Gabriel Campos,Luigi Soares,Vinicius Carvalho,Matheus Possas

Linguagem de Programação COBOL

Belo Horizonte - Brasil

2018

Gabriel Campos,Luigi Soares,Vinicius Carvalho,Matheus Possas

Linguagem de Programação COBOL

Trabalho Teórico Prático apresentado na disciplina de Linguagens de Programação do curso de Ciência da Computação da Pontifícia Universidade Católica de Minas Gerais sobre a linguagem de programação COBOL.
.

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

PUC - MG

Trabalho de graduação

Belo Horizonte - Brasil

2018

Lista de ilustrações

Sumário

1 Introdução

COBOL (acrônimo de “COmmon Business-Oriented Language” – Linguagem comum orientada a negócios) é uma das linguagens de programação mais antigas, pertencendo à segunda geração das linguagens de programação. Tem como objetivo principal a criação de sistemas comerciais, financeiros e administrativos para empresas e governos. Foi criado em 1959, onde em um comitê foi proposta essa nova linguagem para fins comerciais, seu estilo de tipagem é forte e estático, possui um paradigma procedural e orientado a objeto. As especificações foram inspiradas em grande parte pela linguagem FLOW-MATIC inventada por Grace Hopper, e pela linguagem COMTRAN da IBM inventada por Bob Bemer. O COBOL influenciou algumas linguagens como a PL/I que herdou a manipulação de arquivos de COBOL, o ABAP possui uma sintaxe parecida com a do COBOL. Para conseguir uma abordagem na web foi criado em 1999 o CobolScript.

2 Histórico

2.1 Historia

A cinquenta anos atrás, cada fabricante de computadores utilizava sua própria linguagem de programação para controlar os computadores e criar programas. Em 1959, um grupo de programadores planejaram o COBOL, sigla para *"COmmon Business-Oriented Language"* que se traduz para "linguagem comum orientada a negócios". Programas escritos em COBOL poderiam ser executados em computadores de diferentes fabricantes. Em 1960, foi feito um teste, e o mesmo programa em COBOL foi executado com sucesso em dois computadores construídos por dois fabricantes distintos.

Um pequeno grupo de programadores da indústria de computadores e seus clientes se juntaram para criar a nova linguagem COBOL. Ambas corporações e agências do governo necessitavam de manter seus salários, preparar economias para o futuro e rastrear de forma confiável, que são aspectos tradicionais da área de processamento de dados empresariais. O Departamento de Defesa estava especialmente interessado nessa linguagem pois era uma das poucas corporações que comprava computadores de fabricantes diferentes. COBOL foi adotado rapidamente dentro dessa e outras agências federais e na indústria particular também. Linguagens comuns adicionais, como ALGOL e versão antigas de FORTRAN, foram desenvolvidas para o uso de cientistas e engenheiros.

Escrito inicialmente para ser usado em curto prazo, COBOL se provou tão útil que dominou o processamento de dados governamentais e de negócios durante décadas. Milhares de transações bancárias ainda são processadas atualmente utilizando programas escritos em COBOL. À medida que o uso de linguagens de programação comuns tornou-se padrão, surgiu uma crescente indústria de software independente.

A ideia de criar uma nova linguagem surgiu com Mary Hawes, um programador da Corporação Burroughs. Ele se juntou com outros usuários de computador e fabricantes para discutir a criação dessa linguagem. Após pequenas reuniões, fizeram um pedido para Charles Phillips, do Departamento de Defesa dos E.U.A., buscando sua colaboração para patrocinar uma conferência formal com a comunidade mais ampla de computadores. O pedido foi aceito. Em Maio de 1959, por volta de 40 representantes de usuários e fabricantes de computadores se encontraram no Pentágono. Durante o evento foi formado, pelo governo americano, um comitê que iniciaram os fundamentos do planejamento da linguagem. Em uma reunião de planejamento feita em New York e Boston, a nova linguagem foi nomeada COBOL.

Durante 1960, equipes da Philadelphia utilizando o Univac, fabricado por Reming-

ton Rand, e outra em Cherry Hill, New Jersey, utilizando o RCA 501 Systems Center trabalharam ininterruptamente para fazer o COBOL funcionar, escreveram compiladores de COBOL e executaram testes. Um teste, executado em Dezembro de 1960, utilizando o mesmo programa em COBOL foi executado com sucesso em dois computadores construídos por dois fabricantes distintos.

