BIREME / OPS / OMS

Centro Latino Americano e do Caribe de Informação em Ciências dla Saúde

Utilitários CISIS - Manual de Referência

Versão 5.2

São Paulo, SP - 2005-2006

Utilitários CISIS - Manual de Referência

Ficha Catalográfica

BIREME (Brasil)

Utilitários CISIS - Manual de Referência. / BIREME (org.).
São Paulo, SP : BIREME / OPS / OMS, 2005-2006.

211 p.

1. Manual do usuário. 2. Acesso à informação. 3. Sistemas de informação. 4. Gerenciamento de informação. I. BIREME II. Título

Advertência - A menção a companhias e/ou instituições específicas ou a certos produtos não implica que estes sejam apoiados ou recomendados por BIREME / OPAS / OMS, e não significa que haja preferência em relação a outros de natureza similar, citado ou não.

BIREME / OPS / OMS

Centro Latino Americano e do Caribe de Informação em Ciências dla Saúde

Rua Botucatu 862 V. Clementino

Sumário

Abreviaturas utilizadas

- **ANSI**. American National Standards Institute [Instituto Nacional Americano de Normas].
- **ASCII**. American Standard Code for Information Interchange [Código Americano Normalizado para o Intercambio de Informação].
- **BIREME**. Centro Latino-Americano e do Caribe de Informação em Ciências da Saúde.
- **BVS**. Biblioteca Virtual em Saúde.
- **CDS**. Computerized Documentation System [Sistema de Documentação Computarizada].
- **CP**. Code Page [Página de código].
- **FST**. Field Selection Table [Tabela de Seleção de Campo].
- **FTP**. File Transfer Protocol [Protocolo de transferência de arquivos].

- **IFP**. Inverted File Pointer [Ponteiro de arquivo invertido].
- **ISIS**. Integrated Set of Information Systems [Conjumto integrado de sistemas de informação].
- **ISO**. International Organization for Standardization [Organização Internacional para Normalização].
- **LILACS**. Literatura Latino-Americana e do Caribe em Ciências da Saúde.
- **OMS**. Organização Mundial da Saúde.
- **OPS**. Organização Pan-Americana da Saúde.
- **UMESCO**. Umited Nations Educational, Scientific and Cultural Organization [Organização das Nações Unidas para a Educação, a Ciência e a Cultura].

Como usar este manual

Este manual foi concebido como uma referência para o uso dos utilitários CISIS e seu conteúdo principal está dividido em quatro capítulos, sendo:

1. Apresentação: a descrição do CISIS, da interface, instalação, execução e convenções utilizadas pelos utilitários;
2. Utilitário MX: contendo a descrição completa de todos os parâmetros e funções disponíveis;
3. Utilitários do arquivo mestre: descreve a sintaxe e uso dos programas utilitários para arquivos mestres.
4. Utilitários do arquivo invertido: descreve a sintaxe e uso dos programas utilitários para arquivos invertidos.

Há ainda quatro Apêndices contendo informação adicional transversal aos capítulos e também informação sobre a estrutura interna do ISIS.

Um *Glossário* e uma lista de *Abreviaturas utilizadas* complementam o documento.

Prefácio

Sobre BIREME

A BIREME cumpre ano após ano sua missão como centro especializado em informação científica e técnica em saúde para a região da América Latina e Caribe. Estabelecida no Brasil em 1967, com o nome de Biblioteca Regional de Medicina (que originou a sigla BIREME), atendeu desde o princípio à demanda crescente de literatura científica atualizada por parte dos sistemas nacionais de saúde e das comunidades de pesquisadores, profissionais e estudantes. Posteriormente, em 1982, passou a chamar-se Centro Latino-Americano e do Caribe de Informação em Ciências da Saúde para melhor expressar as suas funções orientadas ao fortalecimento e ampliação do fluxo de informação científica e técnica em saúde em toda a região, mas conservou sua sigla.

O trabalho em rede, com base na descentralização, no desenvolvimento de capacidades locais, no compartilhamento de recursos de informação, no desenvolvimento de produtos e serviços cooperativos, na elaboração de metodologias comuns, foi sempre o fundamento do trabalho de cooperação técnica da BIREME. É assim que o centro se consolida como um modelo internacional que privilegia a capacitação dos profissionais de informação em nível gerencial e técnico para a adoção de paradigmas de informação e comunicação que melhor atendam as necessidades locais.

Os principais fundamentos que dão origem e suporte à existência da BIREME são os seguintes:

- acesso à informação científico-técnica em saúde é essencial para o desenvolvimento da saúde;
- a necessidade de desenvolver a capacidade dos países da América Latina e do Caribe de operar as fontes de informação científico-técnica em saúde de forma cooperativa e eficiente;
- a necessidade de promover o uso e de responder às demandas de informação científico-técnica em saúde dos governos, dos sistemas de saúde, das instituições de ensino e investigação.

A BIREME, como centro especializado da Organização Pan-Americana da Saúde (OPAS)/Organização Mundial da Saúde (OMS), coordena e realiza atividades de cooperação técnica em gestão de informação e conhecimento científico com o objetivo de fortalecer e ampliar o fluxo de informação científica em saúde no Brasil e nos demais países da América Latina e Caribe como condição essencial para o desenvolvimento da saúde, incluindo planejamento, gestão, promoção, investigação, educação e atenção.

O convênio que fundamenta a BIREME é renovado a cada cinco anos pelos membros do Comitê Assessor Nacional da instituição (OPAS, Ministério da Saúde do Brasil, Ministério da Educação e Cultura do Brasil, Secretaria de Saúde do Estado de São Paulo e Universidade Federal de São Paulo – Unifesp). Esta última oferece a infra-estrutura física necessária ao estabelecimento da instituição.

Em 2004 a instituição assumiu a responsabilidade de tornar-se uma instituição baseada em conhecimento.

Sobre a BVS

Com o surgimento e consolidação da internet como meio predominante de informação e comunicação, o modelo de cooperação técnica da BIREME evoluiu, a partir de 1998, para a construção e desenvolvimento da Biblioteca Virtual em Saúde (BVS) como espaço comum de convergência do trabalho cooperativo de produtores, intermediários e usuários de informação. A BVS promove o desenvolvimento de uma rede de fontes de informação científica e técnica com

acesso universal na internet. Pela primeira vez abre-se a possibilidade real de acesso eqüitativo à informação em saúde.

A BIREME tem a Biblioteca Virtual em Saúde como modelo para a gestão de informação e conhecimento, o qual envolve a cooperação e convergência de instituições, sistemas, redes e iniciativas de produtores, intermediários e usuários na operação de redes de fontes de informação locais, nacionais, regionais e internacionais privilegiando o acesso aberto e universal.

Hoje todos os países da América Latina e Caribe (Região) participam direta ou indiretamente dos produtos e serviços cooperativos promovidos pela BVS, envolvendo mais de mil instituições em mais de 30 países.

A BVS é simulada em um espaço virtual da internet formada pela coleção ou rede de fontes de informação em saúde da Região. Usuários de diferentes níveis e localização podem interagir e navegar no espaço de uma ou várias fontes de informação, independentemente de sua localização física. As fontes de informação são geradas, atualizadas, armazenadas e operadas na internet por produtores, integradores e intermediários, de modo descentralizado, obedecendo a metodologias comuns para sua integração na BVS.

A BVS organiza a informação em uma estrutura que integra e interconecta bases de dados referenciais, diretórios de especialistas, eventos e instituições, catálogo de recursos de informação disponíveis na internet, coleções de textos completos com destaque para a coleção SciELO (*Scientific Electronic Library Online*) de revistas científicas, serviços de disseminação seletiva de informação, fontes de informação de apoio à educação e a tomada de decisão, notícias, listas de discussão e apoio a comunidades virtuais.

O espaço da BVS constitui, portanto, uma rede dinâmica de fontes de informação descentralizada a partir da qual se pode recuperar e extrair informação e conhecimento para subsidiar os processos de decisão em saúde.

A Biblioteca Virtual em Saúde é visualizada como a base distribuída do conhecimento científico e técnico em saúde registrado, organizado e armazenado em formato eletrônico nos países da Região, acessível de forma universal na internet de modo compatível com as bases internacionais.

Apresentação

CISIS - Interface

MicroISIS (*CDS/ISIS for Mini-microcomputers*) é um software desenvolvido pela UMESCO para bases de dados constituídas principalmente por texto. MicroISIS trata campos (elementos de dados) de tamanho variável. Um campo pode estar ausente em um ou mais registros, pode conter só um elemento de dado, ou dois ou mais subcampos de tamanho variável. Além disto, um campo pode ser repetitivo, isto é, um registro dado pode conter mais de uma ocorrência do campo.

A Interface CISIS é uma biblioteca de funções, escrita em linguagem de programação C, projetada para permitir o desenvolvimento de aplicações para bases de dados MicroISIS (sem chamar o software MicroISIS). As aplicações CISIS são plenamente compatíveis com MicroISIS, incluindo aplicações multiusuárias.

Existem diversas implementações sobre a estrutura CDS/ISIS original, por isso é que hoje em dia é mais apropriado chamar a todas estas variantes de "família Isis". É importante destacar que os dados criados por qualquer variante desta "família" são compatíveis e podem ser intercambiados entre elas.

As aplicações desenvolvidas com a Interface CISIS podem manipular várias bases de dados ao mesmo tempo; o arquivo mestre e o arquivo invertido são processados

de forma independente. Não é necessário ter a definição da base de dados para rodar as aplicações CISIS.

A Interface CISIS e os Programas Utilitários CISIS foram projetados e implementados no Centro de Informação em Ciências da Saúde para América Latina e o Caribe - BIREME, Organização Pan-Americana da Saúde - OPS, e atualmente estão disponíveis para plataformas:

- PC IBM[1] 32 bits
- UNIX baseados em processadores Intel (LINUX, SCO, etc.)
- UNIX baseados em outros processadores (HP-UX, Sum, IBM-AIX, CDC/S4320, etc.)
- VAX sob VMS
- HP3000/950 sob MPE/XL

CISIS - Programas Utilitários

Os Utilitários CISIS (*CISIS Interface Utility Programs*) são um conjunto de programas desenvolvidos em linguagem de programação C que "chamam" as funções oferecidas pela Interface CISIS para realizar distintas funções em bases de dados da família Isis, tais como recuperar e mostrar registros, a manutenção de bases de dados, etc. Além disto, podem efetuar função especiais que permitem ordenar arquivos mestres, gerar tabelas a partir de um arquivo mestre, substituir as etiquetas dos campos, etc.

Este conjunto de programas utilitários é oferecido sob quatro versões: 10/30 e 16/60, LIND, FFI. As diferenças principais são em relação ao tamanho das chaves do arquivo invertido a ao tamanho máximo de registro medido em bytes que suportam, conforme mostrado na tabela seguinte.

[1] Para MS-DOS de 16 bits deverão ser usadas as aplicaçõess de CISIS até à versão 3.4

	10/30	16/60	LIND	FFI
Chaves arquivo invertido	30	60	60	60
Tamanho máximo do registro	32.767	32.767	32.767	1.048.576

Nota: A versão 10/30 é a única compatível com o CDS/ISIS da Umesco

Para mais detalhes sobre a estrutura dos arquivos mestre e invertido veja o Apêndice III: *Estrutura dos registros de uma base ISIS.*

As características particulares destes programas, podem ser verificadas na declaração de versão que se obtém com o comando *what*

Por exemplo

```
mx what
CISIS Interface v5.2a/PC32/M/32767/10/30/I - Utility MX
CISIS Interface v5.2a/.iy0/Z/4GB/GIZ/DEC/ISI/UTL/INVX/B7/FAT/CIP/CGI/MX
Copyright (c)BIREME/PAHO 2006. [http://www.bireme.br/products/cisis]
```

Acrônimo	**Descrição**
V5.2a	Número da versão
PC32	Compilador usado (neste caso Windows PC)
L	Versão Lind se estiver presente
M	Suporte multiusuário
32767	Tamanho máximo do registro em bytes, valor por default
10/30	Tamanho de chaves do arquivo invertido
I	Permite atualizar o I/F
Utility MX	Nome do programa
.iy0	Arquivo físico simples para I/F (realizado por mkiy0)
Z	I/F compactado (realizado por myz – descontinuado)
4GB	Tamanho máximo do arquivo mestre
GIZ	Gizmo
DEC	Decod
ISI	Importação Iso-2709
UTL	Módulo Ciutl
INVX	Pesquisa I/F múltipla
B7	Versão do mecanismo interno de pesquisa
FAT	Fatal()
CIP	Cipar()
GCI	Soporta a operação em ambiente CGI
MX	Cisis_mx()

Utilitário MX

MX

O Programa MX é um utilitário de propósito geral para trabalhar em bases de dados MicroISIS. Pode realizar a maioria das funções da Interface CISIS, incluindo a importação/exportação de arquivos ISO-2709, pesquisas, procedimentos de substituição global de padrões, junção de registros do arquivo mestre por número de registro ou por chave do arquivo invertido, incorporar campos com dados gerados mediante uma Tabela de Seleção de Campos (FST), e funções de inclusão exclusão de campos.

Utilitários para arquivo mestre

MXF0

Analisa todos os registros de um arquivo mestre dado, produzindo informação sobre os campos presentes e dos caracteres usados pelos mesmos.

MXTB

O programa MXTB permite contar o conteúdo dos campos, por exemplo, número de vezes que aparece cada autor, cada descritor, ou a aparição simultânea de um autor e um título de publicação periódica, etc.
O resultado da execução de MXTB é um arquivo mestre que contem um registro para cada frase diferente encontrada (categoria). Estes registros têm campos para armazenar a categoria e sua freqüência.

MXCP

Copia registros a partir de um arquivo mestre de entrada para um arquivo mestre de Saída, possibilitando que o dado de entrada seja modificado pelos procedimentos de substituição global de padrões e/ou procedimentos que suprimem espaços no começo ou no final, caracteres em branco, caracteres não imprimíveis e caracteres de pontuação final.
Também converte em repetitíveis os campos que contenham um delimitador específico e pode descartar campos de entrada, de acordo com os valores de suas tags (*etiquetas*).
Outra característica do MXCP é a de recuperar (*undelete*) registros logicamente apagados do arquivo mestre.

MSRT

Ordena os registros de um arquivo mestre em ordem ascendente, de acordo com chaves gerados aplicando um formato aos registros.

RETAG

Este programa tem duas aplicações:
Alterar as tags (*etiquetas*) dos campos de um arquivo mestre dado, de acordo com uma tabela de renumeração.
Desbloquear (*unlock*) um arquivo mestre.

CTLMFN

Desdobra e atualiza o registro de controle do arquivo mestre.
Deve ser usado quando um arquivo mestre é reinicializado acidentalmente.

MKXRF

É um programa para recuperação do arquivo mestre, que lê um arquivo *.mst* e gera o arquivo *.xrf* correspondente.

Utilitários para arquivo mestre

	Pode ser usado para restaurar todos os registros ativos de um arquivo mestre reinicializado em forma lógica.
I2ID	Lê um arquivo mestre e gera um arquivo ASCII, que pode ser editado e modificado. A idéia é que trabalhe junto com o utilitário *ID2I* que realiza a tarefa inversa: lê um arquivo ASCII e converte os dados lidos em registros de arquivo mestre.
ID2I	Lê um arquivo ASCII gerado por *I2ID* (ou com a mesma estrutura que um arquivo gerado por este) e converte os dados lidos em registros de arquivo mestre.
CRUNCHMF	Converte o arquivo mestre de um sistema operacional para outro, por exemplo de Windows para Linux.

Utilitários para arquivo invertido

IFKEYS	Desdobra os termos do arquivo invertido e a quantidade de *postings* de cada um dos mesmos. Opcionalmente os termos podem ser desmembrados pelas etiquetas dos que foram extraídos.
IFLOAD	Carrega um arquivo invertido a partir dos arquivos de ligações, de acordo com as opções de processamento. Aceita outros formatos, além do formato de arquivo de ligações padrão do MicroISIS.
MYS	Faz um sort do arquivo de ligações (links) para criar o arquivo invertido.
IFMERGE	Combina vários arquivos invertidos de diferentes arquivos mestres em um único arquivo invertido, como um procedimento para recuperar os registros a partir dos arquivos mestres fonte.
MKIY0	Combina os seis arquivos que compõe o arquivo invertido em um único arquivo físico.
CRUNCHIF	Converte o arquivo invertido de um sistema operacional para outro, por exemplo, de Windows para Linux.

Instalação dos utilitários CISIS

Toda a instalação dos Utilitários CISIS consiste em criar um diretório, em geral \CISIS\SYS\, e copiar para o mesmo todos os utilitários.

Por uma questão de comodidade, pode-se colocar o diretório \CISIS\SYS no PATH do sistema operacional, para poder executar os utilitários a partir do ponto em que se encontre, sem ter que referenciar o diretório \CISIS\SYS.

Exemplos do manual

Os exemplos do manual estão baseados na sua maioria na base de dados CDS, e supõe-se que esteja localizada no diretório:

```
\CISIS\BASES\
```

Trabalhar-se-á sobre a base de dados que muitas vezes será alterada, portanto se aconselha fazer uma cópia de segurança da mesma.

Execução dos utilitários

Os programas utilitários CISIS são executados como comandos, a partir do *prompt* do sistema operacional, ou a partir de arquivos *bat* (arquivos de processamento de lote) de MS-DOS ou *scripts* (shell scripts) de UNIX.

Qualquer programa utilitário CISIS pode ser executado escrevendo seu nome e um ou mais parâmetros, supondo que o diretório \cisis\sys (diretório onde se encontram os utilitários CISIS) esteja incluído na lista PATH. Se não se submeter parâmetros na chamada, cada programa utilitário CISIS apresenta uma descrição breve de seu uso. Por exemplo, escrevendo somente o nome MXCP no *prompt* do DOS, apresenta:

```
CISIS Interface v5.2a/PC32/M/32767/10/30/I - Utility MXCP
Copyright (c)BIREME/PAHO 2006. [http://www.bireme.br/products/cisis]

mxcp {in=<file>|<dbin>} [create=]<dbout> [<option> [...]]

options: {from|to|loop|count|tell|offset}=<n>
         gizmo=<dbgiz>[,tag_list>]
         undelete
         clean [mintag=1] [maxtag=9999]
         period=.[,<tag_list>]
```

```
            repeat=%[,<tag_list>]
            log=<filename>

Ex: mxcp in create=out clean period=.,3 repeat=;,7

     in = 3 «   Field 3 occ 1. »
          3 «Field 3 occ 2 . »
          7 « Field 7/1;Field 7/2 ;Field 7/3.»

    out = 3 «Field 3 occ 1»
          3 «Field 3 occ 2»
          7 «Field 7/1»
          7 «Field 7/2»
          7 «Field 7/3.»
```

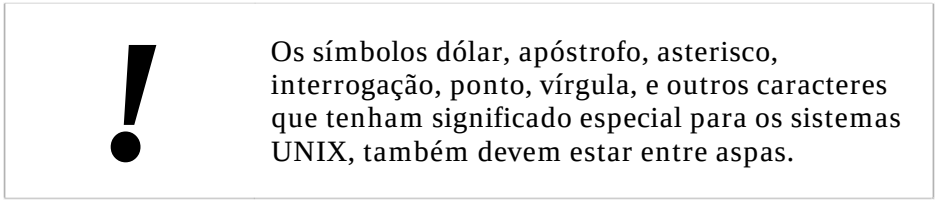
Os parâmetros de chamada são providos como uma lista separados por espaços em branco e, por tanto, cada parâmetro individual deve estar entre aspas quando contiver espaços em branco ou qualquer caráter especial do sistema operacional (tais como sinais de maior e menor, barra vertical, etc.).

O exemplo seguinte executa o programa MX com três parâmetros de chamada (nome da base de dados, expressão de busca e especificação do formato de apresentação):

```
mx \cisis\bases\cds "plants*water" "pft=mfn/,'Ti: 'v24/,(|Au: |v70/)"
```

Para usar aspa dupla como parte de um parâmetro de chamada, deve estar precedida pela barra invertida:

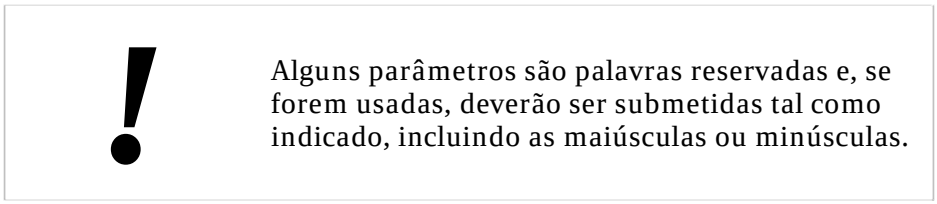
```
mx \cisis\bases\cds "plants*water" "pft=mfn/, \" Ti: \"v24/,(|Au: |v70/)"
```



Convenções de sintaxe

São usadas as seguintes convenções para descrever a sintaxe dos Programas Utilitários CISIS:

<parameter>	Parâmetro obrigatório
[<parameter>]	Parâmetro opcional
{<option 1>|<option 2>}	Deve-se escolher entre <opção 1> ou <opção 2>
<option> [...]	<opção> pode ser repetida



Alguns parâmetros são palavras reservadas e, se forem usadas, deverão ser submetidas tal como indicado, incluindo as maiúsculas ou minúsculas.

Por exemplo, MXCP tem a sintaxe geral:

```
mxcp <dbin> [create=]<dbout> [<option> [...]]

opções:

{from|to|loop|count|tell|offset}=<n>
gizmo=<dbgiz>[,<tag_list>]
undelete
clean [mintag=1] [maxtag=9999]
period=.[,<tag_list>]
repeat=%[,<tag_list>]
log=<filename>
```

Indicando que dois parâmetros são obrigatórios: (a) nome da base de dados de entrada e (b) nome da base de dados de saída.

Então, o comando:

```
mxcp \cisis\bases\cds newcds
```

Copia o arquivo mestre *cds* localizado no diretório \cisis\bases para o arquivo mestre *newcds* situado no diretório atual. O arquivo mestre newcds deverá existir, do contrário o exemplo produzirá um erro.

Se *newcds* não existe pode ser criado mediante o parâmetro opcional *create* como se vê no seguinte exemplo:

```
mxcp \cisis\bases\cds create=newcds
```

As opções de processamento podem ser indicadas, usando os parâmetros opcionais. Para indicar, por exemplo, limites de registros a processar, utiliza-se *from* e *to*

```
mxcp \cisis\bases\cds create=newcds from=10 to=20
```

Utilitário MX

Apresentação

Introdução

MX é um programa de uso geral para bases de dados CDS/ISIS que realiza a maioria das funções da Interface CISIS. Da mesma forma que os outros programas utilitários CISIS, MX é executado a partir da linha de comandos do sistema operacional, indicando as operações a realizar através de parâmetros.

MX pode ser utilizado, por exemplo, para recuperar e mostrar um conjunto de registros de uma base de dados, de acordo com uma expressão de busca e um formato de visualização, como na seguinte linha:

```
mx \cisis\bases\cds "plants * water" "pft=mfn,x1,v24/"
```

Da mesma forma, MX permite realizar buscas em texto livre, mesmo que não exista um arquivo invertido.

MX também pode ler arquivos ISO-2709 ou arquivos ASCII planos, utilizando delimitadores como separadores de campos. Nestes casos os registros de entrada são convertidos para registros de arquivo mestre à medida que são lidos.

Os seguintes procedimentos podem ser aplicados aos registros de entrada:

5. Substituição global de padrões
6. Junção de registros, por número de registro ou por chave de arquivo invertido.
7. Acrescentar campos com os dados gerados por uma Tabela de seleção de campos.
8. Criação e exclusão de registros, especificado através de uma linguagem de formatação.

Os registros processados pelo MX podem ser enviados para um arquivo mestre, um arquivo ISO-2709 ou para a saída padrão (a qual pode ser redirecionada a um arquivo ou impressora). As línhas produzidas por um formato podem ser enviadas ao sistema operacional.

A execução do MX pode gerar uma chamada ao sistema operacional para que este execute determinado programa.

O resultado da aplicação de uma Tabela de Seleção de Campos (FST) a um arquivo mestre pode ser enviado a um arquivo de ligações (*link files*) ou acrescentado a um arquivo invertido.

O arquivo de saída pode ser o mesmo que o de entrada.

MX trabalha também em ambientes multiusuários.

Descrição geral

Para executar o MX é necessário especificar onde estão os dados sobre os quais irá trabalhar. Podem vir de um arquivo mestre, um arquivo ISO-2709 ou um arquivo texto. Este é o único parâmetro obrigatório do programa.

A linha

```
mx \cisis\bases\cds
```

gera uma listagem na tela dos registros da base *cds*, que se encontra no diretório \cisis\bases. Os registros listados são visualizados sem formatação.

Outros parâmetros podem especificar opções de processamento, por exemplo:

```
mx \cisis\bases\cds from=10 to=20
```

apresentando na tela os registros da base de dados *cds* a partir de 10 até 20. A base se encontra no diretório \cisis\bases e os registros são visualizados sem formatação.

A linha de comando

```
mx \cisis\bases\cds from=10 to=20 "pft=mfn,x1,v24(0,7)/"
```

apresenta na tela os registros 10 a 20 da base de dados cds, aplicando-lhes o formato especificado no parâmetro pft=mfn,x1,v24(0,7)/. A base de dados se encontra no diretório \cisis\bases.

É importante ter em conta que a ordem em que se dispõem os parâmetros opcionais **não afeta** a execução do MX. A execução destes parâmetros se fará na ordem em que aparecem na declaração de sintaxe.

Assim a linha anterior poderia ser:

```
mx \cisis\bases\cds   to=20   "pft=mfn,x1,v24(0,7)/"   from=10
```

ou

```
mx \cisis\bases\cds   "pft=mfn,x1,v24(0,7)/"   from=10   to=20
```

Porém, embora as seguintes declarações sejam equivalentes:

```
mx \cisis\bases\cds   pft=@file1  proc=@miproc.prc
mx \cisis\bases\cds   proc=@miproc.prc  pft=@file1

mx \cisis\bases\cds   gizmo=gizfile1  proc=@miproc.prc
mx \cisis\bases\cds   proc=@miproc.prc  gizmo=gizfile1
```

o parâmetro gizmo será aplicado antes que o proc, porque assim é estabelecido na sintaxe.

Se a entrada principal (primeiro parâmetro) é uma base de dados e seu arquivo invertido correspondente está disponível, o conjunto de registros a serem processados podem ser obtidos através de uma busca.

O seguinte exemplo recupera os registros da base *cds*, que se encontra no diretório \cisis\bases, que contenham as palavras plants e water.

```
mx \cisis\bases\cds "plants * water"
```

MX pode ler os dados de entrada a partir de um arquivo no formato ISO-2709 ou um arquivo de texto com delimitadores.

