOL começou em 1959, quando um grupo de especialistas em computação, conhecido como CODASYL (Conference on Data Systems Languages), se reuniu para criar uma linguagem que fosse capaz de atender às necessidades de processamento de dados da comunidade empresarial. A proposta era que a nova linguagem fosse legível, fácil de entender e usável em diversos tipos de sistemas de computação.

O primeiro programa em COBOL foi escrito em 1960, e a linguagem rapidamente ganhou popularidade por sua capacidade de lidar com registros e arquivos, algo crucial para a manipulação de dados empresariais.

Primeira Padronização

Em 1968, foi feito o primeiro padrão para a Linguagem COBOL, o que ajudou a consolidar seu uso e garantir que os programas pudessem ser portáveis entre diferentes sistemas de computação. O padrão de 1974 foi uma evolução significativa que introduziu melhorias como a possibilidade de utilizar estrutura de controle mais sofisticadas e funções.

Evolução ao Longo dos Anos

Nos anos 80 e 90, o COBOL enfrentou um aumento na concorrência de novas linguagens, como C e Java, mas conseguiu se manter relevante, principalmente em ambientes corporativos. Um dos principais fatores para sua continuidade foi a enorme quantidade de código legado já em operação, que as empresas relutavam em substituir.

Durante essa época, importantes versões do COBOL foram lançadas, incluindo o COBOL 85, que trouxe melhorias significativas em funcionalidades de programação orientada a objetos, facilitando a integração com sistemas mais novos.

Adaptação às Novas Tecnologias

Na virada do século, com a chegada da internet e a crescente demanda por soluções em tempo real, a linguagem também passou por reformulações. O COBOL 2002, por exemplo, introduziu recursos modernos, permitindo que os desenvolvedores implementassem soluções web e outras funcionalidades contemporâneas. Essas atualizações demonstraram que o COBOL poderia evoluir e se adaptar às novas necessidades tecnológicas.

Importância Atual

Atualmente, mesmo com o surgimento de novas linguagens e paradigmas de programação, o COBOL continua a desempenhar um papel vital em várias indústrias. Estima-se que mais de 200 bilhões de linhas de código COBOL ainda esteja em uso, especialmente em bancos, seguradoras e outras organizações onde a integração de sistemas legados é crítica.

O legado do COBOL reside não apenas em sua robustez e eficiência, mas também na grande comunidade de desenvolvedores que ainda trabalham com a linguagem, garantindo seu suporte e evolução.

Conclusão - Introdução à Linguagem COBOL: História e Evolução

A introdução à linguagem COBOL destacou sua história e evolução, mostrando sua importância na programação moderna e sua aplicabilidade em sistemas legados.

Estruturas de Dados e Controle de Fluxo em COBOL

02 | Estruturas de Dados e Controle de Fluxo em COBOL

COBOL (Common Business-Oriented Language) é uma linguagem de programação desenvolvida para lidar com grandes volumes de dados, especialmente em contextos empresariais. Dentro do COBOL, as estruturas de dados e o controle de fluxo são fundamentais para a manipulação eficiente de informações e a implementação de lógicas de programação.

Estruturas de Dados

As estruturas de dados em COBOL permitem a organização e o armazenamento de dados de maneira sistemática. As principais categorias de estruturas de dados no COBOL incluem os tipos de dados primitivos, registros, tabelas e arquivos.

Tipos de Dados

Cobol possui vários tipos de dados, sendo os principais:

- **Alfa**: Usado para armazenar caracteres, como letras e símbolos.
- **Numérico**: Destinado a números, podendo incluir decimais.
- **Decimal**: Semelhante ao tipo numérico, mas permite a definição de precisão e escala.

Registros

Registros (ou RECORDS) são estruturas que permitem a combinação de diferentes tipos de dados sob um único nome. Um registro pode conter vários campos, cada um com seu próprio tipo de dado. Os registros são definidos na seção de dados do programa, utilizando a seguinte sintaxe:

```
01 EMPREGADO.
   05 NOME       PIC A(30).
   05 IDADE      PIC 99.
   05 SALARIO    PIC 9(5)V99.
```

Neste exemplo, o registro EMPREGADO possui três campos: NOME, IDADE e SALARIO, cada um com dimensões e tipos específicos de dados.

Tabelas

As tabelas (ou TABLES) são semelhantes a arrays em outras linguagens e permitem armazenar coleções de elementos do mesmo tipo. A tabela é definida da seguinte forma:

```
01 VENDAS.
   05 VENDA-ITEM OCCURS 10 TIMES.
      10 PRODUTO-NOME PIC A(20).
      10 PRODUTO-VALOR PIC 9(5)V99.
```

Nesse exemplo, a tabela VENDA-ITEM contém 10 elementos, cada um armazenando o nome e o valor de um produto.