As equipes do projeto de desenvolvimento do COBOL incluíam pessoas de diversas culturas, A equipe do Remington Rand UNIVAC utilizavam dois computadores que eram do tamanho de um quarto, o UNIVAC I e UNIVAC II. Grace Hopper(mostrada à direita do centro da Figura 1), era a líder do grupo. Os programas eram feitos em fita magnética, utilizando uma máquina de escrever especial chamada Unitype(Figura 2).

Figura 1 – Grace Hopper e sua equipe

Como a linguagem era amplamente utilizada em 1959, existiam muitas versões diferentes, e a maioria das vezes, incompatíveis de COBOL. Para tentar resolver o problema de incompatibilidade de versões, o American National Standards Institute (ANSI) desenvolveu um padrão de formato para a linguagem em 1968. Essa versão ficou conhecida como *American National Standard* (ANS) COBOL. Em 1974, a ANSI publicou uma versão revisada da (ANS) COBOL, contendo um grande número de funcionalidades que não

Figura 2 – Programadores na maquina de escrever Unitype

estavam incluídas na versão de 1968, chamado COBOL-74. Em 1985, ANSI publicou outra revisão que acrescentou mais funcionalidades na versão padrão de 1974, chamado COBOL-85. COBOL orientado a objeto, lançado em 2002, é uma sub linguagem vinda a partir do COBOL 97, que é a quarta edição da evolução contínua do padrão ANSI/ISO COBOL. O COBOL 97 inclui melhorias convencionais assim como funcionalidades de orientação por objetos.

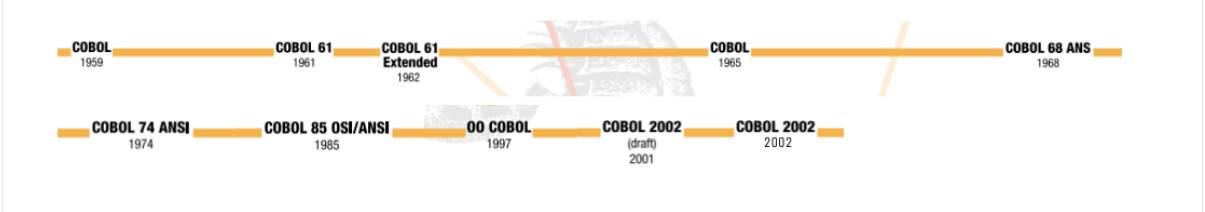

Figura 3 – Linha do tempo de versões da linguagem COBOL

O COBOL foi adotado rapidamente dentro do Departamento de Defesa dos E.U.A., outras agências federais e na indústria particular também. Linguagens comuns adicionais,

como ALGOL e versão antigas de FORTRAN, foram desenvolvidas para o uso de cientistas e engenheiros. Escrito inicialmente para ser usado em curto prazo, COBOL se provou tão útil que dominou o processamento de dados governamentais e de negócios durante décadas. Milhares de transações bancárias ainda são processadas atualmente utilizando programas escritos em COBOL. À medida que o uso de linguagens de programação comuns tornou-se padrão, surgiu uma crescente indústria de software independente.

Atualmente, o COBOL ainda é bastante utilizada no mercado. Em uma pesquisa feita pela Lero, centro de pesquisa de engenharia de software, foi anunciado que são feitas 200 vezes mais transações em COBOL do que pesquisas no google. Isso ocorre pelo fato desses sistemas serem bastante antigos, apesar de bastante relevantes hoje em dia, principalmente na área financeira. E são cruciais, então não podem ocorrer erros pois eles causariam muitos danos, e qualquer tentativa de reescrita de programa seria muito custosa para ser feita, por isso a maioria das empresas que têm software em COBOL escolhem mantê-lo, e contratar novos programadores para isso.

2.2 Linguagens Similares

FORTRAN Como FORTRAN, o COBOL requer uma forma de programação na qual as expressões devem ser enumeradas, Porém, diferente do FORTRAN o COBOL deve ter cada uma de suas sentenças enumeradas sequencialmente O Procedure Division descreve o programa a ser executado, maneira similar ao FORTRAN

ALGOL O Procedure Division descreve o programa a ser executado,maneira similar ao ALGOL O COBOL pode ter sua expressão terminada tanto por ponto(.) como por vírgula(,) e ponto e vírgula(;),enquanto o ALGOL requer apenas o ponto e vírgula.