A seguinte linha apresenta os primeiros 5 registros de um arquivo ISO-2709 chamado cds.iso, que se encontra no diretório \cisis\bases.

```
mx iso=\cisis\bases\cds.iso to=5
```

No próximo exemplo o MX utiliza um arquivo ASCII chamado *name* como fonte de entrada, cujo conteúdo é:

```
Autor 1|título 1|^aParis^bUmesco^c1965
|título 2|^aParis^bUmesco^c1965
Autor 3|título 3|^aParis^bUmesco^c1965
```

e pode ser listado mediante a seguinte chamada ao MX:

```
mx seq=name "pft=mfn,c11,v1,c21,v2,c31,v3/" now
```

Que gerará a saída:

```
000001    Autor 1   Title 1   ^aParis^bUmesco^c1965
000002              Title 2   ^aParis^bUmesco^c1965
000003    Autor 3   Title 3   ^aParis^bUmesco^c1965
```

Os registros processados podem ser armazenados em um arquivo mestre. As seguintes linhas criam o arquivo mestre *sample*:

```
mx\cisis\bases\cds "plants * water" create=sample -all now
mx iso=\cisis\bases\cds.iso to=5 create=sample -all now
mx seq=name create=sample -all now
```

Estes registros, também, podem ser exportados para um arquivo no formato ISO-2709, por exemplo, *sample.iso:*

```
mx \cisis\bases\cds "plants * water" iso=sample.iso -all now
mx iso=\cisis\bases\cds.iso to=5 iso=sample.iso -all now
mx seq=name iso=sample.iso -all now
```

Quando MX realiza um ou mais processos que modificam registros (tanto se forem lidos de uma base de dados, um arquivo ISO-2709 ou um arquivo de texto), essas modificações são realizadas na memória e **não modificam a base de dados**, a menos que se indique explicitamente.

Os registros modificados podem ser vistos na tela ou gravados em uma base de dados ou em um arquivo de saída.

Entre os principais processos de modificação podemos nomear:

Procedimentos para substituição global de padrões (*gizmo*).

Junção de registros (ou parte destes) provenientes de outra base de dados (*join*).

Realizar as operações de atualização de campos (*proc*).

Executar uma Tabela de Seleção de Campos de MicroISIS e acrescentar os dados do resultado a registro na memória (*fst*).

O seguinte exemplo mostra os registros do arquivo mestre cds, que foram escolhidos pelo usuário ao indicar através do teclado o número de registro a ser visualizado.

- MS-DOS:

```
mx seq=con "join=cds='mfn='v1" "proc='D1/1D32001'"
```

- UNIX:

```
mx seq=/dev/ttyp0 "join=cds='mfn='v1" "proc='D1/1D32001'"
```

MX pode atualizar os registros modificados no mesmo arquivo mestre que foi utilizado como entrada:

```
mx cds "proc='D24'" copy=cds -all now
```

Este exemplo apaga o campo com a etiqueta (*tag*) 24 de todos os registros da base de dados cds, realizando as alterações na mesma base.

MX pode pegar parâmetros de um arquivo texto, permitindo, assim, superar as limitações do sistema operacional, que seriam observados se:

1. Uma chamada ao MX tem mais caracteres que os permitidos em uma linha de comandos do sistema operacional (128 caracteres no MS-DOS e 512 caracteres em alguns UNIX).

2. A linha de comandos que está sendo entrada pelo teclado contem caracteres especiais para o sistema operacional.

O seguinte exemplo mostra como utilizar um arquivo de parâmetros:

```
mx in=somefile
```

Onde o arquivo *somefile* contem:

```
\cisis\bases\cds
proc='D1'
copy=\cisis\bases\cds
-all
now
```

Sintaxe

MX versão 5.2a, tabela de sintaxe:

```
CISIS Interface v5.2a/PC32/L/M/32767/16/60/I - Utility MX
Copyright (c)BIREME/PAHO 2006. [http://www.bireme.br/products/cisis]

mx [cipar=<file>] [{mfrl|load}=<n>] [cgi={mx|<v2000_fmt>}] [in=<file>]
   {[db=]<db>|
    seq[/1m]=<file>|
    iso[={marc|<n>}]=<isofile> [isotag1=<tag>]|
    dict=<if>[,<keytag>[,<posttag>[/<postsperrec>]]] [k{1|2}=<key>]}

options:
   {from|to|loop|count|tell|btell}=<n>
   text[/show]=<text>
   [bool=]{<bool_expr>|@<file>}  [invx=<tag101_mf>] [tmpx=<tmp_mf>]

   gizmo=<gizmo_mf>[,<taglist>]  [gizp[/h]=<out_mfx>] [decod=<mf>]

   join=<mf>[:<offset>][,<taglist>]=<mfn=_fmt>
   join=<db>[:<offset>][,<taglist>]=<upkey_fmt> [jmax=<n>]
   jchk=<if>[+<stwfile>]=<upkey_fmt>

   proc=[<proc_fmt>|@<file>]
```

```
        D{<tag>[/<occ>]|*}
        A<tag><delim><data><delim>
        H<tag> <length> <data>
        <TAG[ <stripmarklen>[ <minlen>]]><data></TAG>

        S[<tag>]
        R<mf>,<mfn>
        G<gizmo_mf>[,<taglist>]
        Gsplit[/clean]=<tag>[={<char>|words|letters|numbers|trigrams}]
        Gsplit=<tag>=6words[/if=<if>]
        Gload[/<tag>][/nonl][=<file>]
        Gmark[/<tag>]{/<elem>|/keys|/decs|/<mf>,<otag>[,<ctag>]}=<if>
        Gmarx[/<tag>]/<elem>[@<att>="x"] =<tag>[:&[<att>]|/c[=224]|/i]
        Gdump[/<tag>][/nonl][/xml][=<file>]
        =<mfn>
        X[append=]<mf>

    convert=ansi [uctab={<file>|ansi}]         [actab={<file>|ansi}]
    fst[/h]={<fst>|@[<file>]}                  [stw=@[<file>]]

    [mono|mast|full] {create|copy|append|merge|updatf}=<out_mf>
    [out]iso[={marc|<n>}]=<out_isofile>        [outisotag1=<tag>]
    fullinv[/dict][/keep][/ansi]=<out_if>      [maxmfn=<n>|master=<mf>]
    ln{1|2}=<out_file>                         [+fix[/m]]
    fix=<out_file> tbin=<tag>
    tab[/lines:100000/width:100/tab:<tag>]=<tab_fmt>
    {prolog|pft|epilog}={<diplay_fmt>|@<file>} [lw={<n>|0}]

    {+|-}{control|leader|xref|dir|fields|all} mfrl now
```

MX considera os parâmetros na ordem mostrada na tabela. Em primeiro lugar devem vir, se houver, os parâmetros de inicialização (setup), depois a fonte de entrada de dados, e em último lugar os parâmetros opcionais do processo. Há algumas exceções indicados no manual, por exemplo, *btell=* deve ir antes que *bool=*.

Parâmetros. Descrição geral

Se for entrado o nome do programa MX sem parâmetros, será apresentado o menu de todas as opções possíveis e uma breve descrição de seu uso, tal como é mostrado no quadro anterior.

Parâmetros de inicialização (setup)

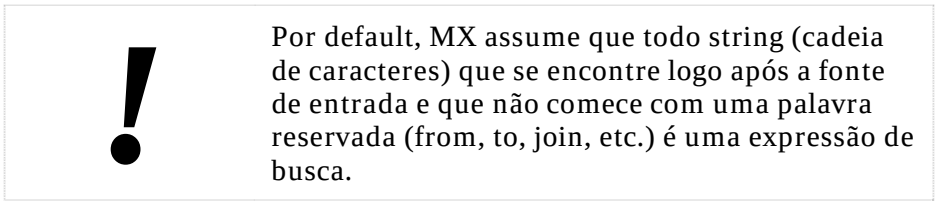
Parâmetros opcionais de inicialização (files, mfrl, fmtl, load), quando um ou mais estão presentes, devem ser colocados antes de qualquer outro parâmetro.

Parâmetros que indicam a fonte de entrada de dados

Um parâmetro obrigatório que indica a fonte de entrada de dados (nome da base de dados, arquivo ISO-2709 ou arquivo de texto), deve ser o primeiro parâmetro, exceto quando na chamada existem parâmetros de inicialização, em cujo caso deve estar imediatamente após os mesmos.

Parâmetros para processamento de dados

Parâmetros opcionais que realizam tarefas sobre os dados de entrada. Seguem ao parâmetro que indica a fonte na linha de comandos.



Os parâmetros de processamento podem ser classificados em:

Parâmetros para seleção de registros

Com estes parâmetros se define um subconjunto da entrada sobre a qual se trabalhará. A forma de definir este subconjunto pode ser por:

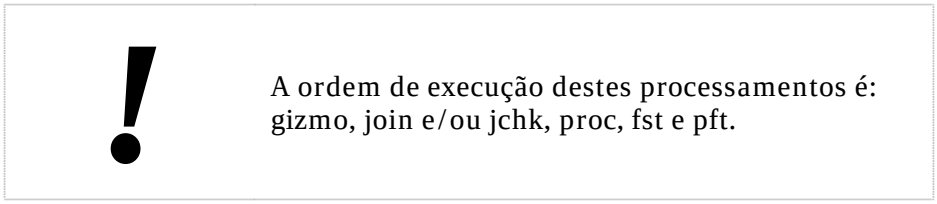
- Uma busca (*bool*)
- Um padrão com o qual se realiza uma busca em texto livre (*text*)
- Um intervalo de registros (cujos limites são indicados por *from*, *to*)
- Quantidade de registros (*count*)
- Salto entre um registro e outro a ser processado (*loop*)

Parâmetros que realizam processamentos

São parâmetros que chamam procedimentos internos que realizam tarefas na memória no conjunto de registros lidos.

Estas tarefas podem ser:

- Realizar substituições globais (*gizmo*)
- Juntar registros (*join*)
- Comparar arquivos mestres com arquivos invertidos (*jchk*)
- Realizar modificações nos campos dos registros (*proc*)
- Aplicar Tabelas de Seleção de Campos (*fst*) aos registros
- Aplicar formatos aos registros (*pft*)



Parâmetros de saída de dados

São parâmetros que permitem, por exemplo, indicar:

- A base de dados de saída (*create*, *copy*, *append*, etc.)
- O nome de um arquivo ISO-2709 de saída (*iso*)
- O nome de arquivos de ligações (*ln1*, *ln2*)
- Chamadas ao sistema operacional (sys)

Parâmetros gerais

São parâmetros que realizam tarefas gerais, por exemplo:

Desativar o *prompt* (interação com o usuário) entre os registros processados (*now*)

Mudar o texto dos *prompts* do mx (*p1*, *p2*)

Modificar opções de visualização (+*fields*, +*all*, -*all*, etc.).

- Redirecionar a saída padrão (>, >>)
- Acompanhamento passo a passo da execução de CISIS (*trace=rec*, *trace=all*, etc.).

Parâmetros que indicam qual é a fonte de entrada de dados

Base de dados de entrada

<[db=]<db>>

Especifica o arquivo mestre a ser lido. Os processamentos a serem realizados, serão executados nos registros deste arquivo mestre.

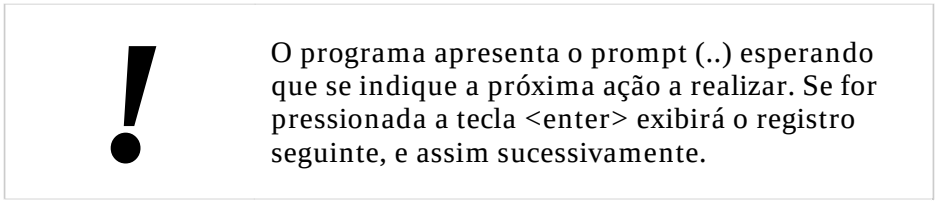
```
C:\isis\data> mx cds
```

ou

```
X:\otrodiretório> mx C:\isis\data\cds
```

Gera uma saída sem formatação, mostrando todos os campos de um registro por vez.

```
mfn= 1
44 «Methodology of plant eco-physiology: proceedings of the Montpellier Symposium»
50 «Incl. bibl.»
69 «Paper on: <plant physiology><plant transpiration> <measurement and instruments>»
24 «Techniques for the measurement of transpiration of individual plants»
26 «^aParis^bUmesco^c-1965»
30 «^ap. 211-224^billus.»
70 «Magalhaes, A.C.»
70 «Franco, C.M.»
..
```




```
mfn= 2

44 «Methodology of plant eco-physiology: proceedings of the Montpellier
Symposium»

50 «Incl. bibl.»

69 «Paper on: <plant evapotranspiration>» 24 «<The> Controlled climate in
the plant chamber and its influence upon assimilation and transpiration»

26 «^c1965»

30 «^ap. 225-232^billus.»

70 «Bosian, G.»

..
```

À frente do *prompt(..)* é possível tomar três ações:

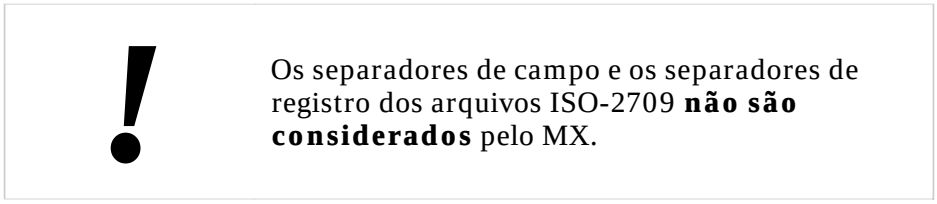
a) Seguir exibindo registros pressionando <enter>
b) Entrar com um *x* minúsculo e pressionar <enter> para sair do programa.
c) Qualquer outro dado entrado será interpretado como uma expressão de busca e logo que pressionar <enter> MX passará a realizar a busca e a mostrar os registros recuperados.

Arquivo ISO-2709 de entrada

iso[={marc|<n>}]=<isofile> [isotag1=<tag>]

Na seção anterior foi considerada como fonte de entrada a bases de dados em formato CDS/ISIS. O programa MX pode ler indistintamente arquivos em formato ISO-2709 e aplicar os mesmos procedimentos que em arquivos mestres (excetuando aqueles processos que requerem o uso do arquivo invertido ou dicionário como, por exemplo, uma busca).

Cada registro em formato ISO-2709 que é lido é transformado internamente em um registro em formato ISIS, no o qual MX trabalha.



Exemplos:

- Percorrer um arquivo ISO-2709:

```
mx iso=\isis\sys\cds.iso
```

  Percorre um arquivo no formato ISO-2709 chamado *cds.iso* que se encontra no diretório *\isis\sys*.

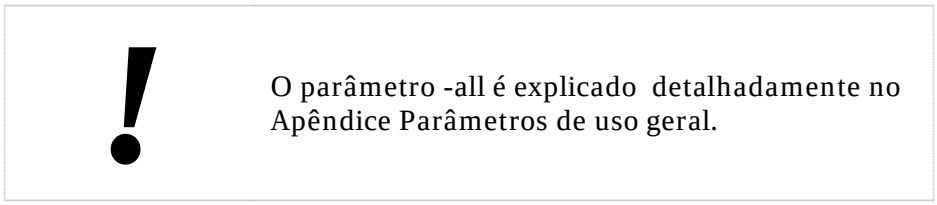
- Ler um arquivo ISO e criar um arquivo mestre com o conteúdo daquele. O *prompt* é suprimido pelo parâmetro *now* (no wait).

```
mx iso=cds.iso create=newcds now
```

- Ler um arquivo no formato ISO-2709 que se encontra no diretório corrente e criar um arquivo mestre chamado *newcds* com os dados fornecidos pelo arquivo ISO.

  Este exemplo realiza o mesmo processamento que o anterior, porém sem apresentar informação na tela. O parâmetro *-all* desativa exibição do fluxo de informação na tela.

```
mx iso=cds.iso create=newcds now -all
```



- Ler um arquivo ISO e criar um arquivo mestre denominado *newcds* com os registros de entrada, começando no mfn 1001. Da mesma forma como no exemplo anterior não será exibida informação na tela (*-all*), nem haverá pausa entre registros, esperando a intervenção do usuário (*now*).

```
mx iso=cds.iso create=newcds from=1001 -all now
```

Tamanho fixo de linha

O MX pode ler arquivos ISO com linha de tamanho fixo. Esta opção é utilizada para intercâmbio de arquivos ISO de computadores e software que assim o requerem (como o MINIISIS de HP). Para ler arquivos com linha de tamanho fixo, deve-se indicar o tamanho da linha logo após o parâmetro *ISO*:

```
mx iso=80=cds.iso create=newcds now -all
```

O exemplo lê o arquivo ISO supondo que o tamanho de linha é de 80 caracteres.

Se o tamanho da linha é variável, então o parâmetro será “0” (zero).

```
mx iso=0=cds.iso create=newcds2 now -all
```

Leitura de arquivos MARC

Os arquivos que cumprem a norma MARC mantêm como fim de cada linha somente o caráter ‘\0Bx’ e não têm ‘\0Cx’ (<CR>) adicional. Para estes arquivos se indicará o parâmetro como segue:

```
mx  iso=marc=input.iso  create=output.iso now –all
```

Dados do Leader do registro MARC

Os registros MARC contêm dados no Leader que não são convertidos automaticamente para CDS/ISIS. Se estes dados forem necessários deverão ser convertidos para campos convencionais, indicando um valor no parâmetro isotag1=<n> como base a partir da qual serão carregados os bytes do Leader. Por exemplo, se for indicado isotag1=3000, o byte de posição 5 do Leader será carregado no campo 3005.

```
mx iso=marc=input.iso  isotag1=3000  create=newcds now -all
```

Arquivo de texto ASCII de entrada

seq[/1m]=<file>

<file>={filename|con|null}

MX pode ter como fonte de entrada um arquivo texto ASCII plano. O tamanho da linha pode chegar até a 1MB, o que é indicado com /1m.

Cada linha deste arquivo de entrada será convertida em um registro do arquivo mestre. Os dados dentro de cada linha poderão ser separados por delimitadores. Desta maneira, cada parte da linha será entrada no registro como campos consecutivos. A quantidade de campos será um a mais do que a quantidade de delimitadores.

O delimitador predefinido em MX é a barra vertical (| *pipe*). Cada linha do arquivo ASCII de entrada pode ter até 32743 caracteres de tamanho para a carga de um campo, 32740 para dois campos, 32735 para três campos, etc.

Tendo-se um arquivo de texto com só uma linha, chamado input.in, cujo conteúdo é:

```
agua|terra|abono
```

Com a linha:

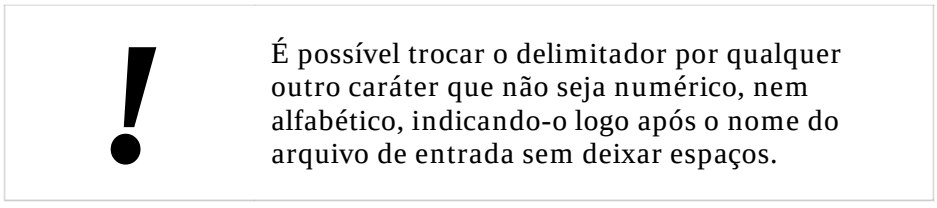
```
mx seq=input.in create=Saída now
```

Se convertirá em:

```
mfn= 1
1 «agua»
2 «terra»
3 «abono»
```

Dois delimitadores consecutivos produzirão um salto no número de campo:

```
agua||tierra||abono
mfn= 1
1 «agua»
3 «terra»
5 «abono»
```



No exemplo seguinte o ponto e vírgula (**;**) é usado como delimitador.

O arquivo input.in contem a linha:

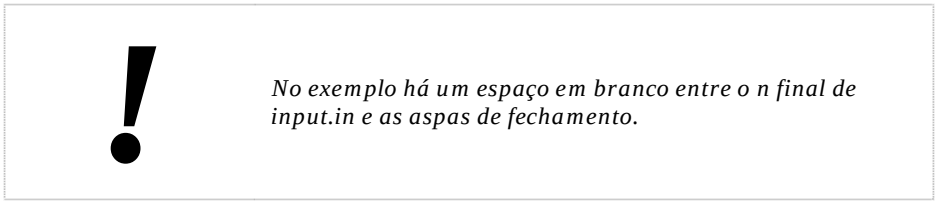
```
agua;abono;terra
```

Para ler o arquivo input.in com ponto e vírgula (;) como separador de campos digitar:

```
mx "seq=input.in;" create=Saída now
```

É possível também usar como delimitador o espaço em branco:

```
mx "seq=input.in " create=Saída now
```



No exemplo há um espaço em branco entre o n final de input.in e as aspas de fechamento.

O arquivo de entrada pode ser gerado, entrando os dados diretamente a partir do teclado, isto é, utilizando o dispositivo padrão de entrada *con* (console) como fonte de entrada dados. Conseqüentemente, é possível criar registros em uma base CDS/ISIS, escrevendo os dados diretamente a partir do teclado.

Examine a seguinte seqüência:

PROCESSAMENTO DO PARAMETRO *con*		
Passos	**Explicação**	**Linhas a digitar**

1	A partir do *prompt* do sistema operacional dá-se a instrução	`C:\path> mx seq=con create=out`
		`(MS-DOS)`
		`unixuser:~$ mx seq=/dev/tty1 create=out`
		`(entrada a partir da consola 1 em UNIX)`

2	Inicia-se uma sessão de entrada de linha que termina ao pressionar <enter>	`agua|terra|vegetales|abono<enter>`

3	Cria-se automaticamente o registro mfn=1 no arquivo mestre *out* e são apresentados os campos do registro	`mfn= 1`
		`1 «agua»`
		`2 «terra»`

PROCESSAMENTO DO PARAMETRO *con*		
Passos	**Explicação**	**Linhas a digitar**
		3 «vegetales» 4 «abono»
4	O *prompt* do MX fica esperando novas instruções. Continua-se a criação de registros, teclando <enter> em resposta à solicitação do *prompt* do MX	..<enter>
5	Entra-se uma nova linha que é terminada ao pressionar <enter>	bovinos|ovinos|equinos<enter>
6	É criado o registro mfn=2 e são apresentados os campos do registro	mfn= 2 1 «bovinos» 2 «ovinos» 3 «equinos»
7	Termina-se a criação de registros, entrando um *x* minúsculo em resposta à solicitação do *prompt* do MX	..x<enter>
8	Finaliza a execução do comando	C:\path> (MS-DOS) unixuser:~$ (UNIX)

O processo pode ser agilizado, evitando a solicitação do *prompt* do MX, colocando o parâmetro *now*. Neste caso, para terminar o processo de entrada, deverão ser pressionadas as teclas <ctrl>+<Z> ou <F6> (em MS-DOS) e <ctrl>+<D> (em UNIX).

A seguinte tabela descreve o processamento, utilizando o parâmetro *now*:

PROCESSAMENTO DO PARÂMETRO *now*		
Passos	**Explicação**	**Linhas a digitar**
1	A partir do *prompt* do sistema operacional dá-se a instrução	C:\path>mx seq=con create=out now (MS-DOS) unixuser:~$ mx seq=/dev/tty1 create=out now (entrada a partir da console 1 em UNIX)

PROCESSAMENTO DO PARÂMETRO *now*		
Passos	**Explicação**	**Linhas a digitar**
2	Inicia-se uma sessão de entrada de linha que termina ao pressionar <enter>	agua|terra|vegetais|abono<enter>
3	É criado o registro mfn=1 E são apresentados os campos do registro	mfn= 1 1 «agua» 2 «terra» 3 «vegetais» 4 «abono»
4	Entra-se uma nova linha que é terminada ao pressionar <enter>	bovinos|ovinos|equinos<enter>
5	É criado o registro mfn=2 E são apresentados os campos do registro	mfn= 2 1 «bovinos» 2 «ovinos» 3 «equinos»
6	Fianliza-se a criação de registros diretamente do teclado pressionando as teclas <ctrl>+<Z> (ou a tecla F6) no DOS, ou as teclas <ctrl>+<D> no UNIX	<Ctrl>+<Z> o <F6> (MS-DOS) <ctrl>+<D> (UNIX)
7	Finaliza a execução do comando	C:\path> (MS-DOS) unixuser:~$ (UNIX)

Base de dados fictícia

con | null

MX permite que a fonte de entrada seja uma base de dados fictícia (*dummy*) chamada *null* (indistintamente). A base *null* consiste de um número ilimitado de registros ativos sem campos de dados (registros vazios). É possível entrar os dados diretamente a partir do teclado, atribuindo o parâmetro *con* (consola) para

entrada. Cada <enter> criará um registro. Para terminar o processo deverá ser dado <ctrl>+Z.

A utilização deste parâmetro não está relacionado diretamente com as operações que são realizadas em uma base de dados. Utiliza-se em processos envolvidos em contagem, gerar listas de números, etc.

Exemplos:

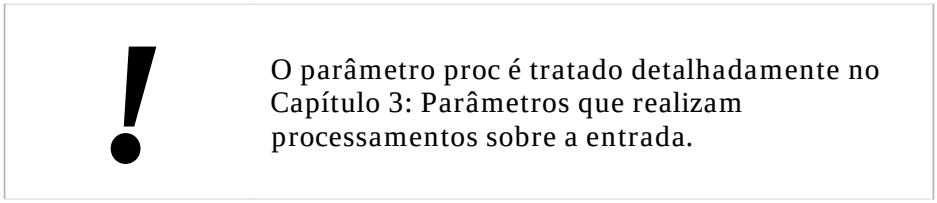
- Apresentar uma lista de números consecutivos de 1 a 50:

```
mx null to=50 pft=mfn(3)/ -all now
```

- Criar um arquivo mestre OUT com registros do 100 ao 200, contendo a literal DBN no campo 999:

```
mx null from=100 to=200 proc='A999#DBN#' now -all create=OUT
```

O parâmetro *proc* realiza a inclusão e a exclusão de campos em um registro. No exemplo, proc adiciona o campo 999 (*A999*) com o conteúdo DBN (*#DBN#*).



O parâmetro proc é tratado detalhadamente no Capítulo 3: Parâmetros que realizam processamentos sobre a entrada.

Arquivo de Parâmetros

in=<file>

Os parâmetros do programa MX podem ser lidos a partir de um arquivo ASCII externo, onde cada parâmetro é entrado como uma linha específica neste arquivo. Desta maneira é possível programar linhas de comandos para MX que podem se estender além do admitido pelo sistema operacional (normalmente limitado em 128 caracteres de tamanho para MS-DOS e em 512 caracteres para UNIX).

Supondo que se tem um arquivo chamado input.in, cujo conteúdo é:

```
iso=entrada.iso
create=Saída
now
from=10
```

```
to=20
-all
```

A linha de comando:

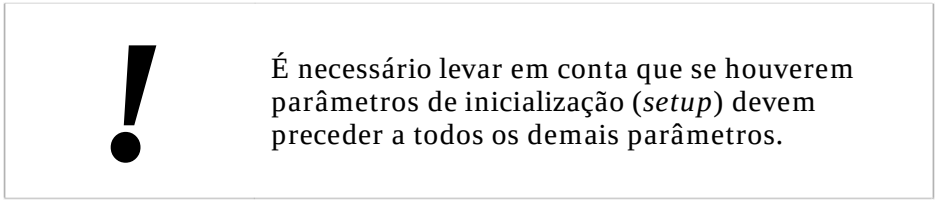
```
mx iso=entrada.iso create=Saída now from=10 to=20 -all
```

Poderia ser escrito como:

```
mx in=input.in
```

O parâmetro *in* pode ser usado em qualquer parte da linha de comandos a partir da primeira posição, pode ser o único parâmetro ou pode ser só uma parte da linha de parâmetros do MX.

Se estiver na primeira posição (ou é o único parâmetro) a primeira linha desse arquivo deve ser o nome do arquivo de dados de entrada.



É possível usar mais de um arquivo *in* em uma linha de comando, e usar o parâmetro *in* dentro de um arquivo *in* com até 8 níveis de recursividade.