Arquivos

COBOL também suporta manipulação de arquivos, que são vitais para o armazenamento permanente de dados. Arquivos podem ser sequenciais, de acesso direto ou de índice. A declaração de um arquivo é realizada na seção de FILE, e o acesso a dados é feito através de registros.

```
SELECT CLIENTES ASSIGN TO UTS-CLIENTES.
DATA RECORD.
01 CLIENTE-REGISTRO.
   05 ID-COMPRADOR PIC 9(5).
   05 NOME-COMPRADOR PIC A(40).
```

Controle de Fluxo

O controle de fluxo em COBOL é responsável pela definição do caminho que a execução do programa deve seguir, baseado em condições e laços. As principais estruturas de controle em COBOL incluem condições (IF, EVALUATE) e loops (PERFORM, WITH, UNTIL).

Condições

A estrutura IF permite a execução condicional de blocos de código. Sua sintaxe básica é:

```
IF CONDICAO
   EXECUTE-ACAO
ELSE
   EXECUTE-ACAO-ALTERNATIVA
END-IF.
```

Por exemplo:

```
IF IDADE > 18
   DISPLAY 'Maior de idade'.
ELSE
   DISPLAY 'Menor de idade'.
END-IF.
```

A estrutura EVALUATE é útil para tratar múltiplas condições, funcionando como um switch-case em outras linguagens:

```
EVALUATE TRUE
   WHEN CONDICAO1
      EXECUTE-ACAO1
   WHEN CONDICAO2
      EXECUTE-ACAO2
   WHEN OTHER
      EXECUTE-ACAO-DEFAULT
END-EVALUATE.
```

Loops

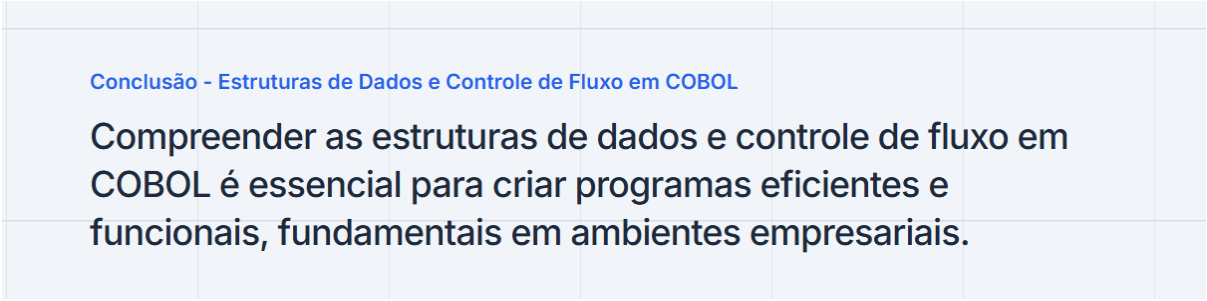
Os loops permitem a repetição de um bloco de código. A instrução PERFORM é utilizada para isso:

```
PERFORM VARYING CONTADOR FROM 1 BY 1 UNTIL CONTADOR > 10
   DISPLAY CONTADOR
END-PERFORM.
```

Neste exemplo, a variável CONTADOR irá assumir o valor de 1 a 10, com sua execução mantendo a condição de parar após este valor.

Considerações Finais

A compreensão das estruturas de dados e do controle de fluxo é essencial para qualquer programador COBOL. A manipulação correta desses elementos permite que programas sejam mais eficientes e flexíveis, atendendo às complexas necessidades de processamento de dados em ambientes empresariais. O COBOL, com suas características robustas e características de processamento de grandes volumes de dados, continua sendo uma linguagem relevante em sistemas legados e na indústria financeira.



Desenvolvimento de Aplicações Comerciais com COBOL e Integração com Bancos de Dados

Introdução ao COBOL para Aplicações Comerciais

O COBOL (Common Business Oriented Language) é uma linguagem de programação desenvolvida nos anos 60, voltada principalmente para aplicações comerciais e de processamento de dados. Sua sintaxe é legível e próxima do inglês, o que facilita a sua compreensão e manutenção, especialmente em ambientes empresariais. A robustez e a eficiência do COBOL o tornaram uma escolha popular para aplicações que exigem processamento intensivo de dados, como sistemas bancários, de folha de pagamento e de gestão empresarial.

Estrutura Básica de um Programa COBOL

Um programa COBOL é estruturado em divisões e seções que definem diferentes componentes e funcionalidades. As quatro divisões principais são:

1. **Identification Division**: Contém informações de identificação do programa.
2. **Environment Division**: Define o ambiente onde o programa será executado.
3. **Data Division**: Declara as variáveis e estrutura de dados que o programa utilizará.
4. **Procedure Division**: Contém a lógica do programa, incluindo instruções para processamento dos dados.