PL/I O COBOL permite as palavras ser conectadas com hífens, assim como no PL/I, ao qual as palavras são conectadas para formar palavras compostas. O COBOL pode ter sua expressão terminada tanto por ponto(.) como por vírgula(,) e ponto e vírgula(;),enquanto o PL/I requer apenas o ponto e vírgula. Tanto em COBOL quanto em PL/I, as palavras chaves devem ser pré definidas.

3 Paradigma

Procedural: O termo programação procedural é às vezes utilizado como sinônimo de programação imperativa, mas pode se referir a um paradigma de programação baseado no conceito de chamadas a procedimento.

Orientação a Objeto: A programação orientada a objetos é um paradigma de programação em que objetos manipulam os dados de entrada para a obtenção de dados de saída específicos, onde cada objeto oferece uma funcionalidade especial.

4 Características

O Cobol é uma linguagem antiga e simples, que não possui alguns recursos suportados pelas linguagens mais atuais, como variáveis locais, recursividade e alocação dinâmica de memória.

É geralmente a linguagem escolhida em cálculos financeiros, por suportar aritmética inteira aplicada a números muito grandes (milhões, bilhões, etc.) ao mesmo tempo que é capaz de lidar com números muito pequenos como frações de centavos. Outra característica é a formatação, classificação e geração de relatórios.

4.1 Estrutura Básica

A estrutura básica de um programa em Cobol consiste em quatro divisões:

Identification division: Possui os metadados sobre o programa: nome do programa, nome do autor, data em que o programa foi escrito, comentários, etc.

Environment division: Realiza a interface do programa com o ambiente, descrevendo o computador, periféricos e unidade de armazenamento físico.

Data division: Descreve os arquivos de entrada e saída. É a seção onde são declaradas as variáveis, constantes e qualquer tipo de dados que irão precisar alocar memória ao longo da execução do programa.

Procedure division: Contém a descrição do algoritmo do programa, que irá manipular os dados declarados na data division. Possui uma estrutura dividida em seções, parágrafos, sentenças e comandos.

4.2 Váriáveis

O Cobol não possui tipos definidos de variáveis. Todas as variáveis são expressas em "*Picture clause*".

4.2.1 *Picture clause*

É um elemento utilizado para descrever um tipo de dado através de caracteres simples que indicam o tipo de item contido na variável e o seu tamanho. Por exemplo, podemos declarar uma variável como "PIC 999", indicando que a variável em questão irá conter um valor de 3 dígitos, ou "PIC X(8)", indicando que a variável receberá um string de até 8 caracteres.

4.2.2 Declaração de variáveis

As variáveis são declaradas da seguinte forma: <número de nível> <nome da variável> <*picture clause*>. Os números de nível possibilitam hierarquizar as variáveis. Eles vão de 01 a 49, sendo que 77 e 89 são níveis especiais que permitem um uso específico.

01 TELEFONE PIC 99999999.

O nível 88 é utilizado para especificar os chamados "*condition-names*" , cuja função é mostrar e especificar um valor ou uma faixa de valores para determinada variável.

88 MES VALUE 1 THRU 12.

4.3 Sintaxe

O COBOL possui uma sintaxe simples, baseada em palavras que executam ações específicas. São os denominados "verbos" em COBOL. O algoritmo propriamente dito se encontra na Procedure division , que é dividida em:

4.3.1 Seções

São identificadas pela palavra SECTION . Possuem um ou mais parágrafos

4.3.2 Parágrafos

São identificados pela palavra PARAGRAPH . Possuem uma ou mais sentenças.

4.3.3 Sentenças

É um trecho de código que contém um ou mais comandos. Não é terminada por uma quebra de linha, apenas por um ponto final.

4.3.4 Comandos

Contém um verbo COBOL, um sujeito e um nome de objeto opcional.

"MOVE A TO B" , "PERFORM <seção>" , "GO TO <parágrafo>" .

4.4 Estruturas

Aqui serão apresentada algumas estruturas básicas da programação em COBOL como: IF/ELSE, EVALUATE WHEN e PERFORM UNTIL.