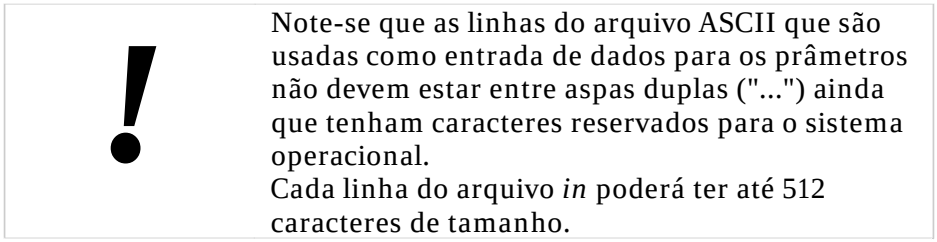
O seguinte exemplo apresenta uma chamada ao MX, utilizando um arquivo de parâmetros *in* entre outros parâmetros:

Seja o arquivo print:

```
\cisis\bases\cds
pft=v70+|; |, " / "v26". "/# mhl,(v69/)
now
-all
tell=10
```

A linha de comando poderia ser:

```
mx in=print from=10 to=50 > arquivo.txt
```



O uso do parâmetro *in* permite preparar arquivos de tarefas em lote (*bat* de MS-DOS e *scripts* de UNIX). Seja o arquivo imprimir.bat cujo conteúdo é o seguinte:

Versão MS-DOS

```
REM Deve entrar os valores from= to=
REM Por exemplo: print 10 20
mx in=print from=%1 to=%2 > arquivo.txt
```

Versão UNIX

```
# Deve entrar os valores from= to=
# Por exemplo: print 10 20
mx in=print from=$1 to=$2 > arquivo.txt
```

Lembre que o arquivo imprimir.bat em UNIX requer permissão de execução.

O operador poderia indicar por exemplo:

- MS-DOS

```
C:\cisis\sys> imprimir 10 40
```

- UNIX

```
unixuser:~/cisis/sys$ imprimir 10 40
```

O que se traduzirá para:

```
mx in=print from=10 to=40 > arquivo.txt
```

Isto por sua vez se converterá em:

```
mx \cisis\bases\cds pft=v70+|; |, " / "v26". "/# mhl,(v69/) now -all
tell=10 from=10 to=40 > arquivo.txt
```

Arquivo Invertido como entrada

dict=<if>[,<keytag>[,<posttag>[/<postsperrec>]]]
[k{1|2}=<key>]

MX lê como entrada de dados as chaves de um arquivo invertido, produzindo registros com etiqueta de campo <ktag>, entre as chaves k1 e k2, com os seguintes parâmetros.

If	Nome do arquivo invertido (I/F)
keytag	Etiqueta que terá a chave lida (*key tag*), por default 1 Este campo tem os seguintes subcampos ^l o tipo de chave (1=curta, 2=longa) ^s o tamanho efetivo da chave (quantidade de caracteres = *keylength*) ^t total de apontadores (*totalpostings*) ^k número seqüencial de leitura (*keyorder*)
posttag	Etiqueta que terá o apontador lido (*posting tag*) Se este parâmetro é indicado, MX combinará as chaves do I/F e seus postings correspondentes. O campo ptag terá os seguintes subcampos ^m MFN do apontador exibido ^t a etiqueta de campo onde a chave foi localizada ^o a ocorrência da etiqueta em que a chave foi localizada ^c a posição, contada em palavras, da chave neste campo ^p o número seqüencial do apontador em exibição ^k o número seqüencial de leitura da chave
postsperrec	Máximo de apontadores lidos por registro, por default todos (*all*)
k1=<key>	Chave inicial a exibir
k2=<key>	Chave final a exibir

Exemplo 1 Lê as chaves do arquivo invertido CDS compreendidas no intervalo entre os termos AFRICA e AFRICAN LANGUAGES

```
Mx dict=cds   k1=africa  "k2=african languages"  now
mfn= 1
1 «AFRICA^l1^s6^t5^k1»
mfn= 2
1 «AFRICAN LANGUAGES^l2^s17^t1^k2»
```

Exemplo 2 Lê três chaves e postings do arquivo invertido CDS a partir do termo AFRICA

```
mx dict=cds,1,2 k1=africa count=3 now
```

```
mfn=     1
  1  «AFRICA^l1^s6^t5^k1»
  2  «^m93^t69^o1^c9^p1^k1»
  2  «^m111^t24^o1^c3^p2^k1»
  2  «^m111^t69^o1^c3^p3^k1»
  2  «^m115^t69^o1^c7^p4^k1»
  2  «^m121^t69^o1^c4^p5^k1»
mfn=     2
  1  «AFRICAN LANGUAGES^l2^s17^t1^k2»
  2  «^m75^t69^o1^c10^p1^k2»
mfn=     3
  1  «AGE^l1^s3^t1^k3»
  2  «^m136^t24^o1^c7^p1^k3»
```

Exemplo 3 Lê as chaves do arquivo invertido CDS, exibindo no máximo três postings cada vez a partir do termo AFRICA, mostra três registros de saída, neste caso três chaves

```
mx dict=cds,1,2/3   k1=africa   count=3 now
mfn=     1
  1  «AFRICA^l1^s6^t5^k1»
  2  «^m93^t69^o1^c9^p1^k1»
  2  «^m111^t24^o1^c3^p2^k1»
mfn=     2
  1  «AFRICA^l1^s6^t5^k1»
  2  «^m115^t69^o1^c7^p4^k1»
  2  «^m121^t69^o1^c4^p5^k1»
mfn=     3
  1  «AFRICAN LANGUAGES^l2^s17^t1^k2»
  2  «^m75^t69^o1^c10^p1^k2»
```

Exemplo 4 Lê as chaves do arquivo invertido CDS, exibindo os postings do termo AFRICA um por um

```
mx dict=cds,1,2/1   k1=africa   count=5 now
mfn=     1
  1  «AFRICA^l1^s6^t5^k1»
  2  «^m93^t69^o1^c9^p1^k1»
mfn=     2
  1  «AFRICA^l1^s6^t5^k1»
  2  «^m111^t24^o1^c3^p2^k1»
mfn=     3
  1  «AFRICA^l1^s6^t5^k1»
  2  «^m111^t69^o1^c3^p3^k1»
mfn=     4
  1  «AFRICA^l1^s6^t5^k1»
  2  «^m115^t69^o1^c7^p4^k1»
mfn=     5
  1  «AFRICA^l1^s6^t5^k1»
  2  «^m121^t69^o1^c4^p5^k1»
```

Parâmetros que realizam processamentos sobre a entrada

Parâmetros que aplicam formatos à entrada

Este Parâmetro provê as especificações de formato para a visualização dos registros. Os registros apagados (*logically deleted*) não são visualizados através do parâmetro *pft*=.

O MX suporta todas as instruções da linguagem de formatação do CDS/ISIS padrão para DOS (exceto *format exits*) e acrescenta algumas extensões desenvolvidas pela Interface CISIS. Muitas destas novas instruções estão incorporadas ao Winisis, mas MX não aceita as instruções para formatos gráficos em RTF que o Winisis usa.

O manual completo de formatos está disponível no site do Modelo da BVS, através da URL: http://bvsmodelo.bvsalud.org/.

Especificação do formato de visualização na linha de comando

pft=<display_fmt>

O seguinte exemplo aplica o formato mfn/v24/v26 aos registros obtidos do arquivo mestre de entrada (cds):

```
mx cds pft=mfn/v24/v26
```

Se a instrução de formato incorpora caracteres reservados para o sistema operacional (tais como: > | % etc.) ou espaços em branco, o parâmetro deverá estar entre aspas duplas:

```
mx cds "pft=mfn,/(v70+|; |)/v24/#"
```

Especificação do formato de visualização através de um arquivo

pft= @[<file>]

O MX permite especificar um arquivo (*pft=@[<file>]*) onde se encontra o formato a ser utilizado. Esta é uma maneira mais prática de especificar um formato de visualização, assim a chamada ao MX fica mais clara e, por outro lado, não se perde o formato uma vez executado o comando.

Além disto, desta forma, é superada uma limitação dos sistemas operacionais, já que o tamanho de uma linha de comandos é limitada (128 caracteres MS-DOS e 512 caracteres UNIX), visto que um formato extenso não poderia ser escrito explicitamente.

Se não for provido nome de arquivo, o MX usará por default o formato que tem o mesmo nome da base de dados:

mx cds pft=@ é equivalente a mx cds pft=@cds.pft

Ao especificar um arquivo, seu nome pode ter mais de seis caracteres de tamanho, pode estar localizado em um diretório diferente do da base de dados, e pode ter ou não extensão (se tiver deverá ser informado, ainda que seja pft).

Exemplos:

```
mx cds pft=@cdsnew.pft
mx cds pft=@\dbisis\outro_dir\outro.pft
mx cds pft=@long_name.pft
mx cds pft=@sin_ext
```

Formatos condicionais

prolog | epilog

prolog é uma especificação que é executada no primeiro registro da saída, e *epilog* no último registro.

Exemplo:

```
mx cds prolog='Primeiro: ' pft=mfn/ epilog='último' from=10 to=20
now
```

Tamanho de linha (*line width*)

lw={<n>|0}

A linha de saída tem um tamanho predefinido de 78 caracteres. É possível alterar o tamanho da linha com o parâmetro *lw=n*.

```
mx cds "pft=mfn,/(v70+|; |)/v24/#" lw=40 to=20 now
```

Extrai dados do conteúdo de uma variável CGI

getenv('cgi=',<varfmt>)

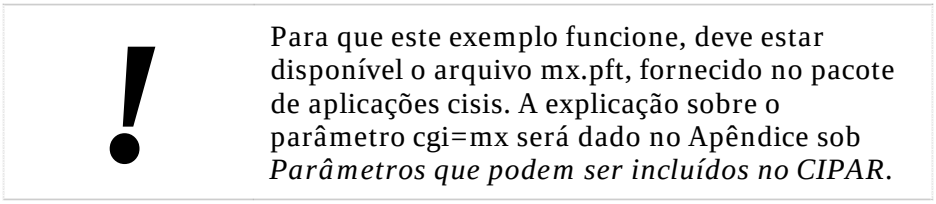
<varfmt> é uma especificação de formato que gera um nome de variável cgi. As múltiplas ocorrências são separadas pelo caráter %

```
set "REQUEST_METHOD=GET"
set "QUERY_STRING=db~cds&btell~0&bool~plants*water"
mx cds cgi=mx "pft='Buscando por \"',getenv('cgi=bool'),'\" na base:
',getenv('cgi=db')/,'  MFN   Titulo'/,mfn,x2,mhl,v24.50,'...'/"
```

que apresenta na saída:

```
 Buscando por "plants*water" na base: cds
  MFN   Titulo
000004  Mc An Electric hygrometer apparatus for me...
..
Buscando por "plants*water" na base: cds
  MFN   Titulo
000011  Measurement of water stress in plants...
..
Buscando por "plants*water" na base: cds
```

```
  MFN   Titulo
000013  Experience with three vapour methods for measuring...
->x
```



Para que este exemplo funcione, deve estar disponível o arquivo mx.pft, fornecido no pacote de aplicações cisis. A explicação sobre o parâmetro cgi=mx será dado no Apêndice sob *Parâmetros que podem ser incluídos no CIPAR*.

Obtém um arquivo temporário vazio

getenv('tmp=',[<pathfmt>])

O formato obtém um nome de arquivo temporário vazio do diretório de trabalho atual, ou do especificado no caminho (*path*) que é especificado no formato <pathfmt>. Se for especificado 'ci_tmpdir' então o caminha é obtido das seguintes variáveis de ambiente: *ci_tempdir*, *temp*, *tmp*.

```
mx null pft=getenv('tmp=')
TMP1.$$$

mx null pft=getenv('tmp=','ci_tempdir')
C:\windows\TEMP\TMP1.$$$

set ci_tempdir=C:\work
mx null pft=getenv('tmp=','ci_tempdir')
C:\work\TMP1.$$$

echo ci_tempdir=C:\work2 >xcip
mx cipar=xcip null pft=getenv('tmp=','ci_tempdir')
C:\work2\TMP1.$$$
```

Parâmetros que selecionam o conjunto de registros a serem processados

Especificação da expressão de busca na linha de comando

bool=<bool_expr_spec>

O parâmetro *bool* permite realizar buscas em bases de dados. MX provê todas as operações booleanas da linguagem de busca do CDS/ISIS.

O resultado de uma busca é um conjunto de registros que atendem à especificação da expressão de busca.

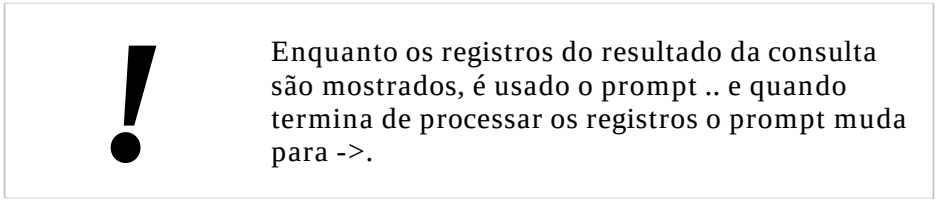
A palavra *bool* é opcional, já que MX considera expressão de busca tudo aquilo que não é um parâmetro. Os dois exemplos seguintes produzem o mesmo resultado:

```
mx cds bool=water
mx cds water
```

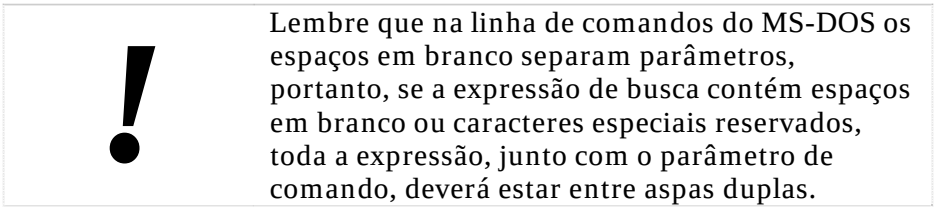
A saída da execução da busca é:

```
mx cds culture pft=mfn/
2       2     CULTURE
2       2     Set #000001
Hits=2
000050
..
000064
->
```

Primeiro é apresentada a resolução da busca e é atribuído um número de seqüência ao resultado (*Set #000001*). Em seguida são mostrados os registros de acordo com o formato preestabelecido.



É possível continuar fazendo consultas a partir do *prompt* de forma interativa, mas não é possível, com este parâmetro, fazer referência aos resultados anteriores em consultas do tipo #1 + #2, etc. Para isto é necessário o uso do parâmetro tmpx (será apresentado mais adiante neste capítulo).



Lembre que na linha de comandos do MS-DOS os espaços em branco separam parâmetros, portanto, se a expressão de busca contém espaços em branco ou caracteres especiais reservados, toda a expressão, junto com o parâmetro de comando, deverá estar entre aspas duplas.

```
mx cds "bool=water + soil"
mx cds "water + soil"
```

O programa MX aceita os operadores booleanos +, *, ^ que CDS/ISIS usa, assim como suas expressões literais OR, AND, NOT.

```
mx cds "bool=water or soil"
mx cds "agricult$ and plants"
```

Carregar a expressão de busca a partir de um arquivo

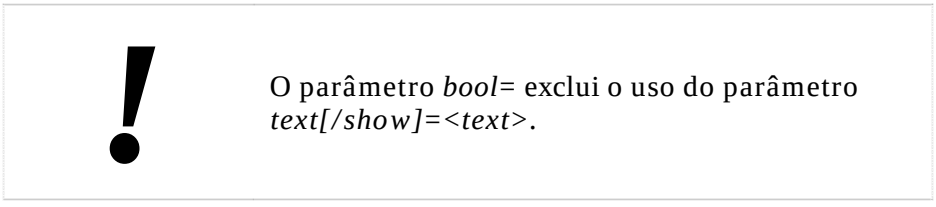
bool=@<file>

bool=@<*file*> pega a expressão de busca a partir de um arquivo ASCII externo. Neste caso a expressão não precisa estar entre aspas.

Supondo que o arquivo *consulta.001* contenha a expressão: agricult$ and plants, então o último exemplo é traduzido para qualquer das duas formas seguintes:

```
mx cds bool=@consulta.001
mx cds @consulta.001
```

O parâmetro *bool* é opcional, mas seu uso poderá ser necessário quando a expressão de busca começa com uma palavra reservada do MX.



O parâmetro *bool*= exclui o uso do parâmetro *text[/show]*=<*text*>.

Querendo-se recuperar registros que contêm a palavra now:

`mx cds now`	Não fará o esperado, visto que interpreta now como parâmetro, não como expressão de busca.
`mx cds "now"`	Não fará o esperado, porque interpreta now como parâmetro e não como expressão de busca.
`mx cds bool=now`	Está correcto.
`mx cds "bool=now"`	Está correcto.

Utilização dos resultados intermediários de uma busca

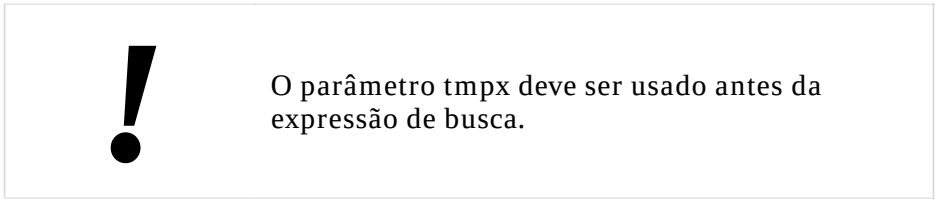
tmpx=<tmpx_query_dbn>

Parâmetro obsoleto:

b70=<b70_query_dbn>

O parâmetro *tmpx* armazena os resultados intermediários de uma sessão de busca, permitindo referenciar resultados anteriores em formulações do tipo #1 + #2.

O nome atribuído à direita do sinal de igual é uma base de dados ISIS que MX cria e reinicializa automaticamente, e não é apagada ao final da execução.

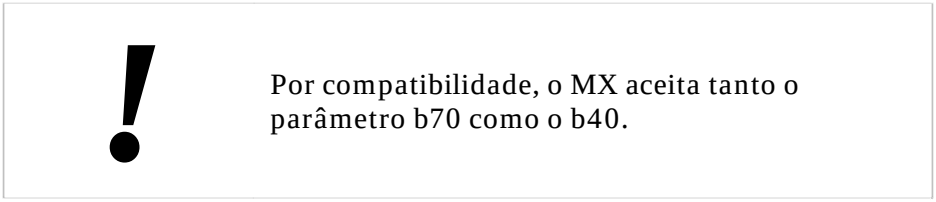


O parâmetro tmpx deve ser usado antes da expressão de busca.

```
mx cds tmpx=x70 plants water #1*#2 pft=mfn/ now
      8  PLANTS
      8  Set #000000001
Hits=8
     14  WATER
     14  Set #000000002
Hits=14
      3  Operation *
      3  Set #000000003
Hits=3
000004
000011
```

```
000013
```

Note-se que, se for informado o parâmetro *tmpx*, é possível realizar consultas interativas a partir do *prompt* do MX. fazendo referências a resultados anteriores.



Por compatibilidade, o MX aceita tanto o parâmetro b70 como o b40.

No seguinte exemplo os termos que são introduzidos um a um pelo operador, como resposta ao *prompt*, são mostrados destacados em negrito. O resultado do processo é armazenado na base X70.

```
mx cds tmpx=x70 plants pft=mfn/
      8  PLANTS
      8  Set #000000004
Hits=8
000001
..water
     14  WATER
     14  Set #000000005
Hits=14
000004
..#1 and #2
      3  Operation *
      3  Set #000000006
Hits=3
000004
..
000011
..
000013
->x
```

É possível realizar consultas complexas, lendo a expressão de busca a partir de um arquivo externo. Supondo que o arquivo PERFIL.001 tenha a seguinte formulação de busca:

```
Water
```

```
Plants
#1 and #2
Transpiration
#3 or #4
```

A seguinte linha de comando produz uma saída para arquivo texto com os registros recuperados de acordo com o perfil atribuído. Observe-se que cada consulta individual deve ser entrada como linha independente no arquivo PERFIL.001.

```
mx CDS tmpx=x70 in=perfil.001 pft=@cds.pft now > out_001.txt
```

Eliminação da estatística da busca

btell=0

É possível suprimir a saída na tela que registra o processamento dos passos intermediários e final da busca. Neste caso o parâmetro *btell=0* deve vir antes da formulação da busca.

Exemplo:

Sem parâmetro *btell*=	• **Com parâmetro *btell=0***
`mx cds "water * plants" pft=mfn/ now`	`mx cds btell=0 "water * plants" pft=mfn/ now`

`14 WATER`
`8 PLANTS`
`3 Operation *`
`3 Set #000000001`
`Hits=3`
`000004`
`000011`
`000013`
	`000004`
	`000011`
	`000013`

btell=2

Lista os termos do dicionário, quando executa o $ de truncamento.

Busca em vários arquivos invertidos

invx=M/F

O invx contém registros com o campo v101 repetitivo, com o seguinte formato:

```
^pPrefijo o asterisco^yArquivo[^uUseprefix][^mMessage]
```

Exemplo:

Cria-se um master **cdsinvx**, com os seguintes campos

```
^p*^ycds^mText words
^pAU ^ycdsaut^mAuthor
^pTI ^ycdstit^mTitle words
^pKW ^ycdkw^mKeywords
```

Cria-se os invertidos associados a CDS

```
mx cds "fst=1 0 (v70/)" fullinv=cdsaut
mx cds "fst=1 4 v24" fullinv=cdstit
mx cds "fst=1 2 v69" fullinv=cdskw
```

Consultas

```
mx cds invx=cdsinvx "KW plants " pft=mfn,x1,v24,x1,v69
      2  PLANTS
      2  Set #000000001
Hits=2
000054 Vegetation as a geological agent in tropical deltas Paper on:
<vegetation><geology><deltas><tropical
zones><sedimentation><plants><organic
matter><wetlands>..
000069 Some important animal pests and parasites of East Pakistan Paper on:
<pests><parasites><biology><ecology><plants><agriculture><public
health><food><Bangladesh>
->x

mx cds invx=cdsinvx "KW plants * east"
      2  PLANTS
      9  EAST
      1  Operation *
      1  Set #000000001
Hits=1
mfn=    69
 24  «Some important animal pests and parasites of East Pakistan»
```

```
 26  «^c1966»
 30  «^ap. 285-291^billus.»
 44  «Scientific problems of the humid tropical zone deltas and their
implicatio
ns: proceedings of the Dacca Symposium»
 50  «Incl. bibl.»
 69  «Paper on:
<pests><parasites><biology><ecology><plants><agriculture><public
 health><food><Bangladesh>»
 70  «Yosufzai, H.K.»
100  «1001»
->x
```

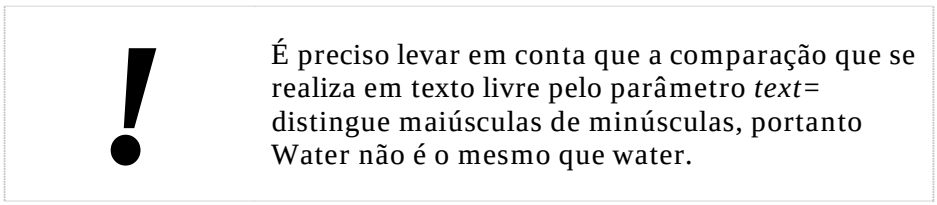
Se o prefixo for indicado entre [ ], então se aplica a todos os termos da formulação booleana:

```
mx cds invx=cdsinvx "[KW] plants * east"
      2  PLANTS
         EAST
         Operation *
         Set #000000001
Hits=0
->x
```

Buscas em texto livre

text[/show]=<text>

Esta função permite realizar buscas em texto livre (não indexado). Busca a cadeia de caracteres indicada em <*text*> em todos os campos da base de dados de entrada e seleciona os registros que atendem à busca.



! É preciso levar em conta que a comparação que se realiza em texto livre pelo parâmetro *text=* distingue maiúsculas de minúsculas, portanto Water não é o mesmo que water.

Exemplo:

```
mx cds text=water iso=Saída.iso now -all
```

O exemplo realiza uma busca e exporta os registros encontrados para um arquivo ISO-2709

Parâmetro text/show

text/show

O parâmetro *text/show* somente exibe a informação do registro e campo do qual a recuperação foi feita. Mostra duas linhas, uma que contem o número de registro, etiqueta, ocorrência onde aparece o texto buscado; a segunda linha apresenta a ocorrência completa onde se encontra o texto buscado.

Exemplo:

```
mx cds text/show=water now
```

Saída em tela:

```
mfn 56|tag 69|occ 1|water
69 «Paper on: <regime of waters><sediment transport><deltas>»
```

Indica que o registro mfn=56 contem a cadeia de caracteres water na primeira ocorrência do campo com tag 69.

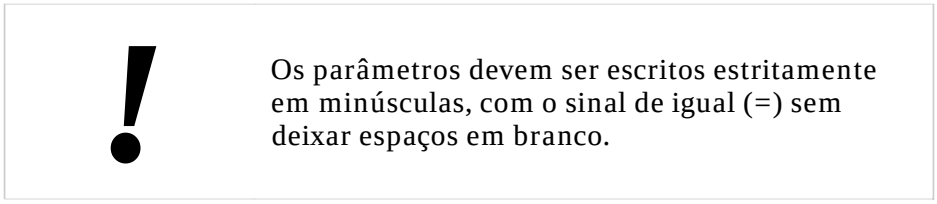
Outras formas de selecionar o conjunto de registros a processar

Seleção por intervalo

from= <n> to=<n>

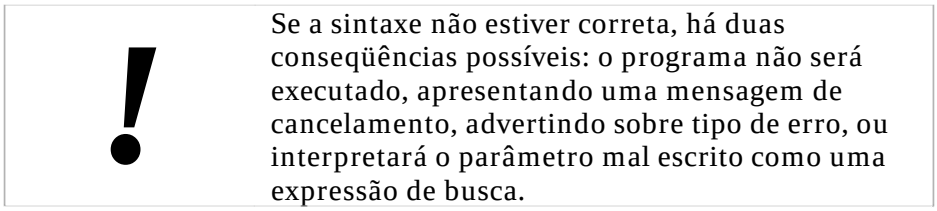
```
mx cds from=24 to=50
```

Esta linha de comandos exibe na tela a partir do registro 24 até o registro 50 inclusive. Se não for indicado *from* será assumido, como início, o primeiro registro da base de dados; se não for indicado *to* o processamento será até o final da base de dados, ou até que o operador saia, pressionando x.



Os exemplos seguintes NÃO são corretos:

`mx cds FROM=10`	O parâmetro deve ser em minúsculas
`mx cds from= 10`	Há um espaço em branco entre o = e o 10
`mx cds from 10`	Falta o sinal =



- A expressão do exemplo (a) gerará o resultado:

```
FROM=10
Set #000001
Hits=0
->
```

Isto significa que buscou a expressão FROM=10 na base de dados.

- A expressão do exemplo (b) gerará o resultado:

```
FROM
Set #000001
Hits=0
Expression syntax error 5: '='
->
```

O MX interpreta que há um erro na expressão de busca devido ao sinal =

- A expressão do exemplo (c) gerará o resultado:

```
FROM
Set #00001
```

```
Hits=0
10
Set #000002
Hits=0
->
```

Isto implica que buscou as palavras FROM e 10 e não encontrou nenhum registro.

Com esta instrução serão visualizados somente os números de MFN dos registros 24 a 50.

```
mx cds from=24 to=50 pft=mfn/
```

Seleção a cada *n* registros

loop=<n>

O parâmetro *loop=n* processa um registro e pula *n* registros, processa o registro *n+1*, pula outros *n* e assim sucessivamente.

Exemplo:

```
mx cds loop=10
```

São recuperados os registros 1, 11, 21, etc.

Selecionar *n* registros

count=<n>

O parâmetro *count=n* seleciona exatamente *n* registros a partir de um início dado. Se não for indicado o registro de início, começa a partir do primeiro registro.