Essa divisão permite uma organização clara do código, facilitando a leitura e manutenção em ambientes corporativos.

Integração com Bancos de Dados

A integração do COBOL com bancos de dados é uma etapa crucial para o desenvolvimento de aplicações comerciais eficazes, pois a maioria das aplicações empresariais requer armazenamento e recuperação de dados. No ambiente COBOL, a combinação com bancos de dados como DB2, IDMS, e VSAM é comum.

Tipos de Acesso a Dados

Existem diferentes métodos para acessar dados em COBOL, incluindo:

- **Acesso Direto**: Utiliza arquivos de disco, permitindo o acesso a registros específicos sem a necessidade de leitura sequencial.
- **Acesso por meio de Banco de Dados**: Permite a comunicação com sistemas de gerenciamento de banco de dados (SGBD). Modelos como a linguagem SQL são frequentemente utilizados em conjunto com COBOL para realizar consultas e manipulações em dados armazenados.

Estrutura de Comandos para Acesso a Dados

Quando se utiliza o COBOL em conjunto com bancos de dados, comandos SQL são empregados dentro da lógica do programa. Esses comandos podem incluir operações de SELECT, INSERT, UPDATE, e DELETE. A interação entre COBOL e SQL pode ser realizada através de pré-processadores que traduzem as instruções SQL em chamadas COBOL equivalentes.

```
IDENTIFICATION DIVISION.
PROGRAM-ID. ExemploAcesso.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
01  Nome-Cliente      PIC A(30).
01  SQL-CODE          PIC S9(4) COMP.
PROCEDURE DIVISION.
    EXEC SQL
        SELECT nome
        INTO :Nome-Cliente
        FROM clientes
        WHERE id_cliente = 1
    END-EXEC.
    IF SQL-CODE = 0
        DISPLAY 'Nome do Cliente: ' Nome-Cliente
    ELSE
        DISPLAY 'Erro ao acessar dados'.
    END-IF.
    STOP RUN.
```

Tratamento de Erros e Transações

Em aplicações comerciais, o tratamento de erros e a gestão de transações são essenciais. O COBOL suporta controle de transações através de comandos SQL, garantindo que as operações sejam atomicamente executadas. É importante implementar mecanismos para reverter transações em caso de falhas, geralmente utilizando comandos como ROLLBACK.

Práticas de Desenvolvimento e Manutenção

Um aspecto crítico do desenvolvimento de aplicações comerciais em COBOL envolve a documentação e a manutenção constante do código. Devido à natureza dos sistemas empresariais, que frequentemente precisam de atualizações e melhorias, uma prática recomendada é a utilização de comentários e documentação clara nas partes do código que lidam com a lógica de negócios e acesso a dados.

Ferramentas de Desenvolvimento

Diversas ferramentas são disponíveis para auxiliar no desenvolvimento e na integração com bancos de dados, como:

- **Compiladores COBOL**: Ferramentas que transformam o código COBOL em executáveis.
- **Ambientes de Desenvolvimento Integrados (IDEs)**: Como o Eclipse ou VSCODE com plugins específicos para COBOL, que ajudam na edição, depuração e gestão de projetos.
- **Drivers ODBC/JDBC**: Para facilitar a comunicação entre COBOL e bancos de dados relacionais.

Conclusão - Desenvolvimento de Aplicações Comerciais com COBOL e Integração com Bancos de Dados

O desenvolvimento de aplicações comerciais com COBOL evidencia a capacidade dessa linguagem em atender as necessidades do mercado, permitindo a criação de soluções robustas.

Exercícios Práticos

Vamos colocar os seus conhecimentos em prática

04 | Exercícios Práticos

Nesta seção, colocaremos a teoria em prática por meio de atividades práticas. Confira cada exercício e onde desenvolver habilidades práticas que o ajudarão a ter sucesso na disciplina.

1) Pesquisa sobre a História do COBOL

Pesquise e escreva um resumo sobre a história do COBOL, desde sua criação até os dias atuais. Inclua informações sobre as versões principais da linguagem e as mudanças significativas que ocorreram ao longo do tempo.

2) Implementação de uma Estrutura de Dados Simples

Crie um programa COBOL que utilize pelo menos duas estruturas de dados diferentes (ex: tabelas, registros) e que implemente controle de fluxo (como IF e PERFORM) para manipular e apresentar dados de uma lista de produtos com suas respectivas quantidades e preços.

3) Desenvolvimento de um Sistema de Gerenciamento de Estoque

Desenvolva um sistema simples de gerenciamento de estoque utilizando COBOL. O sistema deve permitir cadastrar novos produtos, listar os produtos existentes e atualizar a quantidade de um produto específico. Implemente a integração com um banco de dados para armazenar as informações dos produtos.