4.4.1 "IF/ELSE"

```
IF <condição>
 <sentença>.
ELSE
 <sentença>
END IF.
```

Figura 4 – IF/ELSE

4.4.2 "EVALUATE WHEN"

Equivale ao comando "switch".

```
EVALUATE <variável>
WHEN <condição 1> <sentença>.
WHEN <condição 2> <sentença>.
END-EVALUATE.
```

Figura 5 – Evaluate

4.4.3 "PERFORM UNTIL"

É a principal estrutura de repetição do COBOL, equivale ao comando "for" .

```
PERFORM VARYING <variável> FROM <valor1> TO <valor2> UNTIL <condição>.
```

Figura 6 – Perform

4.4.4 "Matriz unidimensional"

```
WS-TABLE.
    WS-A PIC A(10) VALUE 'ARRAY' OCCURS 5 TIMES.
```

Figura 7 – Matriz unidimensional

4.4.5 "Chamada de função"

```
PERFORM <LabelInicial> THRU <LabelFinal>
```

Figura 8 – Chamada de função

4.4.6 "Exceção"

```
1  OPEN INQUIRY <nome-banco> *abrir banco read-only(INQUIRY) para editar usar (UPDATE)
2      ON EXCEPTION
3          CALL SYSTEM <nome-exception>.
```

Figura 9 – Exceção

4.5 Sobre o COBOL

Algumas características do COBOL que justificam a sua utilização ainda nos dias de hoje são:

Segurança: O COBOL possui normas de segurança bem específicas. Por ser utilizado em instituições financeiras, desenvolvedores estão trabalhando há décadas em atualizações que consequentemente, o tornaram referência em segurança de dados e informações sigilosas. Além disso, ele roda apenas em mainframes , computadores menos expostos à web.

Performance: É possível processar grandes volumes de dados com extrema rapidez, principalmente as rotinas Batch, aumentando de forma considerável a capacidade de processamento durante a janela reservada para o ciclo.

Custos: Na maioria das vezes, os custos associados às aplicações são gastos com as plataformas em que estes sistemas são executados, como mainframes. Estratégias de migração de plataforma e integração de sistemas sem abandonar a linguagem tradicional tem se mostrado uma estratégia mais eficiente para inovar sem ter um gasto elevado.

Adaptabilidade: Com o passar dos anos, o COBOL se adaptou a outras plataformas de hardware. É possível reutilizar aplicações muito antigas ou as utilizar integradas à novas plataformas como .Net, possuindo também boa compatibilidade com tecnologias como cloud e mobile .

Dependência operacional: Empresas tradicionais como bancos, seguradoras e redes de varejo construíram seus sistemas em COBOL. Esses legados são parte do núcleo dessas companhias e seu negócio depende dessas aplicações.

5 Exemplos

5.1 Hello world!

Exemplo mais basico das linguagens de programação,o "print"de um Hello world na tela.

```
1  IDENTIFICATION DIVISION.
2  PROGRAM-ID. HELLO-WORLD.
3  PROCEDURE DIVISION.
4      DISPLAY 'Hello world!'.
5      STOP RUN.
6
```

Figura 10 – Hello world

5.2 Fibonacci

Exemplo de apresentação da sequencia de Fibonacci em COBOL.

```
1  IDENTIFICATION DIVISION.
2      PROGRAM-ID. "Fibonacci".
3  ENVIRONMENT DIVISION.
4  DATA DIVISION.
5  WORKING-STORAGE SECTION.
6  01  fim                    BINARY-C-LONG VALUE 0.
7  01  primeiro               BINARY-C-LONG VALUE 0.
8  01  segundo                BINARY-C-LONG VALUE 1.
9  01  temp                   BINARY-C-LONG VALUE 1.
10 01  print                  PIC Z(19)9.
11 PROCEDURE DIVISION.
12
13 START-PROGRAM.
14     MOVE primeiro TO print.
15     DISPLAY print.
16     MOVE segundo TO print.
17     DISPLAY print.
18     PERFORM VARYING fim FROM 1 BY 1 UNTIL fim = 10
19         ADD primeiro TO segundo GIVING temp
20         MOVE segundo TO primeiro
21         MOVE temp TO segundo
22         MOVE temp TO print
23         DISPLAY print
24     END-PERFORM.
25     STOP RUN.
26
```