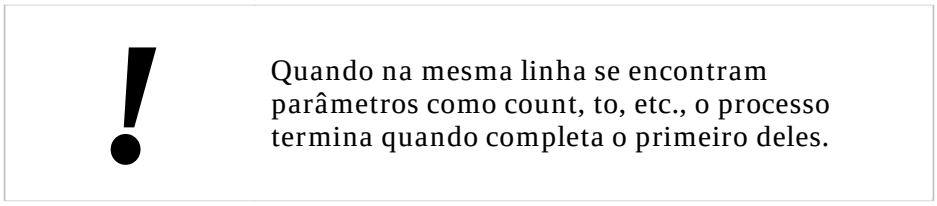
Exemplo	**Saída**

`mx cds from=24 to=50 loop=5 pft=mfn/ now`	`000024`
	`000029`
	`000034`
	`000039`
	`000044`
	`000049`

Exemplo	Saída

`mx cds from=24 to=50 count=3 loop=5 pft=mfn/ now`	`000024`
	`000029`
	`000034`

`mx cds from=24 to=50 count=9 loop=5 pft=mfn/ now`	`000024`
	`000029`
	`000034`
	`000039`
	`000049`



Quando na mesma linha se encontram parâmetros como count, to, etc., o processo termina quando completa o primeiro deles.

Parâmetros que modificam registros

proc=[<proc_fmt>][@<proc_fmt_file>]

<proc_fmt> y <proc_fmt_file> sintaxe

```
D{<tag>[/<occ>]|*}
A<tag><delim><data><delim>
H<tag> <length> <data>
<TAG[ <stripmarklen>[ <minlen>]]><data></TAG>
S[<tag>]
R<mf>,<mfn>
G<gizmo_mf>[,<taglist>]
Gsplit[/clean]=<tag>[={<char>|words|letters|numbers|trigrams}]
Gload[/<tag>][/nonl][=<file>]
Gmark[/<tag>]{/<elem>|/keys|/decs|/<mf>,<otag>[,<ctag>]}=<if>
Gmarx[/<tag>]/<elem>[@<att>="x"] =<tag>[:&[<att>]|/c[=224]|/i]
Gdump[/<tag>][/nonl][/xml][=<file>]
=<mfn>
X[append=]<mf>
```

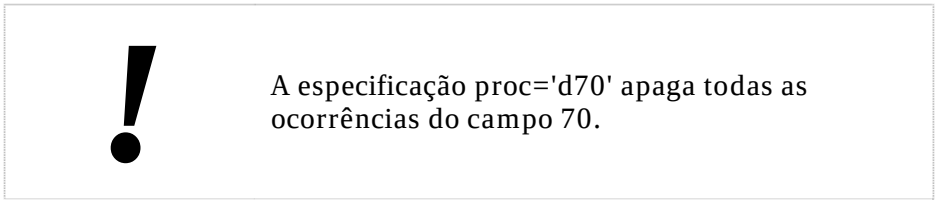
O parâmetro *proc* permite especificar, através de um formato, modificações a serem realizados nos campos do registro fonte. Possibilita apagar, incluir e substituir conteúdo dos campos. As operações a serem realizadas são definidas

como instruções de formato, que pode ser submetido diretamente na linha de comandos ou tomado a partir de um arquivo externo.

Ainda é importante destacar que as modificações dos registros não são realizadas na base de origem (base de onde provêm os dados), mas na base de destino. No caso de não existir a especificação da base de destino, as alterações poderão ser vistas na tela, mas se perderão uma vez terminada a execução. Para que as alterações sejam realizadas na mesma base que foi utilizada como entrada, esta deve ser especificada como base de destino.

Exemplos:

```
mx cds from=1 to=10 now proc='d70'
```



As alterações serão vista na tela, mas não são realizadas realmente, porque não foi especificada a base de destino. Para que as alterações se reflitam no mesmo arquivo mestre que se especificou como entrada, é necessário que seja indicado através do parâmetro *copy*:

```
mx cds from=1 to=10 proc='d70' copy=cds now
```

Outra possibilidade é especificar um arquivo em vez de escrever o formato na linha de comandos:

```
mx cds from=1 to=10 now proc=@modifica
```

Realiza as modificações atribuídas em um arquivo que contem o seguinte formato:

```
'd70'
```

Quantidade máxima de parâmetros aceitos

in=	1024
arquivos in= (recursivos)	16
proc=	1024
join=	128

gizmo=	16
linhas no arquivo stw	799

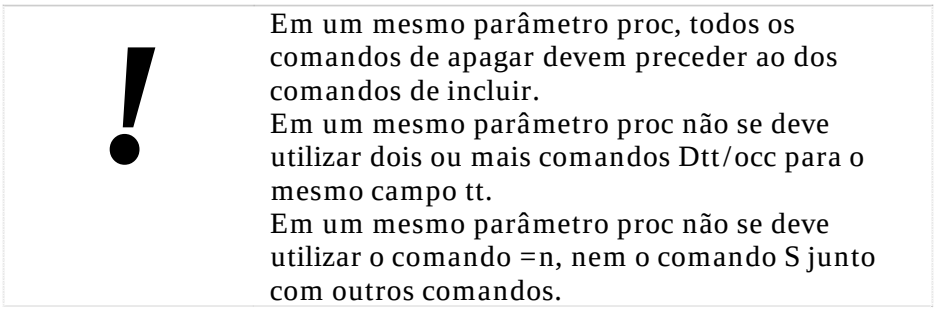
É possível especificar até 1024 proc= parâmetros, incluindo fst, read, write, I/F update *proc* em uma linha de comandos do MX. Cada *proc* sucessivo atuará sobre o registro na sua situação atual, pelo que se um *proc* (ou qualquer procedimento anterior na execução) modifica o registro original de entrada, o próximo *proc* atuará sobre os dados existentes neste momento.

São aceitas todas as instruções padrão de formato do CDS/ISIS, incluindo as condicionais do tipo *IF*, mas não são aceitas as chamadas a programas em IsisPascal (*format exits*). Aceita ainda as variáveis e0...e9, s0 ..s9 e o comando while. É possível incluir proc dentro de proc (recursivo) sem limites.

A linguagem de formatação CDS/ISIS também aceita a instrução proc( ). Para detalhes, veja o Manual de linguagem de formatação CDS/ISIS.

Na tabela seguinte são descritos os comandos de uso mais freqüente que o *proc* pode executar, os outros são explicados em detalhe de forma individual (MX aceita todos os comandos da função *fldupdat( )* da Interface CISIS):

Comandos da função fldupdat ( )		
Comando	**Explicação**	**Exemplo**
D.	Apaga lógicamente o registro.	proc='d.'
D*	Apaga todos os campos do registro.	proc='d*'
Dtt	Apaga todas as ocorrências do campo tt.	proc='d26'
Dtt/occ	Apaga a ocorrência occ do campo tt.	proc='d26/3'
Att#str#	Inclui a cadeia de caracteres str como uma nova ocorrência do campo tt.	proc='A999#cds#'
Htt n str_n	Inclui a cadeia de caracteres str_n de n bytes de tamanho como uma nova ocorrência do campo tt.	proc='H99 8 CDS/ISIS'
=<mfn>	Substitui o número de registro (mfn) por n .	proc='=10'
S<tag>	Ordena as entradas ao diretório do registro por tag.	proc='s' proc='s70'



Os comandos listados na tabela anterior podem ser escritos indistintamente em maiúsculas ou minúsculas.

Basicamente, a idéia é produzir um formato que, como resultado, gere uma cadeia de caracteres do tipo:

```
d36a999#1999#a70#Magalhaes, A.C.#
```

através da qual, o campo 36 é apagado, é incluído 1999 no campo 999 e Magalhaes, A.C. no campo 70.

Esta *string* (cadeia de caracteres) pode ser provida por um formato como o seguinte:

```
if p(v36) and a(v999) then 'd36a999#'v47'#'fi,"a70#"v36"#"
```

A *string* é formada a partir da existência do campo 36, cujo conteúdo é Magalhaes, A.C., e a ausência do campo 999.

Exemplos:

Criar uma base com todos os registros da base de origem (CDS), eliminando os campos 69 e 70:

```
mx cds proc='d69d70' create=Saída now -all
```

Incluir informação no campo 999 para um grupo de registros:

```
mx cds proc='A999#1998#' from=100 to=120 copy=cds now -all
```

Neste exemplo as alterações são realizadas na mesma base utilizada como fonte de entrada, portanto, as alterações se refletirão na mesma base.

Importar um arquivo ISO-2709 com o MFN armazenado no campo 999 dos registros ISO:

```
mx iso=datos.iso proc='='v999 create=master -all now
```

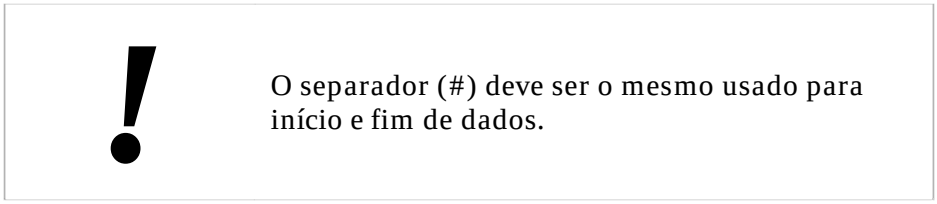
Exportar registros para um arquivo ISO-2709, incluindo MFN do registro de origem no campo 999 dos registros ISO:

```
mx master proc='D999A999/'mfn'/' -all now iso=dato.iso
```

Função ‘A' (inclusão de campo) att#str#

A função ‘A' é composta por:

Comando	Descrição
a	Indica que será incluído um campo.
tt	Tag do campo onde os dados serão incluídos.
#	Separador entre *tt* e os dados. O separador pode ser qualquer caráter, desde que não seja numérico e que não ocorra na cadeia de caracteres a incluir.
str	Cadeia de caracteres a incluir. A cadeia pode entrar como uma cadeia de caracteres estática ou ser extraída de um campo de dados do mesmo registro ou de outro (inclusive de um registro de outra base de dados).
#	Separador que indica o fim da *string* de dados.



O exemplo:

```
mx cds proc='A999#1998#' from=100 to=120 copy=cds now -all
```

É equivalente a:

```
mx cds "proc='a999{1998{'" from=100 to=120 copy=cds now -all
```

Exemplos:

- Eliminar o campo 70 e incluir somente a primeira ocorrência desse campo como nova ocorrência do campo 100.

Os seguintes exemplos apresentam duas formas de realizar a mesma tarefa:

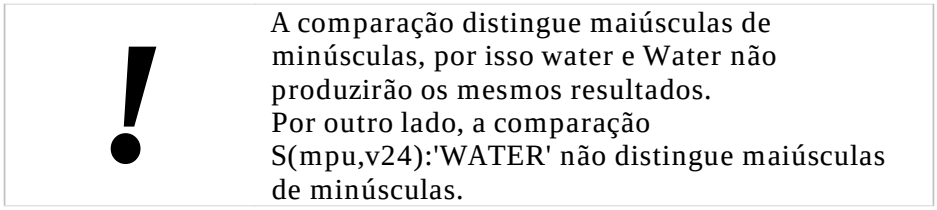
```
mx cds "proc='d70a100#',v70[1],'#' " from=10 to=20 now -all
mx cds "proc='d70',|a100@|v70[1]|@| " from=10 to=20 now -all
```

- Apagar todos os campos do registro quando o campo 24 contém a palavra water, e exportar o registro para uma base chamada Saída. O formato que contem as instruções para o *proc* deve ser obtido de um arquivo externo (teste).

```
mx cds proc=@teste now –all create=Saída
```

O arquivo teste contem a linha de formato:

```
If v24:'water' then ‘D*' fi
```



Também se pode operar sobre a mesma base de dados utilizada como fonte de entrada.

```
mx CDS proc='d70' copy=CDS now to=1
```

- Ordenar os campos do registro por número de tag:

```
mx CDS proc='s' copy=CDS now –all
```

Registro de entrada:

```
mfn= 1
44 «Methodology of plant eco-physiology: proceedings of the Montpellier Symposium»
50 «Incl. bibl.»
69 «Paper on: <plant physiology><plant transpiration> <measurement and instruments>»
24 «Techniques for the measurement of transpiration of individual plants»
26 «^aParis^bUmesco^c-1965»
```

```
30 «^ap. 211-224^billus.»
70 «Magalhaes, A.C.»
70 «Franco, C.M.»
```

Registro de saída:

```
mfn= 1
24 «Techniques for the measurement of transpiration of individual plants»
26 «^aParis^bUmesco^c-1965»
30 «^ap. 211-224^billus.»
44 «Methodology of plant eco-physiology: proceedings of the Montpellier
Symposium»
50 «Incl. bibl.»
69 «Paper on: <plant physiology><plant transpiration><measurement and
instruments>»
70 «Magalhaes, A.C.»
70 «Franco, C.M.»
```

Observe que no exemplo a saída do processo se realiza na mesma base de dados de entrada.

Função R<mf>,<mfn>

Inclui no arquivo mestre de saída os campos do registro ativo

<mf>	nome do master de entrada de dados
<mfn>	número do registro

```
mx null proc='a10/1/a20/1/' create=xxx
10 «1»
20 «1»

mx null proc='a20/2/' append=xxx
20 «2»

mx out  proc='Rxxx,1' proc='Rxxx,2' copy=out count=1 now
mfn= 1
10 «1»
20 «1»
20 «2»
```

Grava no master out no registro ativo (mfn=1) os campos dos registros mfn=1 e mfn=2 do master xxx.

Função <TAG[<stripmarklen>[<minlen>]]><dados></TAG>

Inclui <*dados*> como uma nova ocorrência do campo <*tag*>. As marcações no estilo "<*marks*>" são eliminadas do texto até um máximo de <*stripmarklen*> caracteres. Se o texto, após a retirada dos delimitadores "<*marks*>", for menor do que <*minlen*> caracteres, a edição não será efetuada.

<tag>	Etiqueta do campo
<stripmarklen>	Tamanho máximo da marcação (infinito por default)
<minlen>	Tamanho mínimo do campo resultante (zero por default)

- Inclui uma ocorrência com o termo *text* no campo 10

```
mx null "proc='<10>text</10>'"
mfn= 1
10 «text»
```

- Inclui uma ocorrência com o termo *x*<*mark*>*y*</*mark*>*z* no campo 10 eliminando as marcações <*mark*></*mark*> mas não o conteúdo

```
mx null "proc='<10>x<mark>y</mark>z</10>'"
mfn= 1
10 «xyz»
```

- Inclui uma ocorrência com o termo *x*<*mark*>*y*</*mark*>*z* no campo 10 eliminando a marcação <*mark*> em função do critério de tamanho da marcação, neste caso seis caracteres.

```
mx null "proc='<10 6>x<mark>y</mark>z</10>'"
mfn= 1
10 «xy</mark>z»
```

- Inclui uma ocorrência com o termo *x*<*mark*>*y*</*mark*>*z* no campo 10 eliminando as marcações <*mark*></*mark*> mas não seu conteúdo, pois tem o mínimo esperado de três caracteres.

```
mx null "proc='<10 99 3>x<mark>y</mark>z</10>'"
mfn= 1
10 «xyz»
```

- Não inclui a ocorrência com o termo *x*<*mark*>*y*</*mark*>*z* no campo 10, pois após a eliminação das marcações <*mark*></*mark*> o conteúdo não alcançou o mínimo esperado de quatro caracteres.

```
mx null "proc='<10 99 4>x<mark>y</mark>z</10>'"
mfn= 1
```

Função X[{create|copy|append|merge}=]<mf>

Grava os dados do registro corrente no arquivo <out_dbn>, se não existir será criado. É possível, em uma linha de comandos, atribuir alternativamente diferentes arquivos de saída para um mesmo registro.

- Atribui como saída o master *dbn1*, em seguida grava o mesmo registro no mfn=3 do master *dbn2*.

```
mx null proc='a10/1/' proc='Xdbn1' proc='a20/2/' proc='=3' proc='Xdbn2'
mx dbn1
mfn= 1
10 «1»
mx dbn2
mfn= 3
10 «1»
20 «2»
```

- Cria primeiro um arquivo *xfile.pft*, em seguida atribui como nomes de saída de masters a primeira ocorrência do campo 70 dos registros 1,2,3. Observe o formato *xfile.pft* que acrescenta os sinais -+ ao nome. O registro mfn=1 de CDS é gravado em um master "*Magalhaes*" e os registros mfn=2, mfn=3 são gravados em um master "*Bosian-+*".

```
echo replace(replace(s(v70[1].8),',','-'),' ','+') >xfile.pft
mx cds proc='Xappend='@xfile.pft,  pft=mfn,x1,v70[1]/  count=3 now
000001 Magalhaes, A.C.
000002 Bosian, G.
000003 Bosian, G.
mx Magalhae    pft=mfn,x1,v70[1]/
000001 Magalhaes, A.C.
mx Bosian-+   pft=mfn,x1,v70[1]/
000002 Bosian, G.
000003 Bosian, G.
```

Função G<gizmo_mf>[,<taglist>]

Aplica um gizmo ao registro, que pode ser a uma lista de campos específicos.

```
echo transpiration~xxx> xxx.lst
mx seq=xxx.lst~  create=xxx now
mx cds proc='Gxxx,2' from=2  count=1 now
mfn=     2
```

```
24  «<The> Controlled climate in the plant chamber and its influence upon
assimilation and xxx»
30  «^ap. 225-232^billus.»
26  «^aParis^c1965»
70  «Bosian, G.»
```

Função Gsplit[/clean]=<tag>[={<char>|words|letters|numbers|trigram}]

- Exemplo com <char> usa-se o ;

```
echo Perez, J.; Garcia, María; Machado, A. > xxx.lst
mx seq=xxx.lst proc='Gsplit/clean=1=;'
mfn=     1
  1  «Perez, J.»
  1  «Garcia, María»
  1  «Machado, A.»
..
```

- Separa-se o campo 44 de CDS por palavra

```
mx cds proc='Gsplit=44=words'
mfn=     1
44  «Methodology»
44  «of»
44  «plant»
44  «eco»
44  «physiology»
44  «proceedings»
44  «of»
44  «the»
44  «Montpellier»
44  «Symposium»
..x
```

Função Gsplit=<tag>=6words[/if=<if>]

Extrai as palavras de um texto na seguinte seqüência:

- Palavra 1, palavras 1+2, palavras 1+2+3 ... palavras 1+2+...6,
- quando termina a série, vai para a segunda palavra
- começa a mesma seqüência.

Exemplo:

Methodology of plant eco-physiology: proceedings of the Montpellier Symposium

```
mx cds proc='Gsplit=44=6words'
Methodology
Methodology of
Methodology of plant
Methodology of plant eco
Methodology of plant echo physiology
Methodology of plant echo physiology proccedings
of
of plant
of plant echo
of plant echo physiology
…
```

Opção /if=<if>

Faz um lookup em um arquivo invertido e extrai somente as expressões válidas

Função Gload[/<tag>][/nonl][=<file>]

Carrega o arquivo <file> em formato binário no campo <tag>

Se for indicado /nonl elimina os saltos de fim de linha <CR>

Funçao Gdump[/<tag>][/nonl][/xml][=<file>]

Grava o conteúdo do campo <tag> em um arquivo.

Opção /nonl: elimina as quebras de fim de linha <CR>.

Opção /xml: grava na estrutura xml.

Substituição global de padrões

gizmo=<gizmo_mf>[,<taglist>] [gizp[/h]=<out_mfx>]
[decod=<mf>]

O parâmetro *gizmo* permite realizar substituições globais no conteúdo dos campos de uma base CDS/ISIS, converter uma cadeia de caracteres em outra, e assim realizar modificações, codificação/decodificação, compressão de dados, etc.

Estas substituições podem ser realizadas em todos os registros da base ou em um conjunto de registros (selecionados por meio de uma busca, um intervalo, etc). As substituições, por sua vez, podem abranger todo o registro, ou apenas alguns campos.

Para realizar substituições como, por exemplo, os símbolos < > por / /, ou a cadeia de caracteres plants por PLANTS, etc., é necessário dispor de um arquivo mestre *gizmo*. Este arquivo mestre tem em princípio dois campos: o campo 1 contém o dado a ser substituído, e o campo 2 o novo valor. Cada par de dados será um registro do arquivo mestre *gizmo*.

Cada registro de entrada é submetido ao procedimento de substituição estabelecida no arquivo *gizmo*. Ao começar a execução de MX os dados do arquivo *gizmo* são carregados como uma tabela na memória e são ordenados alfabeticamente pelo valor do campo 1 e pelo seu tamanho (desta maneira as cadeias de caracteres mais longas são convertidas antes que as curtas).

É possível especificar até 16 gizmo em uma linha de comando do MX. Cada *gizmo* sucessivo atuará sobre o registro na sua situação atual, por isso, se um *gizmo* modifica o registro original de entrada, o próximo *gizmo* atuará sobre os dados existentes neste momento.

Exemplos:

- Para o exemplo seguinte é criada uma base de dados chamada *teste*, usando os parâmetros conhecidos do MX e os dados são introduzidos diretamente a partir do teclado:

MS-DOS	UNIX

`mx seq=con create=teste -all now`
`<|/`
`>|/`
`plants|PLANTAS`
`<ctrl>+<Z> (o <F6>)`
	`unixuser:~$ mx seq=/dev/tty1 create=teste -all now`
	`<|/`
	`>|/`
	`plants|PLANTAS`

	<ctrl>+<D>

Obtendo os seguintes registros:

```
mfn= 1
1 «<»
2 «/»
mfn= 2
1 «>»
2 «/»
mfn= 3
1 «plants»
2 «PLANTAS»
```

O conteúdo dos campos título e descritores do registro MFN=1 são:

```
mx cds to=1 "pft=mfn/v24/v69"
000001
Techniques for the measurement of transpiration of individual plants
Paper on: <plant physiology><plant transpiration><measurement and
instruments>
```

Ao aplicar:

```
mx cds gizmo=teste to=1 "pft=mfn/v24/v69"
```

dá como resultado:

```
000001
Techniques for the measurement of transpiration of individual PLANTAS
Paper on: /plant physiology//plant transpiration//measurement and
instruments/
```

A modificação não afeta a base cds que provê os dados de entrada, porque a mesma é realizada na saída (neste caso, na tela).

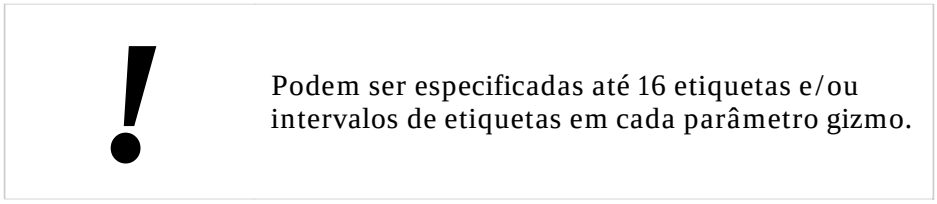
Para modificar realmente os registros, o resultado do processamento deve ser enviado para o mesmo arquivo mestre, como no seguinte exemplo:

```
mx cds gizmo=teste to=1 copy=cds –all now
```

Se, ao contrário, se deseja gerar uma nova base de dados é necessário criá-la:

```
mx cds gizmo=teste to=1 create=Saída –all now
```

É possível que a modificação seja restrita a um campo específico do registro, indicando após o parâmetro *gizmo* a etiqueta ou etiquetas nos quais a substituição será realizada. É possível indicar um intervalo de etiquetas separadas por uma barra (/).




```
mx cds gizmo=teste,69,24 to=1 create=Saída -all now
mx cds gizmo=teste,35/56 to=1 create=Saída -all now
```

- Outra aplicação do *gizmo* pode ser a codificação e eventual compressão de dados, onde termos muito freqüentes podem ser substituídos por códigos ou valores especiais:

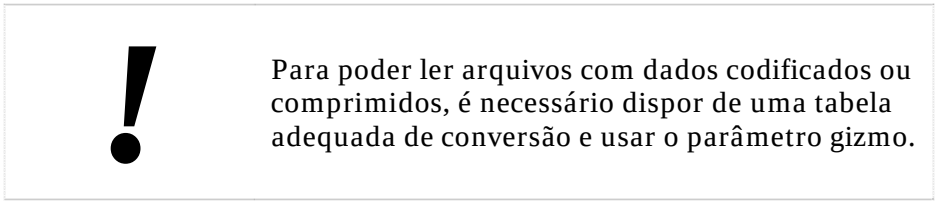
```
mx seq=con create=tabela1 now -all
^aParis^bUmesco|*1*
transpiration|*2*
<ctrl>Z
```

Antes da conversão:

```
mx cds to=1 pft=mfn/v69/v24/v26
000001
Paper on: <plant physiology><plant transpiration><measurement and
instruments>
Techniques for the measurement of transpiration of individual plants
^aParis^bUmesco^c1965
```

Após à conversão:

```
mx cds gizmo=tabela1 to=1 pft=mfn/v69/v24/v26
000001
Paper on: <plant physiology><plant *2*><measurement and instruments>
Techniques for the measurement of *2* of individual plants
*1*^c1965
```



No exemplo a seguir é criada a **tabela2** para realizar a conversão inversa a da **tabela1**, usando o parâmetro *proc* :

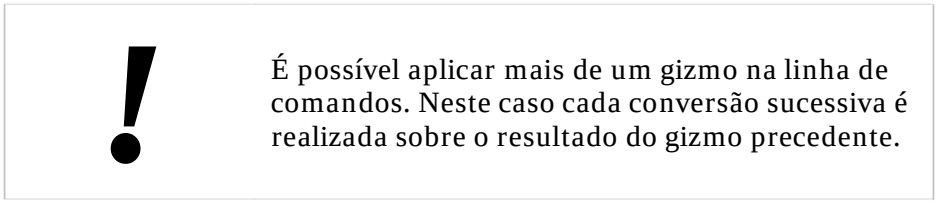
```
mx tabela1 create=tabela2 now -all "proc='d*a1#',v2,'#a2#',v1,'#'"
```

Descrição

A instrução *proc* começa apagando todos os campos, então inclui como campo 1 o conteúdo do campo 2 e por último inclui como campo 2 o conteúdo do campo 1.

Tendo as bases *tabela1* e *tabela2,* pode-se realizar o seguinte exemplo:

```
mx CDS to=1 create=OUT gizmo=tabela1 now –all
mx OUT gizmo=tabela2
```



É possível aplicar mais de um gizmo na linha de comandos. Neste caso cada conversão sucessiva é realizada sobre o resultado do gizmo precedente.

- Se forem aplicados dois *gizmos* com *tabela1* e *tabela2* à base CDS, que produzem duas conversões recíprocas (portanto os dados originais não variam).

```
mx CDS gizmo=tabela1 gizmo=tabela2
```

O parâmetro *gizmo* tem muitas aplicações. Por exemplo, poderia se dispor de tabelas em vários idiomas com as equivalências entre o código numérico de um descritor e sua versão nesse idioma. A base de dados terá como dado o valor numérico e a conversão para os termos do idioma desejado será feita durante o processo de formatação.

Outro exemplo interessante é ter uma tabela com os caracteres acentuados e símbolos como campo 1 e seus códigos correspondentes em html (como &eacute para é) no campo 2, desta maneira podem ser obtidos documentos html como saída.

Outra aplicação poderia ser a decodificação de abreviaturas ou siglas de instituições, ou abreviaturas de idiomas, de países, etc., que no CDS/ISIS padrão só são realizadas através da função ref+lookup.