Figura 11 – Fibonacci

5.3 Leitura do banco para arquivo

Exemplo de Leitura do banco para arquivo em COBOL Unisys.

```
001000 IDENTIFICATION DIVISION.
001500
001800 ENVIRONMENT DIVISION.
001900
002000 CONFIGURATION SECTION.
002100
002200 SPECIAL-NAMES.
002300   DECIMAL-POINT IS COMMA.
002400 INPUT-OUTPUT SECTION.
002500 FILE-CONTROL.
002600     SELECT DADOSMUT ASSIGN TO DISK.
002800 DATA DIVISION.
002900 FILE SECTION.
003000
003100 FD DADOSMUT.
003200   01 REG-DADOSMUT.
003300      03 S-CON-IDENTIFICACAO                    PIC 9(12).
003400      03 S-CON-DIGITO                           PIC 9(01).
003500      03 S-MUT-ORDEM                            PIC 9(02).
003600      03 S-MUT-SEQUENCIA                        PIC 9(01).
003700      03 S-DAT-COMP                             PIC 9(08).
003800      03 S-FIN-DATA-COMPET REDEFINES S-DAT-COMP.
003900         05 S-FIN-SEC-COMPET                    PIC 99.
004000         05 S-FIN-ANO-COMPET                    PIC 99.
004100         05 S-FIN-MEE-COMPET                    PIC 99.
004200         05 S-FIN-DIA-COMPET                    PIC 99.
004300      03 S-FIN-SEQUENCIA                        PIC 9(12).
004400      03 S-MUT-NOME                             PIC X(36).
004500      03 S-MUT-SEXO                             PIC 9(01).
004600      03 S-MUT-ESTADO-CIVIL                     PIC 9(01).
004700      03 S-MUT-CPF-CGC                          PIC X(14).
004800      03 S-MUT-CONTA-COR                        PIC 9(12).
004900      03 S-MUT-NUM-RG                           PIC X(14).
005000      03 S-MUT-ORGAO-RG                         PIC X(05).
005100      03 S-MUT-UF-RG                            PIC X(02).
005200      03 S-MUT-NUM-PIS                          PIC 9(12).
005300      03 S-MUT-RENDA                            PIC S9(14)V99.
005400      03 S-MUT-ENCARGO-MAX                      PIC S9(05)V99.
005500      03 S-DATA-NASC                            PIC 9(08).
005600      03 S-MUT-DATA-NASC REDEFINES S-DATA-NASC.
005700         05 S-MUT-SEC-NASC                      PIC 99.
005800         05 S-MUT-ANO-NASC                      PIC 99.
005900         05 S-MUT-MES-NASC                      PIC 99.
```

```
44  006000          05 S-MUT-DIA-NASC                       PIC 99.
45  006100      03 S-MUT-QTE-COOBRIG                        PIC 9(01).
46  006200      03 S-TEL-RES                                PIC 9(12).
47  006300      03 S-MUT-TELEFONE-RES REDEFINES S-TEL-RES.
48  006400           05 S-MUT-DDD-TEL-RES                   PIC 9(04).
49  006500           05 S-MUT-NUM-TEL-RES                   PIC 9(08).
50  006600      03 S-TEL-COM                                PIC 9(16).
51  006700      03 S-MUT-TEL-COM REDEFINES S-TEL-COM.
52  006800           05 S-MUT-DDD-COM                       PIC 9(04).
53  006900           05 S-MUT-NUM-COM                       PIC 9(08).
54  007000           05 S-MUT-RAMAL-COM                     PIC 9(04).
55  007100      03 S-MUT-EMAIL                              PIC X(60).
56  007200      03 S-MUT-DDD-CEL                            PIC 9(04).
57  007300      03 S-MUT-NUM-CEL                            PIC 9(08).
58  007400      03 S-MUT-CART-PROF                          PIC X(14).
59  007500      03 S-MUT-SERIE-CT                           PIC X(06).
60  007600      03 S-MUT-UF-CT                              PIC X(02).
61  007620  DATA-BASE SECTION.
62  007640
63  007660  DB BDSIACI-TOTAL OF BDSIACI.
64  007680  01 MUTUARIOS.
65  007700