Tabelas de conversão através de códigos ASCII ou hexadecimais

As tabelas de conversão *gizmo* são arquivos mestres compostos de dois campos, cujos conteúdos são a cadeia a substituir (campo 1) e a cadeia pela qual será substituída (campo2).

Também, é possível especificar tais cadeias de caracteres através da representação numérica dos caracteres que os compõe, permitindo que a tabela gizmo inclua caracteres que não podem ser digitados ou visualizados. Ou que não sejam adequados para a transmissão de dados de modo não transparente.

Este tipo de tabela pode ter até quatro campos, que são: os campos 1 e 2 (cadeia a ser substituída e cadeia pela qual será substituída) e os campos 11 e/ou 21 onde se indica que tipo de informação dos campos 1 e/ou 2 respectivamente. O valor destes campos pode ser: **hex** (se o conteúdo do campo 1 ou 2 está codificado em dois dígitos hexadecimais) ou **asc** (se o conteúdo do campo 1 ou 2 está codificado em três dígitos decimais). É possível entrar comentários em cada registro, usando os campos não reservados: 1, 2, 11, 21.

Exemplo:

As seguintes tabelas gizmo são equivalentes:

TAG	Conteúdo
1	OF THE
2	Ç

TAG	Conteúdo
1	OF THE
2	128
21	Asc

TAG	Conteúdo
1	079070032084072069
2	Ç
11	asc

TAG	Conteúdo
1	4F4620544845
2	Ç
11	hex

1. Tendo-se um arquivo mestre onde se deseja substituir o caráter - (hexadecimal 2D), pelo caráter ~ (hexadecimal 7E).

Primeiro é necessário gerar uma tabela para realizar a substituição, que será um arquivo mestre que contenha:

TAG	Conteúdo
1	2D
2	7E
11	Hex
21	Hex

Para isto é preciso criar um arquivo texto chamado *cambio.seq*, cujo conteúdo é:

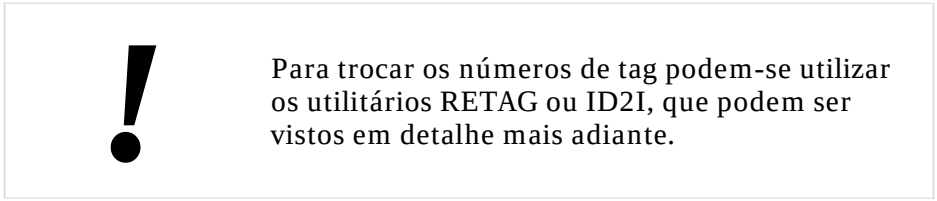
```
2D|7E|hex|hex
```

Em seguida se executa a seguinte linha de comandos:

```
mx seq=cambio.seq create=cambio -all now
```

Com isto foi gerada uma tabela com os quatro campos necessários para realizar a substituição. Embora os campos 3 e 4 devessem ter tag 11 e 21 respectivamente, para alterar os números de *tag* podem-se empregar os utilitários RETAG ou ID2I, mas neste exemplo será feito com o MX, usando o comando *proc*:

```
mx cambio proc='d3d4a11#hex#a21#hex#' copy=cambio
```



Agora está tudo certo para realizar a substituição:

```
mx cds gizmo=cambio -all now
```

Estatística da conversão por gizmo

gizp/h=<out_mfx>

O parâmetro *gizp=<out_mfx>* gera um arquivo mestre para cada *gizmo* utilizado na linha de comando, com informação que registra o processamento efetuado nos registros. Cada base terá como prefixo 'dbnx' mais um número seqüencial, por exemplo: *dbnx0, dbnx1, dbnx2*, etc. O parâmetro /h indica que os dados são guardados em hexadecimal.

Exemplo:

```
mx cds gizmo=codigo,80 gizmo=precod,87 now –all create=OUT gizp=dbnx
```

Criará dois arquivos mestres de saída: *dbnx0* correspondente ao processamento efetuado com *gizmo=codigo*, e *dbnx1* correspondente ao processamento efetuado com *gizmo=precod*.

Cada um destes arquivos mestres terá um registro para cada entrada do gizmo que tenha sido usado efetivamente na execução do MX. Os campos criados são:

TAG	Descrição
1	Campo 1 do arquivo gizmo.
2	Campo 2 do arquivo gizmo.
10	Quantidade de vezes que esta conversão foi usada no processo.
31	Tamanho do campo 1 em caracteres.
32	Tamanho do campo 2 em caracteres.

Opção [decod=<mf>]

Nome de um master que codifica/decodifica os dados. O arquivo mf contem o caráter separador dos elementos para decodificar e o campo onde se aplicam.

Tendo-se dois arquivos de codificação, *decodp* e *zdecsp*:

```
mx decodp
mfn=     1
  1  "ZDECSP"
  2  ";"
  3  "87"
  3  "88"
  3  "71"
  3  "76"
..

mx zdecsp
mfn=     2
  3  "Temefos"
mfn=     3
  3  "Matadouros"
mfn=     4
  3  "Abreviaturas"
```

```
mfn=     5
  3  "Abdome"

mx lilacs "pft=mfn/(|87=|v87/)(|88=|v88/)(|71=|v71/)(|76=|v76)#"

432938
87=^d731;^d9525^s22004;^d2465^s22004

432937
87=^d3977;^d1732^s22045;^d30912^s22045;^d7840;^d6893

432936
87=^d1942^s22004;^d2465^s22004;^d731
76=841;21044;21030
```

Juntar bases de dados - JOIN

join=<mf>[:<offset>][,<taglist>]=<mfn=_fmt>

join=<db>[:<offset>][,<taglist>]=<upkey_fmt> [jmax=<n>]

O parâmetro *join* permite que um ou vários registros da mesma ou outras bases de dados sejam juntados ao registro que está sendo processado neste momento. O registro em processamento pode vir de um arquivo mestre ISIS, um ISO-2709, ou um arquivo ASCII. No registro de saída serão acrescentados todos os campos dos registros referenciados (ou campos selecionados destes) e, além disto, serão criados campos de controle com a informação do processamento (numerados a partir de 32001).

Podem ser passados os seguintes argumentos para o parâmetro *join*:

ARGUMENTOS DO PARÂMETRO *join*	
Argumento	**Descrição**
<mf> / <db>	Uma base de dados alternativa na qual se buscarão registros para juntar com o registro que está sendo processado. A base alternativa poderia ser a mesma dos dados de entrada

ARGUMENTOS DO PARÂMETRO *join*	
Argumento	**Descrição**
[:offset]	Deslocamento para os tags joined
[<taglist>]	Uma lista dos campos que serão extraídos dos registros da base alternativa para anexar ao registro de entrada. Se não for especificada a lista, todos os campos que existem nos registros encontrados serão juntados. Os campos de dados do registro encontrado poderão ser numerados.
<_fmt> @<file>	Formato com o qual o registro de entrada é lido para extrair chaves com as quais os registros da base de dados alternativa serão buscados. Pode ser um formato explícito ou a referência para um arquivo externo.
<upkey_fmt>	<_fmt>|@<file> É necessário gerar a(s) chave(s) de acordo com o I/F, tipicamente em *uppercase*
[jmax=<n>]	Limita a quantidade de registros que serão agregados ao registro original, procedentes da ação do parâmetro *join* para cada chave gerada. Este limite se aplica para cada parâmetro *join* da linha de comandos.

Exemplo:

```
mx cds join=autor,2,3=mhu,(v70/) create=newcds -all now
```

Descrição

1. Pega cada registro da base de dados cds.

Para cada autor gerado pela especificação do formato mhu,(v70/), realiza uma busca na base *autor*.

Se encontrar algum registro na base *autor*, acrescenta um campo de controle e acrescenta os campos 2 e 3 deste ao registro que está sendo processado.

Por último, grava o registro na base *newcds*.

Os seguintes diagramas esquematizam o processo que se desenvolve quando MX executa o *join*:

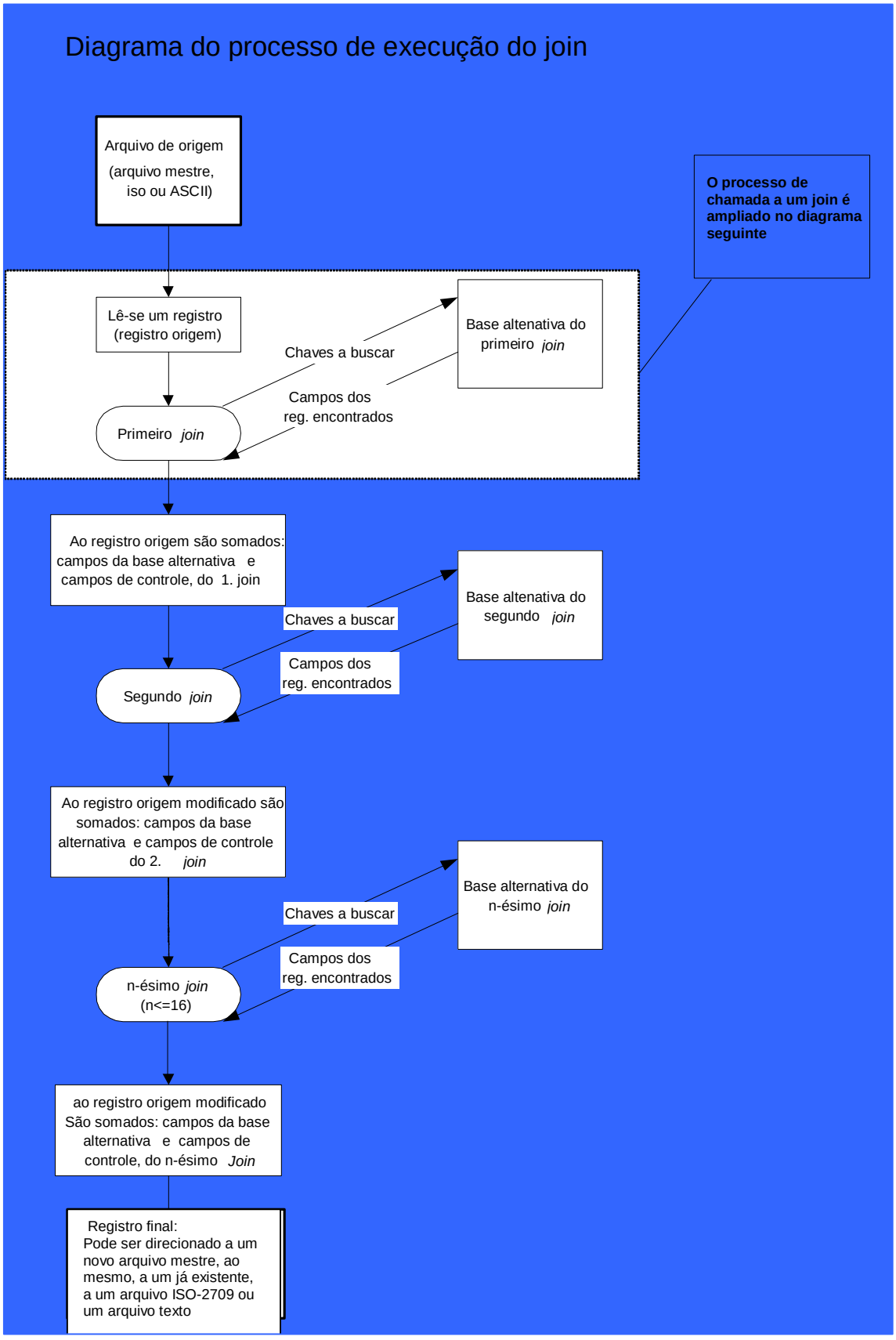
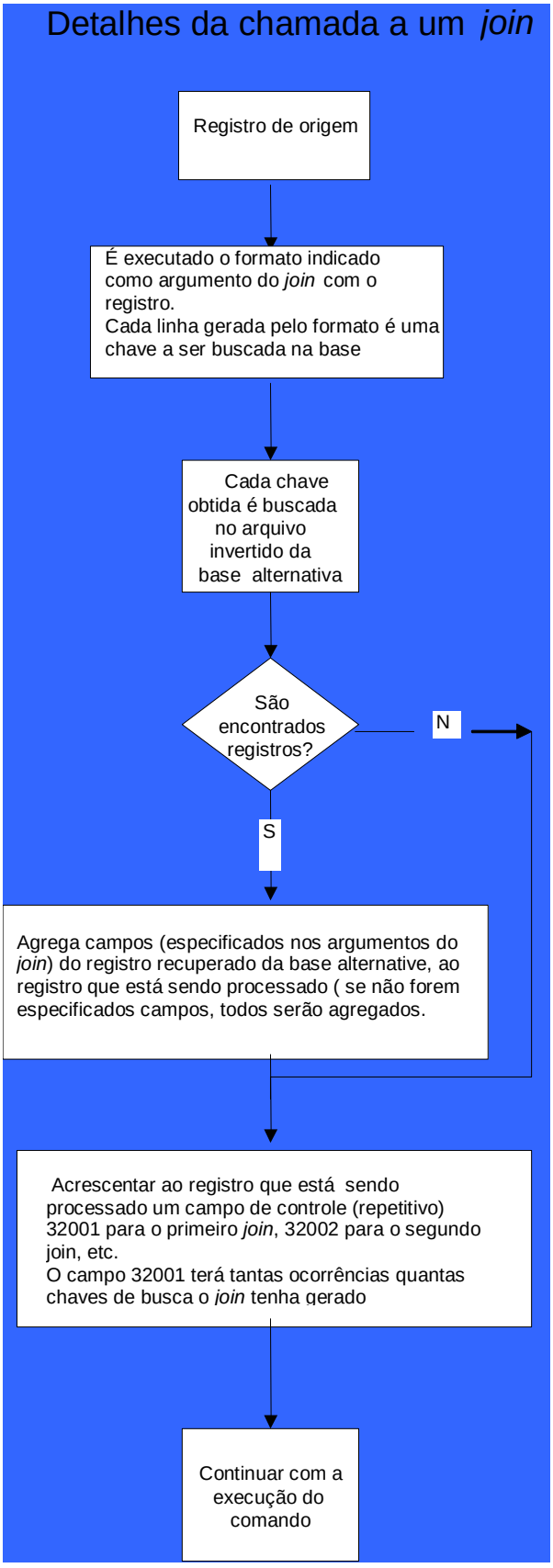


Diagrama do processo de execução do join
Arquivo de origem
(arquivo mestre, iso ou ASCII)
O processo de chamada a um join é ampliado no diagrama seguinte
Lê-se um registro (registro origem)
Chaves a buscar
Base altenativa do primeiro join
Campos dos reg. encontrados
Primeiro join
Ao registro origem são somados: campos da base alternativa e campos de controle, do 1. join
Base altenativa do segundo join
Chaves a buscar
Campos dos reg. encontrados
Segundo join
Ao registro origem modificado são somados: campos da base alternativa e campos de controle do 2. join
Base alternativa do n-ésimo join
Chaves a buscar
Campos dos reg. encontrados
n-ésimo join (n<=16)
ao registro origem modificado São somados: campos da base alternativa e campos de controle, do n-ésimo Join
Registro final:
Pode ser direcionado a um novo arquivo mestre, ao mesmo, a um já existente, a um arquivo ISO-2709 ou um arquivo texto


Detalhes da chamada a um *join*
Registro de origem
É executado o formato indicado como argumento do *join* com o registro.
Cada linha gerada pelo formato é uma chave a ser buscada na base
Cada chave obtida é buscada no arquivo invertido da base alternativa
São encontrados registros?
N
S
Agrega campos (especificados nos argumentos do *join*) do registro recuperado da base alternative, ao registro que está sendo processado (se não forem especificados campos, todos serão agregados.
Acrescentar ao registro que está sendo processado um campo de controle (repetitivo) 32001 para o primeiro *join*, 32002 para o segundo join, etc.
O campo 32001 terá tantas ocorrências quantas chaves de busca o *join* tenha gerado
Continuar com a execução do comando

É possível especificar até 128 *joins* em uma linha de comando do MX. Cada *join* sucessivo atuará sobre o registro na sua situação atual, por isso, se um *join*, ou qualquer procedimento anterior na execução, modifica o registro original de entrada, o próximo *join* atuará sobre os dados existentes neste momento

Veja a explicação mais adiante no parâmetro jmasx=<n>

Exemplo:

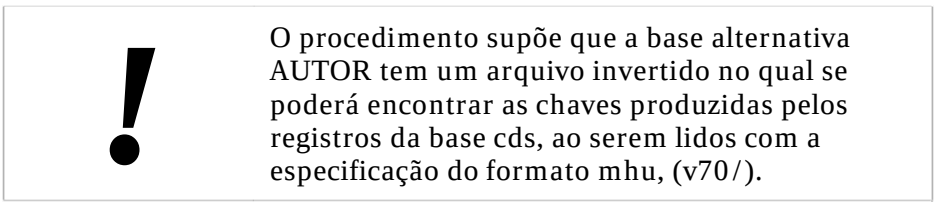
- Dispondo-se de uma base de dados AUTOR que contenha dados normalizados de autores, como se mostra a seguir:

```
mfn= 1
1 «Magalhaes, A.C.»
2 «Ing. Agrónomo»
3 «1935-1990»
mfn= 2
1 «Franco, C.M.»
2 «Bioquímico»
3 «1940-»
...
```

É realizada a seguinte operação sobre a base cds, cujo campo 70 contém os autores.

```
mx cds join=AUTOR=mhu,(v70/)
```

Então: são somados todos os campos da base autor à base cds, são agregados os campos de controle 32001. As substituições são mostradas apenas na tela, já que não foi indicado nenhum destino para a saída.



O procedimento supõe que a base alternativa AUTOR tem um arquivo invertido no qual se poderá encontrar as chaves produzidas pelos registros da base cds, ao serem lidos com a especificação do formato mhu, (v70/).

O processo *join* envolve os seguintes passos:

PROCESSO DO PARÂMETRO *join*	
Descrição	**No exemplo**
1) Lê o registro da base de origem cds.	Base de origem: cds
2) Aplica o formato especificado no *join* ao registro lido. O modo do formato MHU, ou MPU, é necessário em geral para obter as chaves, devido ao fato que o arquivo ISISUC.TAB padrão converte minúsculas em maiúsculas para a geração do arquivo invertido. O registro lido é o registro proveniente da base cds.	mhu, (v70/)
3) Para cada chave obtida - ou seja, para cada linha gerada pelo formato especificado - busca extrair o MFN do próximo *posting* encontrado no arquivo invertido da base de dados alternativa.	Base alternativa: AUTOR
4) Acrescenta ao registro original um campo de controle (repetitivo) 32001 para o primeiro *join*, 32002 para o segundo *join*, etc. O campo 32001 terá tantas ocorrências quantas chaves distintas tenham sido produzidas no passo 2.	
5) Lê na base alternativa o registro correspondente ao MFN obtido e acrescenta os campos selecionados na lista <*tags*>. Se não for especificado, todos os campos serão considerados.	
6) Volta para o passo 2.	

Lista de seleção e renumeração de campos <tags>

- Seleção:
  Podem ser selecionados campos do registro alternativo das seguintes formas:
  1. Indicando os números de campos e separando-os por vírgula

     ```
     mx cds join=AUTOR,1,2,3=mhu,(v70/)
     ```

  2. Indica-se um **intervalo de campos,** usando valor inferior/valor superior.

     ```
     mx cds join=AUTOR,1/3=mhu,(v70/) copy=cds
     ```

- Renumeração:
  Indica-se o novo número de um campo, especificando como
  número novo:número antigo

  ```
  mx CDS "join=AUTOR,100:1=mhu,(v70/)"
  ```

> **!** Em todos os casos um campo é copiado com todas as suas ocorrências.

Conteúdo dos campos de controle (32001, 32002, etc.)

Os campos com tag 32001, 32002, etc. são gerados pelo MX para registrar o processamento do *join*.

Exemplo:

```
32001 «AUTOR^l2^kFRANCO, C.M.^o2^m2»
```

SUBCAMPOS DO CAMPO 32001	
Campo	**Conteúdo**
	Primeiro dado, sem indicador de subcampo: nome da base alternativa.
^l[1|2]	Subárvore do arquivo invertido de onde a chave foi extraída: Para termos até 10 caracteres de tamanho. Para termos entre 11 e 30.
^k	Chave com a qual foi feita a busca.
^o	Número da ocorrência da chave.
^m	MFN do registro da base alternativa da qual procedem os dados que são acrescentados ao registro original. Se não for encontrado o termo no dicionário da base alternativa, este subcampo não é gerado.

Exemplos:

- O seguinte exemplo produzirá a saída mostrada abaixo. Note que o campo 1 do registro alternativo é renumerado como 100 quando é copiado para o registro original.

```
mx cds join=AUTOR,2,3,100:1=mhu,(v70/)

mfn= 1

44 «Methodology of plant eco-physiology: proceedings of the Montpellier
Symposium»

50 «Incl. bibl.»

69 «Paper on: <plant physiology><plant transpiration> <measurement and
instruments>»

24 «Techniques for the measurement of transpiration of individual plants»

26 «^aParis^bUmesco^c-1965»

30 «^ap. 211-224^billus.»

70 «Magalhaes, A.C.»

70 «Franco, C.M.»

32001 «AUTOR^l2^kMAGALHAES, A.C.^o1^m1»
```

```
100 «Magalhaes, A.C.»
2 «Ing. Agrónomo»
3 «1935-1990»
32001 «AUTOR^l2^kFRANCO, C.M.^o2^m2»
100 «Franco, C.M.»
2 «Bioquimico»
3 «1940-»
..
```

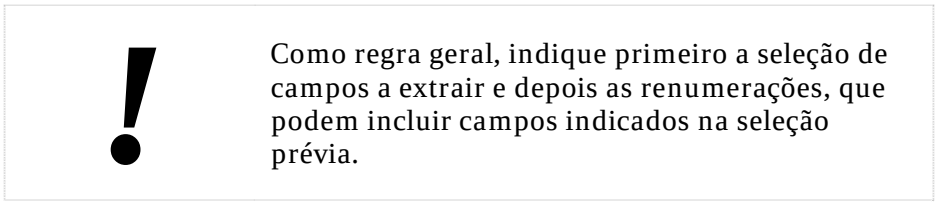
Observe que são produzidas duas ocorrências do campo 32001, correspondentes a cada chave gerada pelo registro de origem, e após cada uma estão os campos extraídos para cada chave da base alternativa.

Também não foi especificado o destino, portanto, as modificações só serão vistas na tela.

A mesma saída poderia ser produzida com:

```
mx cds join=AUTOR,1/3,100:1=mhu,(v70/)
```

Onde o campo 1 é incluído no intervalo 1/3, que depois é renumerado para 100.



É importante ter em conta que estes campos passam a ser campos da base de dados. Para eliminá-los pode-se utilizar o parâmetro *proc*. Por exemplo:

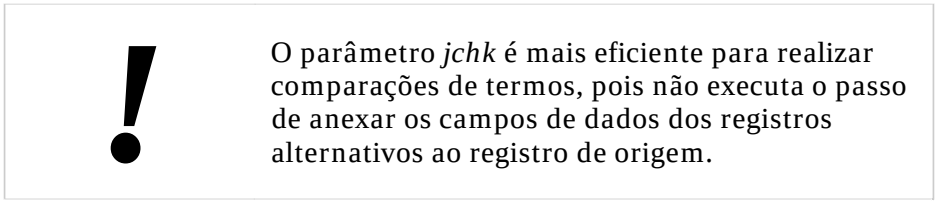
```
mx cds proc='d32001' copy=cds -all now
```

- Produzir uma listagem de autores não válidos:

```
mx CDS join=AUTOR=mhu,(v70/) pft=@check.pft –all now
```

O arquivo check.pft tem as seguintes especificações de formato:

```
if p(v32001) then (if a(v32001^m) then mfn,x2,v32001^k | Não existe| / fi)
fi
```



O parâmetro *jchk* é mais eficiente para realizar comparações de termos, pois não executa o passo de anexar os campos de dados dos registros alternativos ao registro de origem.

Join por número de registro

É possível fazer um *join* por número de MFN. Neste caso a especificação de formato deverá incluir, no início, a cadeia de caracteres *mfn* seguida pelo número de registro que deve ser recuperado.

O procedimento de *join* por MFN não requer a existência de um arquivo invertido.

Exemplos:

- Supondo que o arquivo chamado NUMS contenha uma lista com os números de registros que se quer extrair da base CDS.

```
mx seq=NUMS join=CDS='mfn=',v1 create=EXPORT now -all proc='d1/1d32001'
```

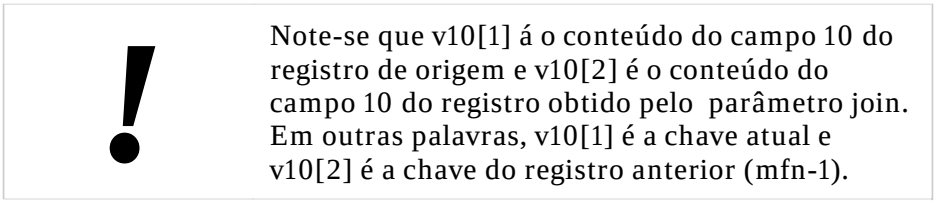
A base resultante EXPORT tem dois campos que não estão presentes nos registros de CDS. O campo 1 que provem do arquivo NUMS, onde cada linha foi considerada um registro de um só campo, e o campo 32001 que é gerado para registrar o procedimento do *join*. Estes dois campos são eliminados no final com o procedimento *proc*.

Supondo que o arquivo mestre chamado SORTED contenha seus registros ordenados de acordo com o campo 10 (não repetitivo).

```
mx SORTED "join=SORTED=if mfn > 1 then 'mfn=',f(mfn-1,1,0) fi"
pft=@umavez.pft –all now
```

A seguinte especificação de formato imprime de uma só vez os diferentes conteúdos do campo 10:

```
if v10[1] <> v10[2] then v10[1]/ fi
```



Note-se que v10[1] á o conteúdo do campo 10 do registro de origem e v10[2] é o conteúdo do campo 10 do registro obtido pelo parâmetro join. Em outras palavras, v10[1] é a chave atual e v10[2] é a chave do registro anterior (mfn-1).

Parâmetro [jmax=<n>]

O parâmetro *jmax*=<*n*> limita a quantidade de registros que são juntados ao registro original, procedentes da ação do parâmetro *join*, para cada chave gerada. Este limite se aplica para cada parâmetro *join* da linha de comandos.

Se forem especificados mais de um *jmax*, só será válido o último deles, por isso se recomenda indicá-lo apenas uma vez e depois detalhar todos os *joins* e *jchks* necessários para o processo. Por default o valor de *jmax* é infinito.

É possível usar até um máximo de 128 parâmetros join e jchk em conjunto.