66  008700 WORKING-STORAGE SECTION.
67  008800 01 CONTADOR      PIC 99 VALUE ZEROS.
68  009300 01 LETRA         PIC  X(01).
69  009400 PROCEDURE DIVISION.
70  009450
71  009500 MAIN.
72  009520   PERFORM ABRIR-ARQUIVOS THRU F-ABRIR-ARQUIVOS
73  009540   PERFORM PROCESSA THRU F-PROCESSA
74  009560   PERFORM FECHAR-ARQUIVOS THRU F-FECHAR-ARQUIVOS
75  009580   DISPLAY "FIM DE PROCESSAMENTO"
76  009600   STOP RUN.
77  009650
78  009700 ABRIR-ARQUIVOS.
79  009800  OPEN OUTPUT DADOSMUT
80  009900  OPEN INQUIRY BDSIACI-TOTAL
81  010000       ON EXCEPTION
82  010100          CALL SYSTEM DMTERMINATE.
83  010400 F-ABRIR-ARQUIVOS. EXIT.
84  010450
85  010500 PROCESSA.
86  010600   SET MUT-POR-NOME TO BEGINNING
87  010650   MOVE "A" TO LETRA
```

```
010700   MOVE 1 TO CONTADOR.
010800 LE-REGISTROS.
010900   FIND NEXT MUT-POR-NOME
011200         ON EXCEPTION
011220            IF DMSTATUS(NOTFOUND)
011240                GO TO F-PROCESSA
011260            ELSE
011300                CALL SYSTEM DMTERMINATE
011400            END-IF
011500   END-FIND
011525   IF MUT-NOME(1:1) = SPACES
011530      GO TO LE-REGISTROS
011535   END-IF
011540
011560   IF MUT-NOME(1:1) NOT EQUAL LETRA
011580      MOVE  MUT-NOME(1:1) TO LETRA
011600      MOVE 1 TO CONTADOR
011620   END-IF
011640   IF CONTADOR < 4
011660      ADD 1 TO CONTADOR
011662      MOVE CON-IDENTIFICACAO TO   S-CON-IDENTIFICACAO
011664      MOVE CON-DIGITO        TO   S-CON-DIGITO
011666      MOVE MUT-ORDEM         TO   S-MUT-ORDEM
011668      MOVE MUT-SEQUENCIA     TO   S-MUT-SEQUENCIA
011670      MOVE FIN-SEC-COMPET    TO   S-FIN-SEC-COMPET
011672      MOVE FIN-ANO-COMPET    TO   S-FIN-ANO-COMPET
011674      MOVE FIN-MES-COMPET    TO   S-FIN-MEE-COMPET
011676      MOVE FIN-DIA-COMPET    TO   S-FIN-DIA-COMPET
011680      MOVE MUT-NOME          TO   S-MUT-NOME
011682      MOVE FIN-SEQUENCIA     TO   S-FIN-SEQUENCIA
011684      MOVE MUT-SEXO          TO   S-MUT-SEXO
011686      MOVE MUT-ESTADO-CIVIL  TO   S-MUT-ESTADO-CIVIL
011688      MOVE MUT-CPF-CGC       TO   S-MUT-CPF-CGC
011690      MOVE MUT-CONTA-COR     TO   S-MUT-CONTA-COR
011692      MOVE MUT-NUM-RG        TO   S-MUT-NUM-RG
011694      MOVE MUT-ORGAO-RG      TO   S-MUT-ORGAO-RG
011696      MOVE MUT-UF-RG         TO   S-MUT-UF-RG
011698      MOVE MUT-NUM-PIS       TO   S-MUT-NUM-PIS
011700      MOVE MUT-RENDA         TO   S-MUT-RENDA
011702      MOVE MUT-ENCARGO-MAX   TO   S-MUT-ENCARGO-MAX
011704      MOVE MUT-SEC-NASC      TO   S-MUT-SEC-NASC
011706      MOVE MUT-ANO-NASC      TO   S-MUT-ANO-NASC
011708      MOVE MUT-MES-NASC      TO   S-MUT-MES-NASC
011710      MOVE MUT-DIA-NASC      TO   S-MUT-DIA-NASC
011712      MOVE MUT-QTE-COOBRIG   TO   S-MUT-QTE-COOBRIG

011714      MOVE ZEROS             TO   S-MUT-DDD-TEL-RES
011716      MOVE ZEROS             TO   S-MUT-NUM-TEL-RES
011718      MOVE ZEROS             TO   S-MUT-NUM-COM
011720      MOVE ZEROS             TO   S-MUT-RAMAL-COM
011722      MOVE MUT-EMAIL         TO   S-MUT-EMAIL
011724      MOVE ZEROS             TO   S-MUT-DDD-CEL
011726      MOVE ZEROS             TO   S-MUT-NUM-CEL
011728      MOVE MUT-CART-PROF     TO   S-MUT-CART-PROF
011730      MOVE MUT-SERIE-CT      TO   S-MUT-SERIE-CT
011732      MOVE MUT-UF-CT         TO   S-MUT-UF-CT
011734      WRITE REG-DADOSMUT
011736      GO TO LE-REGISTROS
011740   END-IF
011760
012060   GO TO LE-REGISTROS.
012062
012064
012066 F-PROCESSA. EXIT.
012068
012070 FECHAR-ARQUIVOS.
012100   CLOSE DADOSMUT SAVE
012200   CLOSE BDSIACI-TOTAL.
012300 F-FECHAR-ARQUIVOS. EXIT.
```