Comparar Bases de dados com arquivos invertidos

jchk=<if>[+<stwfile>]=<upkey_fmt>}

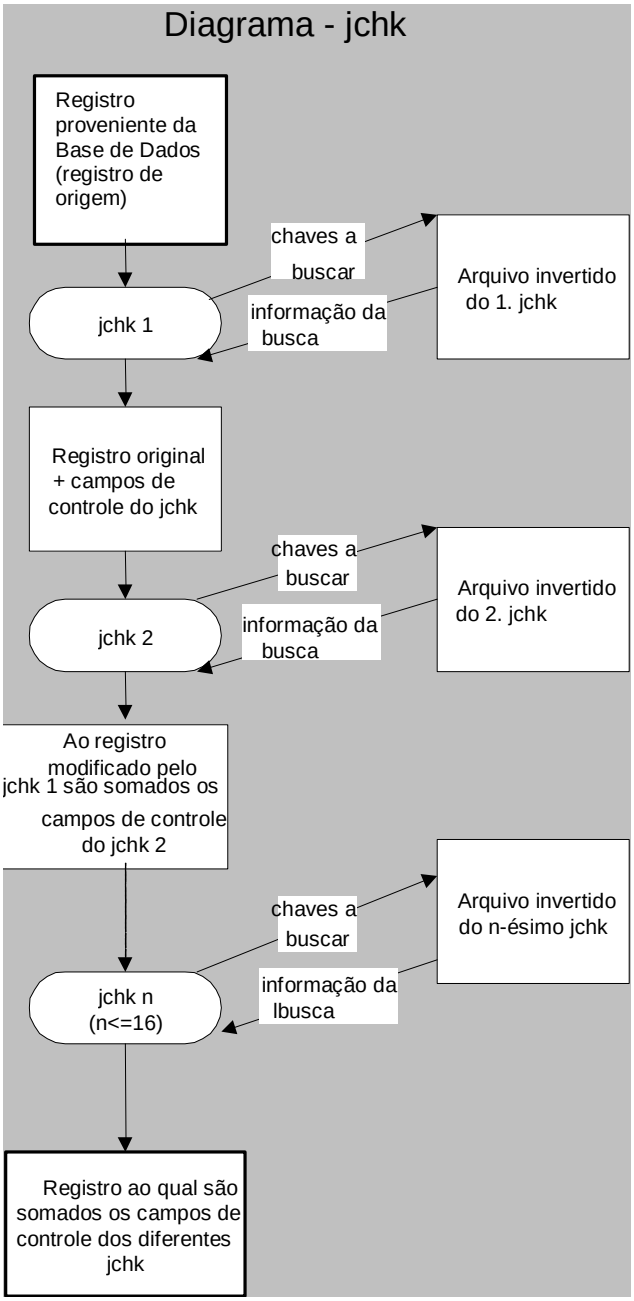
O parâmetro *jchk* é usado para comparar registros (provenientes de um arquivo mestre, um ISO-2709 ou um arquivo ASCII) com os termos de um arquivo invertido.

Os seguintes argumentos podem ser passados ao parâmetro *jchk*:

ARGUMENTOS DO PARÂMETRO *jchk*	
Argumento	**Descrição**
`<if>`	Um arquivo invertido alternativo no qual serão buscadas as chaves extraídas do registro que está sendo processado.
`[+<stwfile>]`	Uma lista de palavras a descartar (*stopword file*), opcional. O uso desta opção faz com as chaves sejam extraídas de acordo com a técnica de indexação *palavra por palavra* (TI 4) e que cada chave gerada seja filtrada pelo arquivo de *stopwords*.

ARGUMENTOS DO PARÂMETRO *jchk*	
Argumento	**Descrição**
<upkey_fmt> @<file>	Formato com o qual se lê o registro de entrada para extrair chaves com as quais serão buscados os termos no arquivo invertido. Pode ser um formato explícito ou a referência a um arquivo externo.

O seguinte diagrama mostra como se realizam as chamadas ao *jchk*:



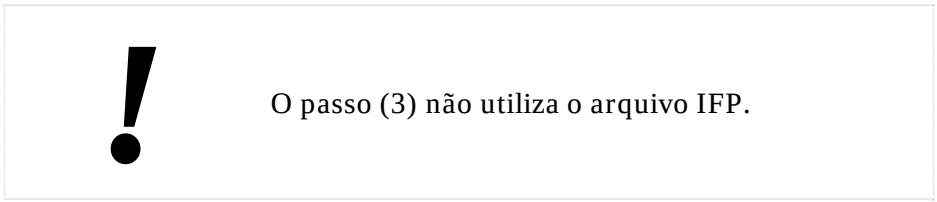
Para os exemplos será usada a mesma base *AUTOR* que foi utilizada com o *join*.

Exemplo

```
mx CDS jchk=AUTOR=mhu,(v70/)
```

A execução do *jchk* é composta pelos seguintes passos:

EXECUÇÃO DO *jchk*	
EXPLICAÇÃO	**NO EXEMPLO**
Pega um registro original da base de entrada.	Um registro de cds
Executa o formato indicado como argumento do *jchk* sobre o registro de origem e considera cada linha produzida por este formato como uma chave de consulta. O modo de formato MHU, ou MPU, é necessário, em geral, para obter as chaves, devido ao arquivo ISISUC.TAB padrão converter as minúsculas em maiúsculas para a geração do arquivo invertido	Formato: mhu,(v70)/ Registro de origem: registro que provem da base cds.
Para cada chave obtida no passo anterior, faz busca no arquivo invertido alternativo.	Arquivo invertido alternativo: AUTOR
Agrega ao registro original de entrada um campo (repetitivo) 32001 para o primeiro *jchk*, 32002 para o segundo *jchk*, etc. O campo 32001 terá tantas ocorrências quantas chaves distintas tenham sido produzidas no passo 2.	Ao registro que está sendo processado são acrescentados os campos de controle.



Se a chave estiver presente no arquivo invertido, *jchk* assume que é uma chave válida e no passo (4) anexará na saída, no campo 32001, um subcampo ^m com o valor 1, caso contrário o campo 32001 não terá um subcampo ^m.

Exemplo:

O exemplo produzirá a saída que é mostrada a seguir.

```
mx cds jchk=AUTOR=MHU,(v70/)
```

```
mfn= 1
44 «Methodology of plant eco-physiology: proceedings of the Montpellier
Symposium»
50 «Incl. bibl.»
69 «Paper on: <plant physiology><plant transpiration><measurement and
instruments>»
24 «Techniques for the measurement of transpiration of individual plants»
26 «^aParis^bUmesco^c-1965»
30 «^ap. 211-224^billus.»
70 «Magalhaes, A.C.»
70 «Franco, C.M.»
32001 «AUTOR^l2^kMAGALHAES, A.C.^o1^m1»
32001 «AUTOR^l2^kFRANCO, C.M.^o2^m1»
..
```

Conteúdo dos campos de controle (32001, 32002, etc.)

O campo 32001 e os sucessivos são gerados para registrar o resultado do processo de *jchk*, e é composto dos seguintes subcampos:

Exemplo:

```
32001 «AUTOR^l2^kFRANCO, C.M.^o2^m1»
```

SUBCAMPOS DO CAMPO 32001	
Campo	**Conteúdo**
	Primeiro dado, sem indicador de subcampo: nome do arquivo invertido da base alternativa;
^l[1/2]	Subárvore do arquivo invertido de onde foi extraída a chave, 1 para termos até 10 caracteres de tamanho e 2 para termos entre 11 e 30;
^k	Chave com a qual se fez a busca;
^o	Número de ocorrências do campo do registro original do qual procede a chave.
^m1	Se a chave existir, é criado um subcampo *m* que contem: "1". Se a chave não existir no arquivo invertido alternativo, o subcampo ^m não é criado.

Exemplo:

1. Produzir uma listagem de autores inválidos:

```
mx CDS jchk=AUTOR=mhu,(v70/) pft=@check.pft –all now
```

O arquivo *check.pft* tem as seguintes especificações de formato:

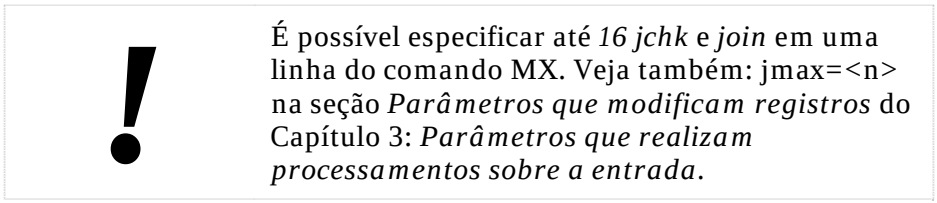
```
if p(v32001) then (if a(v32001^m) then mfn,x2,v32001^k | No existe| / fi)
fi
```

Note-se que este exemplo é igual ao realizado com *join* porém, como não é necessário realizar um *ref* para cada termo e recuperar o registro, o tempo de execução é menor.

Vantagens do *jchk* em relação ao *join*

A execução é muito mais rápida, devido ao fato que não é necessário acessar cada registro; é suficiente saber que o termo existe.

Outra vantagem é que *jchk* não usa o arquivo .IFP. No caso de distribuir arquivos de autoridade (*authority files*) não é necessário entregar esse arquivo, o que implica numa economia de espaço em disco.



É possível especificar até *16 jchk* e *join* em uma linha do comando MX. Veja também: jmax=<n> na seção *Parâmetros que modificam registros* do Capítulo 3: *Parâmetros que realizam processamentos sobre a entrada*.

Tabelas para conversão de caracteres

convert=ansi

O parâmetro *convert=ansi* converte os caracteres ASCII decimais para caracteres alfanuméricos definidos para este procedimento em uma tabela interna. Por default CISIS usa caracteres ASCII definidos para a versão IBM PC, de acordo com o código de página instalado no computador.

Tabelas para definição de caracteres alfanuméricos

[actab=<file> |ansi]

Através do parâmetro *actab* é indicada a tabela de códigos decimais ASCII para os caracteres alfanuméricos que serão considerados como parte de uma palavra. Esta

tabela é usada para a técnica de indexação (TI) palavra por palavra (TI 4, 8, 1004, 1008). *actab=ansi* utiliza uma tabela interna para este procedimento.

Se não for indicado este parâmetro, CISIS usa, de forma predeterminada, a tabela padrão correspondente do CDS/ISIS (ISISAC.TAB), definida para a versão IBM PC.

Tabelas para conversão de caracteres alfabéticos para maiúsculas

[uctab=<file>|ansi]

Através do parâmetro *uctab* é indicada a tabela de conversão dos 256 caracteres ASCII para seus correspondentes em maiúsculas.

Se não for indicado este parâmetro, CISIS usa, de forma predeterminada, a tabela padrão correspondente do CDS/ISIS (ISISUC.TAB), definida para a versão IBM PC. *uctab=ansi* utiliza uma tabela interna para esta conversão.

Tabela de Seleção de Campos - geração de chaves - fst

fst[/h]={<fst>|@[<file>]} [stw=@[<file>]]

ln{1|2}=<out_file> [+fix[/m]]

O parâmetro *fst* extrai as chaves do registro de acordo com a especificação da tabela *FST* e o arquivo opcional *stopword*. Se forem indicados os argumentos ***ln1*** e/ou ***ln2***, as chaves geradas são guardados nestes arquivos. Se não forem indicados estes argumentos, as chaves são acrescentadas como novos campos ao registro que está sendo processado, com *tag* igual ao identificador da linha de *FST* com a qual foram extraídos.

No caso de existirem ocorrências com tal *tag*, são adicionadas como novas ocorrências do campo.

Exemplo:

```
mx cds "fst=24 4 v24"
mfn= 1
24 «Techniques for the measurement of transpiration of individual plants»
26 «^aParis^bUmesco^c-1965»
30 «^ap. 211-224^billus.»
70 «Magalhaes, A.C.»
70 «Franco, C.M.»
24 «TECHNIQUES^m1^o1^c1^l1»
24 «FOR^m1^o1^c2^l1»
24 «THE^m1^o1^c3^l1»
24 «MEASUREMENT^m1^o1^c4^l2»
24 «OF^m1^o1^c5^l1»
24 «TRANSPIRATION^m1^o1^c6^l2»
24 «OF^m1^o1^c7^l1»
24 «INDIVIDUAL^m1^o1^c8^l1»
24 «PLANTS^m1^o1^c9^l1»
```

Conteúdo dos subcampos:

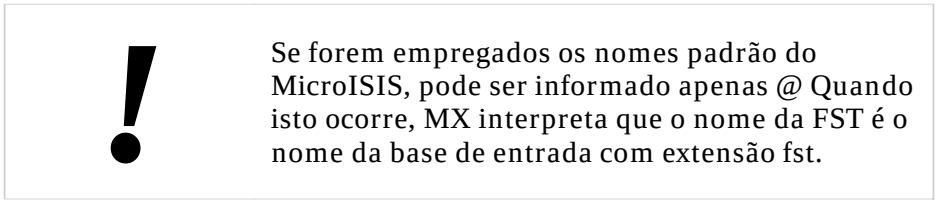
Subcampo	Conteúdo
CHAVE	Extraída com a FST indicada.
^m	MFN do qual é extraída a chave.
^o	Ocorrência do campo onde aparece esta chave.
^c	Seqüência desta chave dentro do campo.
^l[1/2]	Arquivo onde será armazenada a chave: 1 para termos de até 10 caracteres de tamanho, 2 para termos de 11 a 30 caracteres de tamanho.

Referência a uma tabela de seleção de campos externa

fst[/h]=@[<file>]

Pode-se referenciar um arquivo externo no lugar de escrever a FST na linha de comando:

```
mx CDS fst=@newcds.fst
```



Exemplo:

```
mx CDS fst=@cds.fst
```

É equivalente a:

```
mx CDS fst=@
```

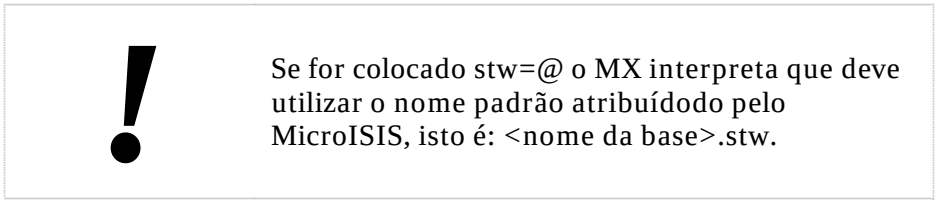
O parâmetro /h permite que o formato de extração das chaves não esteja limitado a 30 (60) caracteres. Todas as chaves são geradas no campo 33000, gerando uma ocorrência deste campo por cada linha de extração da FST, não por ocorrência dos dados originais.

Arquivo de palavras não significativas (stopwords)

stw=@[<file>]

Através de um arquivo de palavras não significativas (*stopwords*) é evitada a geração de termos não significativos (pronomes, preposições, etc.).

Um arquivo de palavras não signifcativas é um arquivo texto ASCII com uma palavra por linha.

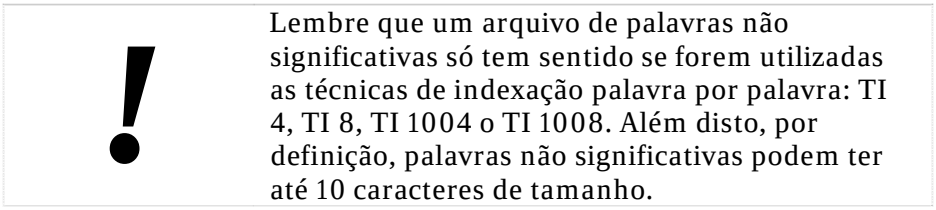


Exemplo:

```
mx cds fst=@cds.fst stw=@
```

É equivalente a:

```
mx cds fst=@cds.fst stw=@cds.stw
```



Lembre que um arquivo de palavras não significativas só tem sentido se forem utilizadas as técnicas de indexação palavra por palavra: TI 4, TI 8, TI 1004 o TI 1008. Além disto, por definição, palavras não significativas podem ter até 10 caracteres de tamanho.

Exemplo:

O seguinte exemplo utiliza um arquivo de *stopwords*.

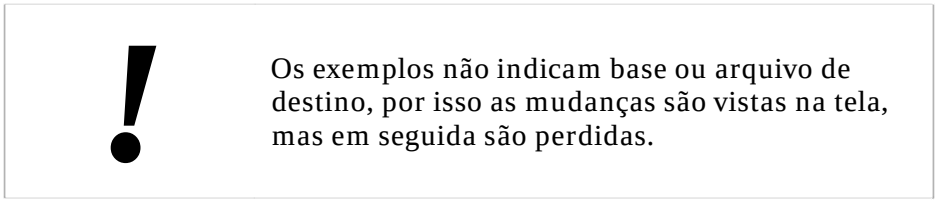
O exemplo supõe a existência de um arquivo de *stopwods* (cds.stw*)* com o seguinte conteúdo:

```
A
AN
AND
AS
BY
FOR
FROM
IN
INTO
ITS
OF
ON
THE
TO
UPON
WITH
```

Linha de comando:

```
mx CDS "fst=24 4 v24" stw=@cds.stw
Mfn= 1
24 «Techniques for the measurement of transpiration of individual plants»
26 «^aParis^bUmesco^c-1965»
30 «^ap. 211-224^billus.»
```

```
70 «Magalhaes, A.C.»
70 «Franco, C.M.»
24 «TECHNIQUES^m1^o1^c1^l1»
24 «MEASUREMENT^m1^o1^c4^l2»
24 «TRANSPIRATION^m1^o1^c6^l2»
24 «INDIVIDUAL^m1^o1^c8^l1»
24 «PLANTS^m1^o1^c9^l1»
..
```



Os exemplos não indicam base ou arquivo de destino, por isso as mudanças são vistas na tela, mas em seguida são perdidas.

Técnicas de indexação 1-8 / 1000 - 1008

A Interface CISIS aceita todas as técnicas padrão de indexação (TI) do MicroISIS, (TI 0 a TI 8), às quais se somam oito novas técnicas (TI 1000 - TI 1008).

As técnicas TI 1000...1008 geram uma chave composta de:

```
<id>   1000+IT   ‘/’, <mfn_fmt>, ’/’,  <fmt>
```

1. Caráter delimitador de início.
2. O MFN.
3. Um caráter delimitador de término (o mesmo que foi utilizado para início).
4. A chave normal extraída de acordo com IT 0 a 8 que corresponda.

Exemplos:

```
1 1004  ‘|’, f(l->idif(id.10),1,0), ’|’, (v83^*)
1 1004  ‘|’, v98 ’|’, v83
```

Como resultado desta técnica é possível controlar por programa o componente MFN do *posting* que é gerado.

Exemplo:

1. Supondo que, no campo 999 do primeiro registro da base cds se encontre o valor 23, e que o registro 2 não tenha campo 999. Então a linha:

```
mx CDS to=2 "fst=1 1000 '/', if p(v999) then v999 else mfn
fi,'/',,,(|AU=|v70/) "
```

Gera a saída:

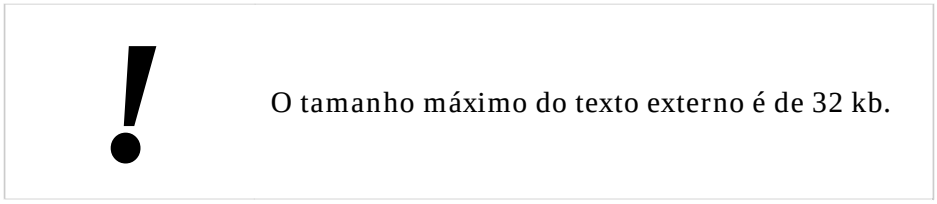
```
mfn= 1
44 «Methodology of plant eco-physiology: proceedings of the Montpellier
Symposium»
50 «Incl. bibl.»
69 «Paper on: <plant physiology><plant transpiration><measurement and
instruments>»
24 «Techniques for the measurement of transpiration of individual plants»
26 «^aParis^bUmesco^c-1965»
30 «^ap. 211-224^billus.»
70 «Magalhaes, A.C.»
70 «Franco, C.M.»
999 «23»
1 «AU=MAGALHAES, A.C.^m23^o1^c1^l2»
1 «AU=FRANCO, C.M.^m23^o1^c2^l2»
..
mfn= 2
44 «Methodology of plant eco-physiology: proceedings of the Montpellier
Symposium»
50 «Incl. bibl.»
69 «Paper on: <plant evapotranspiration>»
24 «<The> Controlled climate in the plant chamber and its influence upon
assimilation and transpiration»
26 «^c1965»
30 «^ap. 225-232^billus.»
70 «Bosian, G.»
1 «AU=BOSIAN, G.^m2^o1^c1^l2»
..
```

Observe que no registro MFN=1 os *postings* para os autores são indicados como procedentes do registro MFN=23, enquanto o registro MFN=2 gera um *posting* padrão

Exemplo

- Indexar as palavras de um texto externo associado a um registro

```
1 1004 '/',mfn,'/' proc('Gload=arquivo.txt'=v1)
```



Geração de arquivos de ligações (links)

ln1=<out_file>/ ln2=<out_file>

Se forem acrescentados os argumentos *ln1* e/ou *ln2* ao Parâmetro *fst* , serão gerados os arquivos de ligações (*links*) correspondentes.

Se desejar que os arquivos tenham as extensões padrão do CDS/ISIS deverão ser indicadas expressamente. No CISIS os arquivos podem ter ou não as extensões padrão de ISIS (pft, fst, ln1, ln2, etc.).

Exemplos:

```
mx CDS "fst=24 4 v24" ln1=cds.ln1 -all to=10 now
mx CDS fst=@cds.fst ln2=cds.ln2 -all to=10 now
mx CDS fst=@otra ln1=cds.ln1 ln2=cds.ln2 -all to=10 now
mx CDS fst=@otra ln1=otro.1 ln2=otro.2 -all to=10 now
```

Arquivos de ligações de tamanho fixo

Opción +fix[/m]

A opção *+fix[/m]* gera arquivos .ln[1/2] de tamanho fixo, não padrão em CDS/ISIS, que só poderão ser processados com utilitários do CISIS. Os arquivos de tamanho fixo requerem mais espaço em disco, mas podem ser ordenados em menor tempo.

A saída padrão de *CDS.LN1*, reconhecida pelo *CDS/ISIS* é:

1 24 1 1 TECHNIQUES
1 24 1 8 INDIVIDUAL
1 24 1 9 PLANTS

A saída especial de tamanho fixo tem a estrutura:

```
CHAVE          MFN    TAG   OCC     CNT
     mx CDS "fst=24 4 v24" ln1=cds.ln1 +fix
```

Produzirá o seguinte arquivo:

```
TECHNIQUES    1        24     1        1
INDIVIDUAL    1        24     1        8
PLANTS        1        24     1        9
```

Com a opção *+fix/m* são gerados arquivos .ln[1/2] de tamanho fixo somente com a CHAVE e o MFN do *posting*.

Os arquivos de ligações com CHAVE e MFN, sem os componentes TAG e OCC, são indicados para aplicações que manipulam mais de um arquivo invertido, gerados para um mesmo arquivo mestre.

Saída

Execução de programa externo

sys[/show]={<sys_fmt_spec> | @<file>}

O parâmetro *sys=* provoca a execução de um comando do sistema operacional, o qual deve ser especificado como uma instrução de formato. O resultado deste formato deve produzir comandos válidos para o sistema operacional.

Cada linha gerada pelo formato será executada como um comando do sistema operacional.

Exemplo:

Supondo que no campo 777 dos registros da base de dados conste o nome de um arquivo texto (alltext1.txt) que se quer mostrar, e que contem texto ASCII (o texto completo de um trabalho, por exemplo):

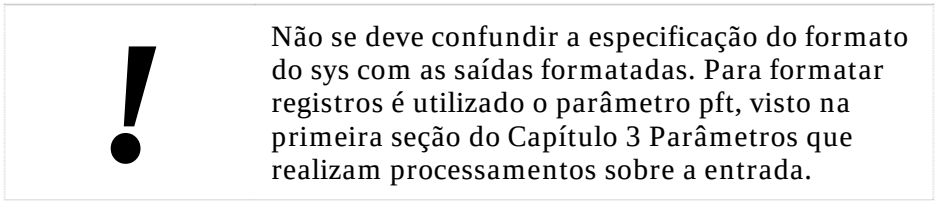
Acrescentar ao registro 1 da base cds o campo 777 com conteúdo alltext1.txt

```
mx cds to=1 proc='a777#alltext1.txt#’  copy=cds
```

Criar um arquivo alltext1.txt com algum conteúdo

Executar o MX indicando que mostre o arquivo alltext1.txt com o parâmetro *sys*

```
mx cds "sys=if p(v777) then 'more 'v777 fi"
```



Não se deve confundir a especificação do formato do sys com as saídas formatadas. Para formatar registros é utilizado o parâmetro pft, visto na primeira seção do Capítulo 3 Parâmetros que realizam processamentos sobre a entrada.

O exemplo seguinte está errado:

```
mx cds "sys=if v24:'water' then mfn/  ‘dir *.mst'  fi"
```

Deve ser indicado como:

```
mx cds "pft=if v24:'water' then mfn/ fi"  "sys=if v24:'water' then ‘dir
*.mst ' fi"
```

Opção */show*

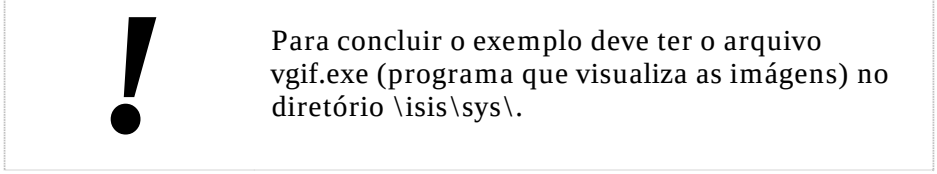
A opção */show* paralisa a execução do comando indicado em *sys*, mostra o comando que vai ser executado, e espera a intervenção do operador para seguir adiante.

mx cds "sys/show=if v24:'water' then ‘dir *.mst' fi" to=20 now

Uma aplicação do parâmetro *sys* poderia ser executar um programa externo com os dados de um registro para visualizar, por exemplo, uma imagem cujo nome de arquivo está armazenado em um campo.

Exemplo:

```
mx cds -all "sys=|\isis\sys\vgif.exe |v100^a"
```



Para concluir o exemplo deve ter o arquivo vgif.exe (programa que visualiza as imágens) no diretório \isis\sys\.

Parâmetros que criam/modificam bases de dados

{create | copy | append | merge | updatf}=<out_dbn>

Criação de um arquivo mestre

create=<out_dbn>

O Parâmetro *create* cria e inicializa incondicionalmente um arquivo mestre com o nome atribuído em <out_dbn>, arquivo mestre no qual serão guardados os registros resultantes do processo.

> **!** Se existir uma base de dados com o mesmo nome, será apagada e reinicializada sem consultar previamente o operador, perdendo assim todos os seus dados.

Exemplos:

- O seguinte exemplo lê os registros 10 a 20 da base cds e os grava na base cds2;

```
mx cds from=10 to=20 create=cds2 now -all
```

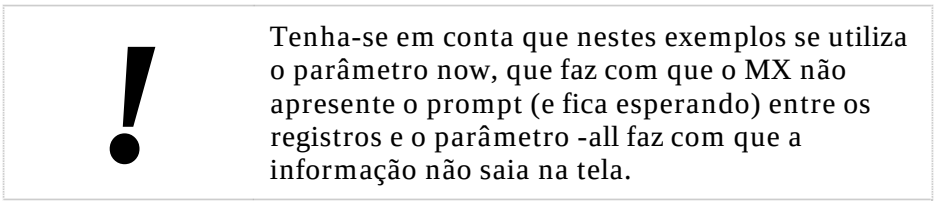
> **!** Os registros 1 a 9 da base cds2 aparecem como fisicamente apagados (ou seja, não existem) e os registros 10 a 20 são os provenientes da base cds.

- Criar uma base com os registros resultantes de uma busca:

```
mx cds "plants and water" create=plawat now -all
```

- Criar uma base a partir de um arquivo ISO-2709 (deveria existir o arquivo cds.iso):

```
mx iso=cds.iso create=cds3 now -all
```



Copiar registros para um arquivo mestre

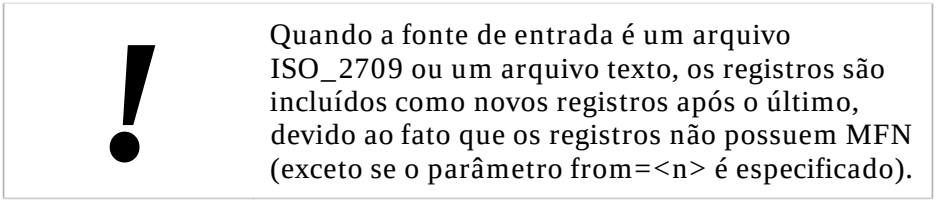
copy=<file>

O parâmetro *copy* grava os registros processados no arquivo mestre de saída <out_dbn> com o MFN que o registro que foi lido possuía. Se em <out_dbn> já existir um registro com este mfn, seu conteúdo será perdido e se o registro não existir, será criado.

Quando a base de saída (<out_dbn>) é a mesma que a de entrada, *copy* funciona como se modificasse os registros, já que estes são lidos, modificados e gravados na mesma base com o mesmo mfn.

Ao contrário do *create*, o *copy* não reinicializa a base de destino.

Se a base de destino não existir, é criada. Neste caso *copy* funciona exatamente igual a *create*.

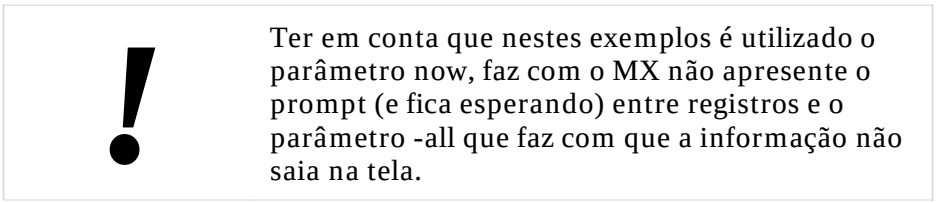


Exemplos:

- Copiar os registros 30 a 40 da base cds para a base cds2:

```
mx cds copy=cds2 now from=30 to=40 -all
```

- Copiar o resultado de uma busca para uma nova base de dados:
  ```
  mx cds water copy=cds2 now -all
  ```
- Ler registros de um arquivo ISO-2709 e guardá-los em uma base já existente:
  ```
  mx iso=cds.iso copy=cds2 now -all
  ```



Acrescentar registros a uma base de dados

append=<dbn_out>

O parâmetro *append* grava os registros processados como novos registros na base de saída após o último existente. Ao contrário de *create* e *copy*, os registros processados perdem seu número de mfn original.

Se a base de destino não existir, é criada.

Exemplos:

- Acrescentar os registros 30 a 40 da base cds à base cds2:
  ```
  mx cds append=cds2 now from=30 to=40 -all
  ```
- Acrescentar o resultado de uma busca a uma nova base de dados:
  ```
  mx cds "plants * water" append=cds2 now -all
  ```
- Ler registros de um arquivo ISO-2709 e gravá-los em uma base já existente:
  ```
  mx iso=cds.iso append=cds2 now -all
  ```

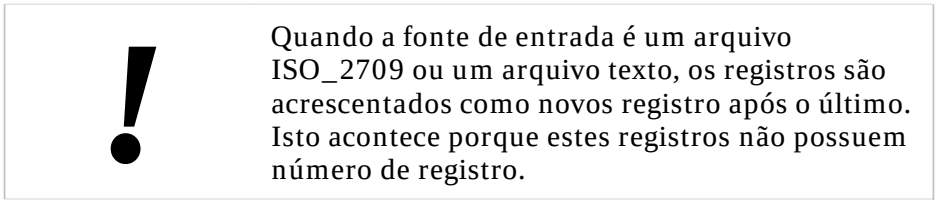
Mesclar/Intercalar registros

merge=<outdbn>

O parâmetro *merge* grava os registros processados na base de saída somente no caso de esses registros não existirem em <out_dbn>.

Supondo que na base <out_dbn> existam todos os registros entre MFN=1 e MFN=20 com exceção do registro MFN=10, e que <in_dbn> é composta de 15 registros com MFN 1 a 15. Se aplicarmos um merge, a base resultante estará composta de 20 registros dos quais o 10 é proveniente de <in_dbn> e os demais são os que já estavam em <out_dbn>.

Se a base de saída <out_dbn> não existir, é criada.



Exemplos:

- Inserir os registros de cds, com mfn entre 1 e 20, que não estejam em cds2:

```
mx cds merge=cds2 now from=1 to=20
```

- Inserir os registros resultantes de uma busca feita em cds, que não estejam em cds2:

```
mx cds merge=cds2 energy$ -all now
```

Atualização de campos

updatf=<out_dbn>

O parâmetro *updatf* (*update fields*) pega os dados do registro de entrada e substitui os campos do registro de saída, no caso de as *tags* coincidirem.

out_dbn é a base que tem os registros com campos sem atualizar e que serão regravados. O processo de atualização que MX realiza em *out_dbn* é o seguinte:

1. Lê um registro da base de entrada e executa os processos de *gizmo*, *join*, *proc*, *fst*, se foram especificados;
2. Busca um registro com o mesmo mfn em *out_dbn*, com o seguinte resultado:
   - todos os campos do registro lido de *out_dbn*, que existirem no registro lido da base de entrada, são substituídos por estes,
   - todos os campos do registro lido de *out_dbn*, que não existirem no registro lido da base de entrada, são mantidos,
3. Regrava o registro em *out_dbn*.

!	Os registros da base de entrada, eventualmente processados por outros processos do MX, devem se encontrar necessariamente em out_dbn. Quando é especificado o parâmetro updatf, não podem ser especificados outros parâmetros de saída de dados
!	O resultado da execução do parâmetro updatf não é exibido pelo MX, quer dizer, o registro que é regravado em out_dbn não é visualizado, porém é visualizado o registro que irá atualizar o registro de out_dbn.

Exemplos:

Tendo-se duas bases de dados:

INPUT	CDS
Mfn= 1 1 «campo 1 novo» 2 «campo 2 novo» 4 «campo 4 novo»	Mfn= 1 44 «Methodology of plant eco-physiology: proceedings 69 «Paper on: <plant physiology><plant transpiration> 24 «Techniques for the measurement of transpiration 26 «^aParis^bUmesco^c-1965» 30 «^ap. 211-224^billus.» 70 «Magalhaes, A.C.» 70 «Franco, C.M.» 1 «campo1» 1 «campo 1b» 2 «campo 2» 2 «campo2b» 2 «campo 2c» 3 «campo3»

O seguinte exemplo realiza a atualização de campos dos registros de CDS, tomando como fonte os registros da base INPUT.

```
mx INPUT updatf=CDS -all now
mx CDS
```

Este é o resultado de um registro da base CDS com as alterações efetuadas:

```
mfn= 1
44 «Methodology of plant eco-physiology: proceedings
69 «Paper on: <plant physiology><plant transpiration>
```

```
24 «Techniques for the measurement of transpiration
26 «^aParis^bUmesco^c-1965»
30 «^ap. 211-224^billus.»
70 «Magalhaes, A.C.»
70 «Franco, C.M.»
3 «campo3»
1 «campo 1 novo»
2 «campo 2 novo»
4 «campo 4 novo»
```

Observe que os campos 1 e 2 de CDS foram substituídos pelos procedentes da base INPUT, mas os outros campos (incluindo o campo 3) não foram modificados e, além disto, foi acrescentado o campo 4.

Gerar um Arquivo ISO_2709

[out]iso[={marc|<n>}]=<out_isofile> [outisotag1=<tag>]

O programa MX pode ler e gravar indistintamente arquivos em formato ISO-2709 e aplicar os mesmos procedimentos que em arquivos .MST/XRF (excetuando aqueles processos que requerem o uso do arquivo invertido ou dicionário).

A opção *iso=<out_isofile> ou iso=marc|<n>=<out_isofile>* permite gerar arquivos ISO-2709. Da mesma maneira que, como resultado de um processo,é gerado um arquivo mestre, pode ser criado um arquivo ISO-2709.

Com a opção *iso=<n>=<out_isofile>* pode ser indicado um tamanho fixo para as linhas. Esta opção é utilizada para intercambiar arquivos .ISO de PCs com computadores de maior porte como o HP (Minisis). O separador de campo e o separador de registro no arquivo gerado pelo MX é o caráter #. Indicando *iso=0=<filename>* as linhas dos registros serão de tamanho variável.

A opção *iso=marc=<out_isofile>* gerará linhas de saída com as seguintes características, compatíveis com a definição do MARC:

- Linhas de tamanho variável
- Os separadores de campo e registro serão os caracteres ASCII ‘\029’ ‘\030’
- As linhas terminarão somente com ‘\012’ e não incluirão o ‘\013’ (CR).

Exemplos:

- Este exemplo produz um arquivo ISO-2709 da base CDS

```
mx CDS iso=Saída.iso -all now
```

- Usa como entrada uma base de dados (CDS), realiza uma busca (plants and water) e gera como saída (cds.iso) um arquivo ISO-2709 com os registros recuperados.

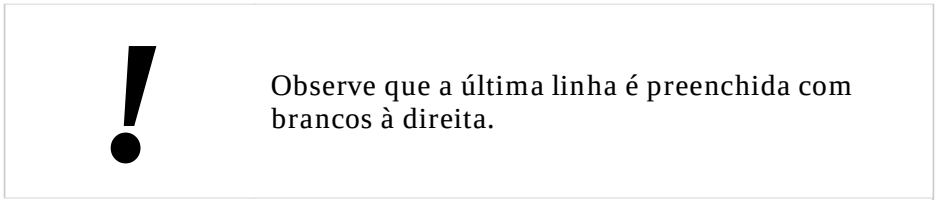
```
mx cds "plants and water" iso=cds.iso now -all
```

- Este exemplo gera um arquivo ISO-2709 com o registro 1 da base CDS, com tamanho de linha fixo=40.

```
mx cds iso=40=Saída.iso to=1 now -all
```

- Cria uma saída do tipo:

```
00409000000000109000450004400780000006900
07900078024006900157026002300226030002100
02490700016002700700013002860#Methodology
of plant eco-physiology: proceedings of
the Montpellier Symposium#Paper on: <pl
ant physiology><plant transpiration><mea
surement and instruments>#Techniques for
the measurement of transpiration of ind
ividual plants#^aParis^bUmesco^c-1965#^a
p. 211-224^billus.#Magalhaes, A.C.#Franc
o, C.M.##
```



Observe que a última linha é preenchida com brancos à direita.

Lembre que um arquivo ISO-2709 pode ser lido com MX, indicando somente que será a fonte de entrada:

```
mx iso=cds.iso
```

Gerar um Arquivo ASCII com separadores

fix=<out_file>

Exibe cada registro em uma só linha em formato ASCII puro, separando todos os campos pela barra vertical (|). Toda a informação sobre etiquetas é perdida. O tamanho máximo da linha é determinado pelo sistema operacional.

Exemplo:

- Dispõe-se de uma base de dados chamada ADMIN que contem os dados de editora, título e preço de cada livro. A instrução seguinte gera o arquivo de saída PREÇOS:

```
mx admin fix=preços now -all
```

Arquivo PREÇOS**:**

Editora 1|Titulo 1|124
Editora 1|Titulo 2|100
Editora 2|Titulo 3|89
Editora 2|Titulo 4|99
Editora 2|Titulo 5|101.50

Este arquivo pode ser importado por outros programas que admitam campos de dados separados por delimitadores. Por exemplo, pode ser importado para uma planilha eletrônica do tipo Excel para realizar cálculos e análises do tipo estatístico, econômico, etc.

Intercambiar dados do Leader do registro

[[out]isotag1=<tag>]

Os registros MARC armazenam dados nas posições 5-8 e 17-19 no cabeçalho do registro (leader). As instruções de formato do CISIS não têm acesso direto a essas posições, mas é possível converter esses bytes em campos convencionais do arquivo mestre, tanto na entrada como na exportação de registros.

No processo de entrada os dados do leader são carregados em uma série consecutiva de campos a partir de um valor base + a posição do byte do leader. Por exemplo, se for designado como base o campo 3000, então o byte 5 do leader será carregado no campo 3005. Durante a exportação, os valores destes campos, se existirem, serão movidos para os lugares correspondentes do leader. Se não houver

dado no campo, será movido um espaço em branco para a posição correspondente do leader.

Exemplo

```
mx mrclte
mfn=     1
  1  «00089048230 /AC/r91»
  3  «DLC»
  8  «891101s1990    maua   j      001  0eng  »
 10  «##^a###89048230 /AC/r91»
 20  «##^a0316107514»
 40  «##^aDLC^cDLC^dDLC»
 50  «00^aGV943.25^bB74 1990»
 82  «00^a796.334/2^220»
100  «##^aApter, David Ernest^d1919-1999»
245  «10^aMake the team.^pSoccer :^ba heads up guide to super soccer! /^cRi
J. Brenner.»
250  «##^a1st ed.»
260  «##^aBoston :^bLittle, Brown,^cc1990.»
300  «##^a127 p. :^bill. ;^c19 cm.»
3005  «c»
3006  «a»
3007  «m»
3017  «5»
3018  «i»
```

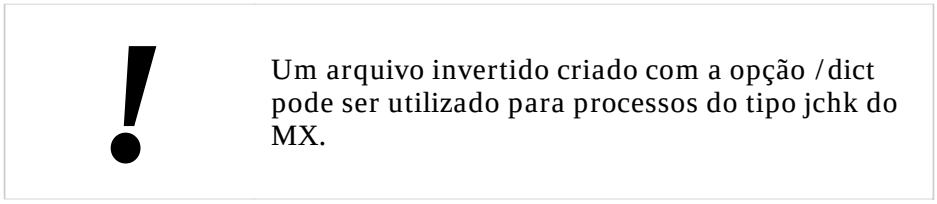
Carregar elementos gerados por uma FST

Função fullinv

Opción /dict

A opção */dict* cria apenas o dicionário do arquivo <out_inf>, não cria conteúdo no arquivo out_inf.ifp.

```
mx CDS fst=@ ifupd/create/dict=cds now from=10 to=30
```



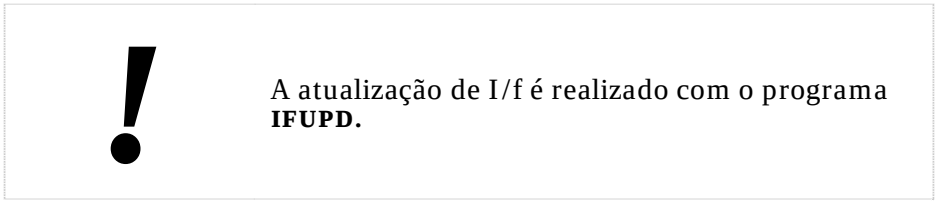
Um arquivo invertido criado com a opção /dict pode ser utilizado para processos do tipo jchk do MX.

Opción /m

Extrai as chaves (atualiza o I/F) apenas com o componente MFN do posting

Opción /ansi

Quando é indicado **ansi**, será usada uma tabela interna padrão para o conjunto de caracteres ANSI, tornando desnecessário empregar os parâmetros **actab** e **uctab**



A atualização de I/f é realizado com o programa **IFUPD.**

Tabulação de freqüência

tab[/lines:100000/width:100/tab:<tag>]=<tab_fmt>

Gera uma tabulação de freqüência do conteúdo de um campo/subcampo. O valor tabulado é a diferença com 1.000.000.000 (*high-value*), o que permite a ordenação ascendente/descendente.

Gravar o conteúdo da tag <tag> em um arquivo de nome <mfn>.<tag>

Exemplo

```
mx cds  "tab=(v26^c/)" now lw=0 | sort | more
999999949|50|1966
999999962|37|1976
999999974|25|1965
999999991|8|1973
999999994|5|1975
999999995|4|1974
999999997|2|1968
```

```
999999997|2|1971
999999997|2|27 Aug. 1976
999999998|1|11 Dec. 1975
999999998|1|14 Jume 1976
999999998|1|15 Jume 1976
999999998|1|17 Sept. 1976
999999998|1|1983
999999998|1|23 Oct. 1975
999999998|1|25 Jume 1976
999999998|1|6 Feb. 1976
```

Parâmetros de inicialização / variáveis de ambiente (setup)

Estes parâmetros permitem alterar valores que MX usa por default, tais como: o tamanho máximo de um registro, o de uma saída de formato, a definição de variáveis de ambiente ou de nomes lógicos, etc.

Arquivo de Parâmetros CISIS

cipar=<arquivo>

O arquivo CIPAR é uma ferramenta que provê as funções do SYSPAR.PAR dos <dbn>. par do CDS/ISIS padrão; além disto, agrega outras funções.

O CIPAR é um arquivo ASCII puro que – entre outras funções – relaciona nomes lógicos definidos no CIPAR aos nomes físicos (reais) dos arquivos, desta maneira se torna independente do nome real e de sua localização.

Explicação detalhada deste parâmetro no Apêndice Arquivo CIPAR.

Exemplo:

Se o arquivo *dir**filename* contém:

```
cds.*=\cisis\bases\cds.*
cds1.pft=\cisis\bases\cds1.pft
```

O comando:

```
mx cipar=\dir\filename cds pft=@cds1.pft
```

É equivalente a:

```
mx \cisis\bases\cds pft=@\cisis\bases\cds1.pft
```

Tamanho máximo de um registro

mfrl=<n>

Este parâmetro determina o tamanho máximo que um registro pode chegar a ter na leitura, processamento e gravação, incluindo os *join proc* e *fst*. O valor predeterminado para MX é 32767 bytes, em CDS/ISIS para MS-DOS é de 8.192 bytes e em Winisis é 64Kb

É necessário levar em conta que quando os registros são processados pelos diferentes procedimentos internos do MX (como *join* ou *proc*) podem chegar a ter tamanhos transitórios superiores a esse limite.

Exemplos:

```
mx mfrl=32000 CDS a$ join=AUTHORS=mhu,(v70/) create=OUT
```

Tamanho máximo para o resultado de um formato

fmtl=<n>

Este parâmetro determina o tamanho máximo da área interna onde é armazenado o resultado de todo formato especificado (*pft*, *sys*, *join*, etc). O tamanho predefinido é por default igual ao tamanho do registro.

Exemplo:

```
mx mfrl=20000 fmtl=20000 TITLES join=ISSUES=v30 pft=@holdgins.pft
```

Parâmetros gerais

Parâmetros que controlam a saída na tela

{+ | -}{control | leader | xref | dir | fields | all }

Estes parâmetros determinam a saída que o procedimento de *dump* oferecerá na saída padrão (por default a tela), dos dados, o que é útil para revisar os arquivos físicos .MST y .XRF.

A Saída predeterminada É *+fields*, que implica na saída sem formato de todos os campos do registro.

Os registros apagados são incluídos no procedimento de *dump* e é indicado com a mensagem [DELETED].

Parâmetro +

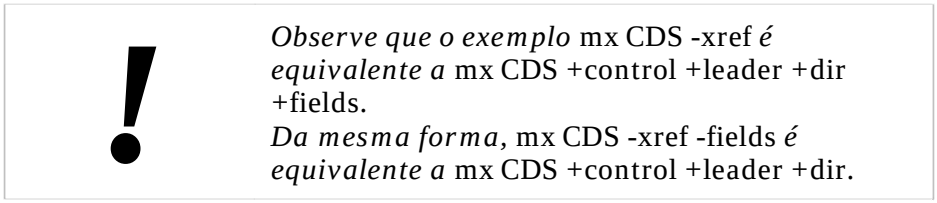
Ativa a opção associada ao procedimento de saída na tela. Na primeira vez que é empregado, cancela as opções predefinidas.

Parâmetro -

Desativa a opção associada ao procedimento de saída na tela. Na primeira vez que é empregado, ativa todas as outras opções predefinidas.

Exemplos:

```
mx CDS +xref
mx CDS -xref
mx CDS +control +leader +dir
mx CDS +all
```



Os Parâmetros se referem às seguintes partes de .MST e .XRF, às quais não se tem acesso direto pelo CDS/ISIS padrão:

Opção	Visualiza
control	Registro de controle do MST, identificado como MFN=0.
leader	Segmento de tamanho fixo de 18 bytes no começo de cada MFN.
dir	Diretório dentro do registro que contém os índices para os campos de dados.
fields	Os dados contidos nos campos do registro.
xref	Conteúdo do arquivo .XRF.
all	Ativa ou desativa todas as opções anteriores.

Para entender melhor a utilidade destes parâmetros é imprescindível conhecer a estrutura dos registros ISIS.

Um registro com estrutura ISIS tem duas características especiais que oferecem uma grande versatilidade para o manejo de informação textual: campos repetitivos e de tamanho variável.

Visto que os registros não têm um tamanho predeterminado, nem os campos têm um tamanho fixo, nem uma quantidade predeterminada de ocorrências, não é possível ter acesso direto a nenhuma porção de dados dentro da base.

O acesso ao registro se faz de modo indireto, através de ponteiros em um arquivo auxiliar com extensão .XRF, e dentro do registro os dados são acessados através de ponteiros em um diretório.

O arquivo .XRF contem toda a informação necessária para encontrar o ponto de início do registro solicitado dentro do .MST.

Para mais detalhes veja Apêndice: Estrutura dos registros de uma base ISIS.

Parâmetros para ambientes multiusuários

Opções de processo

[mono | mast | full]

Se uma base de dados é atualizada em um ambiente multiusuário deve ser indicado antes o Parâmetro *mast* ou *full*. O valor *mono* (monousuário) é o valor por default.

Modo monousuário: mono

É o modo por default, não é verificada consistência nem acesso aos dados. Supõe um único usuário.

Este parâmetro corresponde ao parâmetro 14=0 do Syspar de CDS/ISIS

Acesso limitado aos dados: mast

Este Parâmetro permite de forma simultânea a recuperação e atualização do arquivo mestre. Deve se certificar que o arquivo invertido não seja atualizado.

Este parâmetro é correspondente ao parâmetro 14=2 do Syspar de CDS/ISIS

Acesso completo: full

Quando este parâmetro está presente, pode-se recuperar e modificar dados simultaneamente, tanto no arquivo mestre como no invertido. Por outro lado, o sistema se torna mais lento.

Este parâmetro é correspondente ao parâmetro 14=1 do Syspar de CDS/ISIS

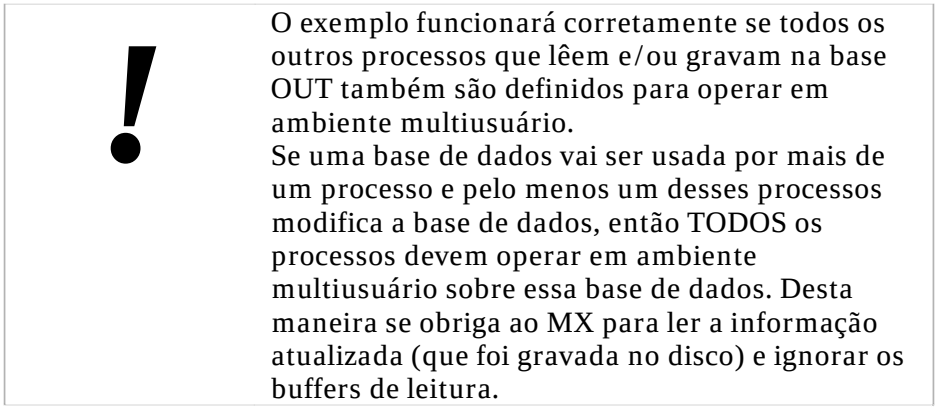
Exemplos:

- Colocar em ambiente multiusuário duas bases de dados, CDS e OUT:

```
mx mast CDS from=120 append=OUT now -all
```

Colocar em ambiente multiusuário só uma base de dados. A base CDS estará em ambiente monousuário, enquanto que a base OUT estará em ambiente multiusuário.

```
mx CDS from=120 mast append=OUT now -all
```



Outros Parâmetros

Delimitadores

?

O sinal de igual usado como delimitador pode ser substituído pelo ponto de interrogação ?

```
mx CDS from=10     es correcto
mx CDS from?10     es equivalente al anterior
```

Indicadores (*prompts*) predeterminados

p1|p2=<prompt>

É possível mudar os indicadores (*prompt*s) predeterminados do MX, durante uma sessão de MX.

```
Parâmetro p1=<prompt1>    muda .. para <prompt1>
Parâmetro p2=<prompt2>   muda -> para <prompt2>
```

Exemplo:

Pode-se mudar os prompts de tal maneira que entre um registro e outro MX diga: pressione <enter> para próximo registro ou <x> para sair. (*prompt1*) e quando acabarem os registros para visualizar apresente a mensagem: pressione <x> para sair ou digite uma expressão de busca (*prompt2)*.

```
mx cds "p1=<enter> para próximo registro <x> para sair" "p2=pressione <x>
para sair os digite uma expressão de busca:" plants
```

Visualização de um dos registros da busca (não o último):

```
mfn= 13
24 «Experience with three vapour methods for measuring water potential in
plants»
26 «^c1965»
30 «^ap. 369-384^billus.»
44 «Methodology of plant eco-physiology: proceedings of the Montpellier
Symposium»
69 «Paper on: <plant physiology><water><pressure><measurement and
instruments>»
70 «Barrs, H.D.»
70 «Slatyer, R.O.»
50 «prueb»
<enter> para proximo registro <x> para salir
```

Visualização do último registro da busca:

```
mfn= 68
24 «Some important animal pests and parasites of East Pakistan»
26 «^c1966»
30 «^ap. 285-291^billus.»
44 «Scientific problems of the humid tropical zone deltas and their
implications: proceedi+»
50 «Incl. bibl.»
69 «Paper on:
<pests><parasites><biology><ecology><plants><agriculture><public healt»
70 «Yosufzai, H.K.»
presione <x> para salir o tipee uma expressão de busca:
```

Parâmetro *trace*

trace

O parâmetro *trace* ativa a saída de informação interna de alguns processos.

Parâmetro *mfrl*

mfrl

O parâmetro *mfrl* mostra o maior tamanho de registro que foi processado até o momento.

Exemplo:

```
mx CDS now -all mfrl
MFN=1 -> MFRmfrl=408
MFN=3 -> MFRmfrl=450
MFN=21 -> MFRmfrl=452
MFN=27 -> MFRmfrl=822
MFN=104 -> MFRmfrl=980
```

MX: código de retorno de execução

MX termina a execução com valor de saída 1 ou 0, dependendo se ocorre ou não algum erro. Este código de retorno pode ser usado para o controle de execução de processamentos em lotes (*errorlevel* no MS-DOS).

Utilitários do arquivo mestre

MXF0 - Programa

O programa MXF0 analisa todos os registros de um arquivo mestre, produzindo informação sobre seus campos e sobre a freqüência dos caracteres.

O resultado da execução do MXF0 é um arquivo mestre com um registro que contem a seguinte informação:

- Nome da base de dados, data, hora, quantidade de registros, quantidade de registros ativos, quantidade de registros apagados logicamente e quantidade de registros apagados fisicamente.
- Para cada *tag* de campo distinto encontrado nos registros de entrada apresenta uma ocorrência de um campo repetitivo que contem: a etiqueta (*tag*), a freqüência, total de ocorrências, tamanho maior e menor e a quantidade total de caracteres.
- Um campo repetitivo com uma ocorrência para cada caráter diferente encontrado nos dados de entrada, seu código hexadecimal e a quantidade de vezes que aparece.

Este programa utilitário pode ser usado para produzir uma lista de etiquetas dos campos presentes em uma base de dados. Serve também para verificar se:

- campos obrigatórios estão presentes,
- campos não repetitivos ocorrem mais de uma vez,

- campos de tamanho fixo (como as datas normalizadas) têm o tamanho correto, etc.