Figura 12 – Leitura do banco para arquivo

6 Considerações Finais

COBOL é uma linguagem de programação antiga com paradigma procedural, sendo ela uma linguagem forte e estática, além de orientada a objeto, ela se destacou nos setores financeiros, administrativos e em empresas, pois o COBOL tem campos de comprimento fixo, decimais aritméticos, Segurança no código, possui boa performace, baixo custo associado às aplicações e boa compatibilidade com as novas tecnologias. Características marcante nesses setores do mercado, pois as mesmas auxiliam na velocidade da resolução de problemas relacionados a aritmética e possuem boa segurança.

7 Referências Bibliográficas

F.P Mathur, A Brief Description and Comparison of Programming Languages FORTRAN, ALGOL, COBOL, PL/I and LISP 1.5 From a Critical Standpoint,p.1 - 20, NASA, California Instute of Tecnology, 1972

Wikipedia, COBOL. Disponível em: <https://pt.wikipedia.org/wiki/COBOL>. Acesso em 28 de março de 2018.

Jean Sammet, The Early History of COBOL, p.121–161, ACM SIGPLAN Notices, Association for Computing Machinery, Inc., 1978

Garfunkel, Jerome (1987). The Cobol 85 Example Book. New York: Wiley.

Wexelblat, Richard (1981). History of Programming Languages. Boston: Academic Press.

João Araújo,COBOL.2009.13f.Trabalho de Características das Linguagens de Programação - Universidade do Estado do Rio de Janeiro,Brasil,2009.

Wikivesity, Introdução às linguagens de programação/COBOL. Disponível em: <https://pt.wikiversity.org/wiki/Introdução_às_Linguagens_de_Programação/COBOL>.

Behrouz A. Forouzan; Sophia Chung Fegan. Foundations of Computer Science: From Data Manipulation to Theory of Computation . Cengage Learning Editores; 2003. ISBN 978-970-686-285-3. p. 197.

"(...)para escrever um simples programa em Cobol são necessárias muitas linhas de código, o que torna a linguagem não muito fácil de aprender.", Mario Leite. Técnicas de Programação – Uma Abordagem Moderna . Brasport; ISBN 978-85-7452-229-6. p. 191. Stephen R. Schach. Engenharia de Software . McGraw Hill Brasil; ISBN 978-85-63308-44-3. p. 465.

Cinco motivos que mantém o Cobol firme e forte nas empresas. Computerworld, 2016. Disponível em: <http://computerworld.com.br/cinco-motivos-que-mantem-o-cobol-firme-e-forte-nas-e mpresas>. Acesso em: 27 mar. 2018.

The PICTURE Clause. Microfocus. Disponível em: <https://supportline.microfocus.com/docun Acesso em: 04 abril 2018.

Hussain Ezaj,History Of Cobol Programming Language. Disponível em: <http://ejaz007.experts cobol-programming-language>. Acesso em 30 de março de 2018.

The COBOL Programming Language. Disponível em: <http://groups.umd.umich.edu/cis/cours Acesso em 30 de março de 2018.

http://americanhistory.si.edu/cobol/introduction COBOL: Introduction. Disponível em: <http://americanhistory.si.edu/cobol/introduction>. Acesso em 30 de março de 2018.

Reddy Karthik,Exactly what is COBOL and why is COBOL still a widely used language in IT?. Disponível em: <https://freedomafterthesharks.com/2016/06/27/exactly-what-is-cobol-and-why-is-cobol-still-a-widely-used-language-in-it/>. Acesso em 30 de março de 2018.