MXF0 - Apresentação

A seguinte linha processa o arquivo mestre *cds* localizado no diretório \cisis\cds, reinicializa o arquivo mestre *x* (no diretório atual) e armazena o resultado no primeiro registro, como é indicado a seguir:

```
mxf0 \cisis\cds\cds create=x 0
```

O registro resultante na base x é:

```
mfn=     1
1001  «cds»
1003  «20051014 09:44:40 Fri»
1009  «      149»
1010  «      149»
1011  «        0»
1012  «        0»
1013  «      150»
1020  «^t024^d      148^o      148^l    6^u  179^n      9589»
1020  «^t025^d        5^o        7^l    2^u   22^n        55»
1020  «^t026^d      147^o      148^l    6^u  101^n      2897»
1020  «^t030^d      146^o      146^l    6^u   36^n      2484»
1020  «^t044^d       80^o       80^l   22^u  112^n      7525»
1020  «^t050^d       99^o       99^l   11^u  209^n      2043»
1020  «^t069^d      148^o     1134^l    3^u   36^n     16318»
1020  «^t070^d      121^o      163^l    8^u   35^n      2493»
1030  «^tall^x20^n      4516»
1030  «^tall^x22^n         8»
1030  «^tall^x27^n        29»
...
1030  «^tall^xa1^n        10»
1030  «^tall^xa2^n        10»
..x
```

Este registro informa que o arquivo mestre \cisis\cds\cds tem 149 registros ativos, e 8 campos de dados diferentes: etiquetas 24, 25, 26, 30, 44, 50, 69 e 70.

O campo com tag 24 (título) ocorre uma vez em cada um dos 149 registros (a ocorrência mais curta é de 6 caracteres de tamanho, a mais longa 179, em conjunto estas ocorrências somam 9589 bytes).

O campo 70 (autor) falta em 27 registros, e há no total 163 ocorrências.

Considerando todos os campos de dados de \cisis\cds\cds, o caráter de código 20 (caráter de espaço em hexadecimal, em ASCII decimal: 2x16 + 0 = 32) ocorre 4.516 vezes. Por outro lado, o caráter de código *a1* (i com acento agudo, em ASCII decimal: 10x16+1=161) em hexadecimal, só 10 vezes.

MXF0 - Sintaxe

```
mxf0 <dbname> [create=]<dbnout> [<tnrecs>] [noedit] [tell=<n>]
```

Os Parâmetros devem vir na ordem indicada, por exemplo:

```
mxf0 cds create=x noedit 0
```

dará erro, a forma correta de colocar é:

```
mxf0 cds create=x 0 noedit
```

Parâmetros obrigatórios

<dbname> <dbnout>

Nome do arquivo mestre de entrada

<dbname>

Arquivo mestre sobre o qual se realizará a análise

Nome do arquivo mestre de saída

<dbnout>

Arquivo mestre de saída é o arquivo mestre que contem a tabela gerada. Se já existir um arquivo mestre com este nome, os registros são acrescentados.

Criar arquivo mestre de saída

create=<dbnout>

O arquivo mestre de saída é criado. Se existir um arquivo mestre com este nome, toda sua informação se perde.

Quantidade aproximada de registros

<tnrecs>

Número de registros do arquivo mestre de entrada. No caso de desconhecê-lo, se especificado, deve indicar zero. Quando o parâmetro é diferente de zero, o programa finaliza a execução com código de retorno igual a zero somente se a quantidade de registros lidos for igual ao valor do parâmetro.

Parâmetros opcionais

[noedit] [tell=<n>]

Eliminação de espaços em branco

noedit

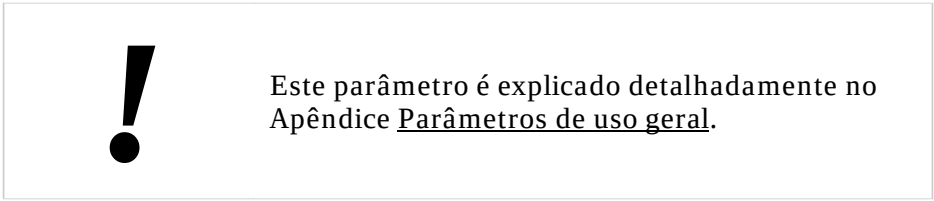
Quando está presente suprime todos os espaços à esquerda nos campos de dados de saída.

Quando este parâmetro não está presente (valor por default), o programa utiliza tamanho fixo para os campos, preenchendo com brancos à esquerda.

Informação sobre a execução do processo

tell=<n>

Envia uma mensagem de estado para *stderr (standard error)* a cada <*n*> registros.



Este parâmetro é explicado detalhadamente no Apêndice Parâmetros de uso geral.

MXF0 - Saída

O registro de saída de MXFO tem os seguintes campos:

TAG	CONTEÚDO
1001	Nome do arquivo mestre de entrada.
1003	Data e hora.
1009	Quantidade de registros processados.
1010	Quantidade de registros ativos.
1011	Quantidade de registros apagados logicamente.
1012	Quantidade de registros apagados fisicamente.
1013	Próximo MFN a ser atribuído.
1020	^tTAG ^dDOCS ^oOCCS ^lMINLEN ^uMAXLEN ^nDATA BYTES
1030	^tall ^xCHRCODE ^nCHRFREQ

O campo repetitivo 1020 tem uma ocorrência para cada etiqueta de campo diferente encontrada nos registros de entrada:

SUBCAMPO	CONTEÚDO
TAG	Etiqueta.
DOCS	Quantidade de registros que tem TAG.
OCCS	Total de ocorrências
MINLEN	Menor tamanho.
MAXLEN	Maior tamanho.
DATA BYTES	Total de caracteres

O campo repetitivo 1030 tem uma ocorrência individual para cada caráter diferente encontrado na totalidade de campos de dados da entrada:

SUBCAMPO	CONTEÚDO
CHRCODE	Código do caráter, no formato hexadecimal (x00-xFF)

CHRFREQ	Freqüência do código.

MXTB - Programa

O programa MXTB permite contar o conteúdo dos campos, por exemplo, quantidade de vezes que aparece cada autor, quantidade de vezes que aparece cada descritor, quantidade de vezes que aparece um autor e uma revista (juntos), etc.

O resultado da execução de MXTB é um arquivo mestre com um registro por palavra/frase diferente encontrada. Cada registro contem, entre outros dados, cadeia de caracteres, freqüência, etc.

MXTB é um programa de tabulação para registros do arquivo mestre em múltiplas colunas, que define as chaves de tabulação através da linguagem de formatação.

À medida que os registros são lidos, o formato provido é executado e os dados resultantes são tomados como valores para as correspondentes chaves de tabulação.

Todas as n-tuplas possíveis são tabuladas e, ao final, são entregues como registros individuais do arquivo mestre junto com suas freqüências.

O processo de tabulação acontece na memória, usando uma técnica de *hash*. Isto requer uma quantidade de memória suficiente para carregar todas as n-tuplas geradas, suas freqüências, mais um espaço adicional. Quanto maior for este espaço, mais rápida será a execução (será visto com maior detalhamento quando for explicado o parâmetro *class* mais adiante neste capítulo).

Este programa utilitário pode ser usado para produzir tabelas acumulativas de alguns dados armazenados no arquivo mestre. Por exemplo: uma lista de autores contendo as freqüências que há em uma base de dados bibliográfica, ou uma lista destes desmembrados por ano de publicação, ou outro dado.

Opcionalmente, MXTB pega uma especificação adicional de formato para conseguir um valor para o processo de tabulação. Neste caso, a informação numérica de saída é a soma destes valores para uma tupla em particular.

MXTB - Apresentação

```
mxtb \cisis\cds\cds create=frecaut 40:(v70/)
```

Esta linha de comando grava um registro no arquivo mestre frecaut para cada ocorrência diferente do campo 70 em *\cisis\cds\cds*, como é indicado a seguir:

```
MFN 1
1 «Magalhaes, A.C.»
998 «999999997»
999 «2»
MFN 2
1 «Franco, C.M.»
998 «999999998»
999 «1»
...
```

Campos dos registros de frecaut:

TAG	CONTEÚDO
1	Cadeia de caracteres a contar, no exemplo, um autor gerado pelo formato (v70/). (categoria)
998	Máximo possível de ocorrências de uma categoria menos a freqüência encontrada. (999999999 - freqüência)
999	Quantidade de vezes que ocorre o conteúdo do campo com TAG 1 de frecaut no arquivo de entrada \cisis\cds\cds. (freqüência)

O campo 998 provê uma forma simples de realizar listagens em ordem descendente das categorias encontradas (por exemplo, com o comando MSRT freqaut 10 v998).

O MXTB também pode pegar um valor a ser tabulado externo, isto quer dizer: no lugar de somar o valor um para cada ocorrência encontrada, pode somar um valor entrado mediante um formato, inclusive este valor não precisa ser fixo, pode ser um campo da mesma ou outra base de dados, onde o conteúdo do campo 999 será a soma dos valores encontrados em cada caso.

Exemplos de uso do MXTB

Tendo entrado na base de dados o custo dos documentos, data de aquisição, e área à qual pertencem. Pede-se obter uma tabela que contenha –por exemplo– o valor investido em documentos no último ano para cada área.

Supondo que entre os dados entrados nos registros esteja o custo do documento e a editora, com MXTB facilmente pode-se obter uma tabela que contenha a editora e o montante que foi pago.

```
mxtb holdings x 40:(v26^b/) tab=v90
```

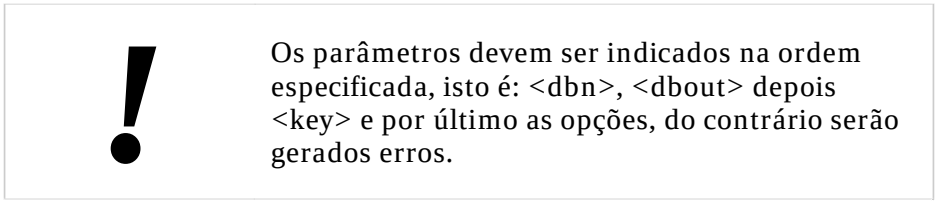
Para os primeiros 40 caracteres de cada editor distinto armazenado como subcampo ^*b* do campo *26* na base holdings, grava um registro na base x, como é indicado a seguir:

```
MFN 1
1 «Umesco Press»
998 «999993999»
999 «6000»
MFN 2
1 «ESCAP»
998 «999999969»
999 «30»
MFN 3
1 «Praeger Publishers»
998 «999999599»
999 «400»
...
```

O exemplo devolve no campo 999 a soma do conteúdo do campo v90 de cada categoria, então – no lugar de somar 1 para cada elemento encontrado – soma o que indica o campo 90. Para utilizar o parâmetro *tab* o campo 90 de todos os registros deve conter um valor numérico, do contrário os registros cujos campos 90 não contenham um valor numérico não serão considerados.

MXTB - Sintaxe

```
mxtb <dbn> [create=]<dbnout> <key> [<key> [...]] [<option> [...]]
keys:    keylen:key_fmtspec

options: {from|to|loop|count|tell|btell}=<n>
         tab=<tab_val_fmt>
         class=1000
         bool=<expr>
         {min|max}{avg|freq}=<n> -
          uctab={<file>|ansi}

Ex: mxtb in out len1:fmt1 len2:fmt2 len3:fmt3

     out = 1  key/key1_value (max len1 chars)
           2  key/key2_value (max len2 chars)
           3  key/key3_value (max len3 chars)
         998  999999999 - key_frequency
         999  key_frequency

Ex: mxtb in out len:fmt tab=Vtag

     out = 1  key_value (max len chars)
         998  999999999 - Vtag_subtotal
         999  Vtag_subtotal
```



Parâmetros obrigatórios

<dbn> <dbnout> <key>

Nome do arquivo mestre de entrada

<dbn>

Nome do arquivo mestre do qual MXTB obtém os dados para gerar a tabela.

Nome do arquivo mestre de saída

<dbnout>

o arquivo mestre de saída é o arquivo mestre que contém a tabela gerada. Se já existir um arquivo mestre com esse nome, os registros são adicionados.

Criar arquivo mestre de saída

create=<dbnout>

O arquivo mestre de saída é criado. Se existir um arquivo mestre com este nome, toda sua informação será perdida.

Chave

<key>

O parâmetro <key> é aplicado a cada uma de ate 8 chaves de tabulação, e tem a seguinte forma:

```
<tamanho da chave>:<formato que especifica a chave>
```

Tamanho máximo da chave

<keylen>

<keylen> é a quantidade máxima de caracteres retornados pela especificação de formato <key_fmtspec>. Isto significa que se a cadeia de caracteres gerada por <key_fmtspec> for maior que <keylen>, serão considerados só os primeiros <keylen> caracteres.

Formato que especifica a chave

Parâmetro <key_fmtspec>

É o formato que gera as chaves de tabulação. Por exemplo, para tabular por editor o formato deve gerar os editores, sendo um por linha.

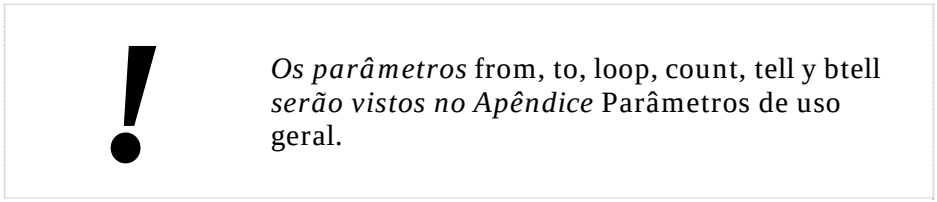
Parâmetros opcionais

bool=<exp>

{from|to|loop|count|tell|btell}=<n>

tab=<tab_val_fmt>

class=<n>



Os parâmetros from, to, loop, count, tell y btell *serão vistos no Apêndice* Parâmetros de uso geral.

Processar o resultado de uma busca

bool=<exp>

Pode-se tabular o resultado de uma busca, indicando o parâmetro *bool* e a expressão booleana. Isto permite, por exemplo, contar todos os livros com uma determinada propriedade: todas as revistas da editoria X ou todos os livros de uma área e o custo total, etc.

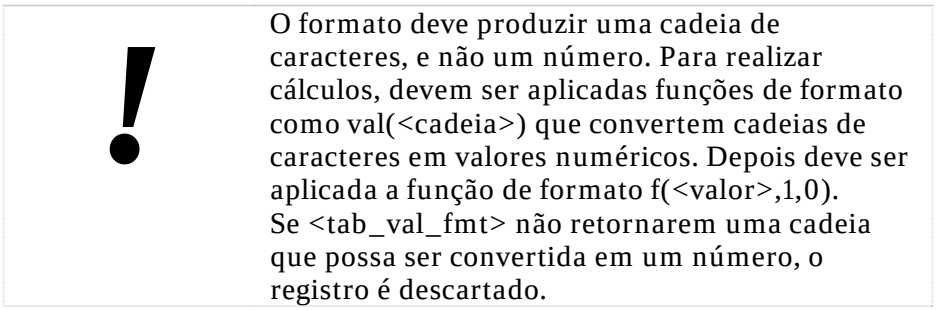
Tabulação do resultado de formato

tab=<tab_val_fmt>

O valor por default de tabulação (se não for especificado *tab*) será 1. Isto faz com que MXTB some o valor 1 a uma categoria para cada ocorrência encontrada.

tab permite alterar o valor default, mediante um formato, o qual pode especificar um valor constante (como '5'), ou pode especificar um campo de onde apanhar o valor (por exemplo, o campo preço de uma base de dados de livros).

tab permite realizar tabelas que contenham, por exemplo, o valor monetário que a biblioteca investiu em livros e periódicos para cada área.



O formato deve produzir uma cadeia de caracteres, e não um número. Para realizar cálculos, devem ser aplicadas funções de formato como val(<cadeia>) que convertem cadeias de caracteres em valores numéricos. Depois deve ser aplicada a função de formato f(<valor>,1,0). Se <tab_val_fmt> não retornarem uma cadeia que possa ser convertida em um número, o registro é descartado.

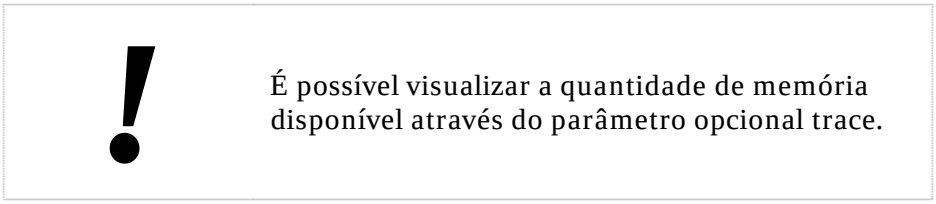
Quantidade de categorias

class=<n>

Atribui espaço para tabulação de até <n> categorias.

O valor por default para este parâmetro é 1000.

Para os efeitos de *hashing*, é recomendado especificar um parâmetro *class* com valor que esteja entre 2 e 3 vezes a quantidade de categorias esperadas.



É possível visualizar a quantidade de memória disponível através do parâmetro opcional trace.

MXTB - Saída

Os registros de Saída do MXTB são compostos pelos seguintes campos:

MFN n	
TAG	**CONTEÚDO**
1	Valor produzido por<key_fmtspec1>, até<len1> [caracteres]
2	Valor produzido por<key_fmtspec2>, até<len2> [caracteres]
3	Valor produzido por<key_fmtspec3>, até<len3> [caracteres]

998	Valor 999999999 menos o valor do campo 999
999	Freqüência para a categoria formada pelos campos 1, 2, 3, ...

Quando se usa a opção *tab*=<*tab_val_fmt*> o campo 999 contém a soma dos valores produzidos pela especificação do formato <*tab_val_fmt*> para cada registro pertencente a essa categoria.

Os registros que não geram saída para a primeira especificação <key> são descartados e, portanto, não são incluídos na tabulação.

MXCP - Apresentação

O comando:

```
mxcp \cisis\cds\cds create=x clean undelete
```

Reinicializa o arquivo mestre **x** (no diretório atual) e copia para o mesmo todos os registros do arquivo mestre cds localizado no diretório \cisis\cds.

Nenhum campo de dados no arquivo mestre **x** terá espaços em branco no começo ou no final do campo, devido ao fato que foi especificada a opção *clean*. Além disto, cada caráter que não pode ser impresso, que apareça nos campos de entrada, será convertido para um caráter de espaço na saída.

Devido ao fato que foi especificada a opção *undelete*, todos os registros logicamente apagados do arquivo mestre \cisis\cds\cds estarão ativos no arquivo mestre x.

MXCP pode converter campos em repetitivos que não são repetitivos, sempre e quando haja um caráter separador dos dados.

No exemplo seguinte, MXCP converte em repetitivo o campo com tag 70 contendo o símbolo "**;**" como delimitador de repetição de campo.

```
mxcp in create=out repeat=;,70
```

Suponhamos que, para o exemplo, os registros 1 e 2 da base in tenham no campo com tag 70:

```
MFN 1
```

```
70 «Magalhaes, A.C.; Franco, C.M.»
MFN 2
70 «Bosian, G.»
```

Então em out (o arquivo mestre resultante) os registros ficam desta forma:

```
MFN 1
70 «Magalhaes, A.C.»
70 «Franco, C.M.»
MFN 2
70 «Bosian G.»
```

Quando se usa o parâmetro *repeat*, MXCP também realiza seu processo de "limpeza".

Se não for especificada a etiqueta no parâmetro *repeat*, todos os campos presentes serão incluídos:

```
mxcp in create=out repeat=%
```

Pode-se também especificar uma lista de campos:

```
mxcp in create=out repeat=%,70,99,100,200
```

Um intervalo:

```
mxcp in create=out repeat=%,99/105
```

Uma mistura de ambos:

```
mxcp in create=out repeat=%,70,99/105,110
```

Outra característica do MXCP é a de suprimir os sinais de pontuação final. O exemplo seguinte remove o caráter de ponto final do campo 70:

```
mxcp in create=out period=.,70
```

Suponhamos que, para o exemplo, os registros 1 e 2 da base in tenham no campo com tag 70:

```
MFN 1
70 «Magalhaes, A.C.»
70 «Franco, C.M.»
MFN 2
«Bosian G.»
«xxx.»
```

Então em out (o arquivo mestre resultante) os registros ficam assim:

```
MFN 1
70 «Magalhaes, A.C»
70 «Franco, C.M»
MFN 2
«Bosian G»
«xxx.»
```

MXCP também realiza um procedimento de substituição de padrões, permitindo que os conteúdos especificados dos campos de entrada sejam substituídos à medida que são lidos.

Digitando exatamente o que é apresentado a seguir (finalizando cada linha com <enter>) é criada uma tabela de conversão para substituir os sinais de maior e menor pelo sinal por cento (%):

```
mx seq=con create=xtable -all now
<|
>|
><|%
> <|%
> <|%
Paper on: <|
<Ctrl>Z
```

Então os campos com etiqueta 69 da base cds cujo conteúdo é:

```
MFN 1
69 «Paper on: <plants><water><plant transpiration>»
MFN 2
69 «Paper on: <plant physiology><plant transpiration><measurement and
instruments»
```

Ao executar:

```
mxcp cds create=out gizmo=xtable repeat=%,69
```

Faz com que os registros com etiqueta 69 de cds sejam transformados em out da seguinte maneira:

```
MFN 1
69 «plants»
```

```
69 «water»
69 «plant transpiration»
MFN 2
69 «plant physiology»
69 «plant transpiration»
69 «measurement and instruments»
```

MXCP - Sintaxe

```
mxcp {in=<file>|<dbin>} [create=]<dbout> [<option> [...]]
options: {from|to|loop|count|tell|offset}=<n>
         gizmo=<dbgiz>[,tag_list>]
         undelete
         clean [mintag=1] [maxtag=9999]
         period=.[,<tag_list>]
         repeat=%[,<tag_list>]
         log=<filename>

 Ex: mxcp in create=out clean period=.,3 repeat=;,7

      in = 3 «   Field 3 occ 1. »
           3 «Field 3 occ 2 . »
           7 « Field 7/1;Field 7/2 ;Field 7/3.»
     out = 3 «Field 3 occ 1»
           3 «Field 3 occ 2»
           7 «Field 7/1»
           7 «Field 7/2»
           7 «Field 7/3.»
```

Os parâmetros presentes devem estar na ordem estipulado, primeiro <dbin> depois <dbout> e por último as opções, do contrário será gerado um erro.

Nome do arquivo mestre de entrada

<dbin>

Arquivo mestre de entrada.

Nome do arquivo mestre de saída

<dbout>

No arquivo mestre de saída é onde serão copiados os registros provenientes de *dbin*.

Se *dbout* não existir, será produzido um erro; para criá-lo, deve-se utilizar o parâmetro *create,* que é explicado a seguir.

Criar arquivo mestre de saída

create=<dbnout>

Quando o arquivo mestre de saída não existe, pode ser criado através deste parâmetro.

Se *dbout* existir todos os seus dados serão perdidos, já que a base de dados de saída é reinicializada antes de copiar os registros.

Parâmetros opcionais [option]

{from|to|loop|count|tell|offset}=<n>

undelete

gizmo=<dbgiz>[,tag_list>]

period=.[,<tag_list>]

repeat=%[,<tag_list>]

clean [mintag=1] [maxtag=9999]

log=<filename>

Os parâmetros from, to, loop, count, tell *e* offset *serão vistos no Apêndice* I: Parâmetros de uso geral.

Recuperar registros apagados

undelete

Recupera registros do arquivo mestre (ativa os registros da entrada que estão logicamente apagados).

```
mxcp cds newcds undelete
```

Substituição global de padrões

gizmo=<dbgiz>[,<tag_list>]

Aplica um procedimento gizmo aos dados de entrada, usando o arquivo mestre *dbgiz* como tabela gizmo; pode ser aplicado a todos os campos de um registro ou só aos campos especificados em <tag_list>.

A lista <tag_list> pode ser especificada das seguintes maneiras:

- Uma etiqueta.
- Uma lista de etiquetas separadas por vírgula.
- Um intervalo de etiquetas na forma *valor inferior / valor superior*.
- Uma combinação destas opções.

Se for especificado mais de um gizmo=<dbgiz>[,<tag_list>], o segundo é executado sobre o registro que resulta do procedimento do primeiro gizmo, e assim sucessivamente.

```
mxcp cds newcds gizmo=xtable
mxcp cds newcds gizmo=xtable,24,50/70
```

O parâmetro gizmo é amplamente explanado no Apêndice Parâmetros de uso geral.

Supressão de caracteres de pontuação

```
period=<char>[,<tag_list>]
```

MXCP provê a possibilidade de suprimir o caráter <*char*> como sinal de pontuação final; pode ser aplicado a todos os campos de um registro ou só aos campos presentes em <tag_list>.

A lista <tag_list> pode ser especificada das seguintes maneiras:

- Uma etiqueta
- Uma lista de etiquetas separadas por vírgula
- Um intervalo de etiquetas na forma *valor inferior / valor superior*.
- Uma combinação destas opções.

```
mxcp cds newcds period=;
mxcp cds newcds period=;,24,50/60,70
```

Converter campos em repetitivos

repeat=<char>[,<tag_list>]

Converte em repetitivos os campos de entrada que têm <*char*> como delimitador; pode ser aplicado a todos os campos de um registro ou só aos campos presentes em <tag_list>.

A lista <tag_list> pode ser especificada das seguintes maneiras:

- Uma etiqueta.
- Uma lista de etiquetas separadas por vírgula.
- Um intervalo de etiquetas na forma *valor inferior / valor superior*.
- Uma combinação destas opções.

```
mxcp cds newcds repeat=/
mxcp cds newcds repeat=;,24,50/60,70
```

Supressão de espaços em branco

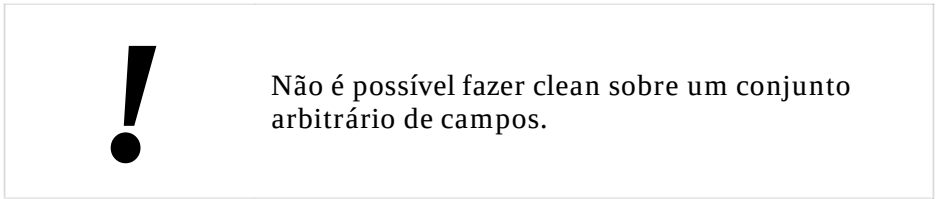
clean

Suprime todos os espaços em branco no início e no final do campo e substitui todos os caracteres não imprimíveis por espaços em branco.

Se for usado *repeat* ou *period* a opção *clean* é iniciada automaticamente.

A opção *clean* é aplicada a todos os campos do registro.

```
mxcp cds newcds clean
```



Eliminação de campos por tag

mintag=<n>

Apaga todas as ocorrências dos campos com tag menor do que <n>. Por default n=1

```
mxcp cds newcds mintag=10
 maxtag=<n>
```

Apaga todas as ocorrências dos campos com tag maior do que <n>. Por default n=9999

```
mxcp cds newcds maxtag=70
```

Registro de eventos

log=<filename>

Direciona as mensagens de controle e estado para o arquivo <*filename*>.

MXCP - Saída

MXCP não produz saídas de registros vazios. Os registros de entrada ou os processados que não tenham campos são descartados. Os registros logicamente apagados só são processados, se for usada a opção *undelete*.

Pode ser usado o mesmo arquivo mestre tanto para a entrada como para a saída, porém só será possível reorganizar um arquivo mestre, copiando-o a um novo. A opção *create=<dbout>* assegura que <*dbout*> será criado o reinicializado.

O comando:

```
mxcp in create=out period=.,3 repeat=;,7
```

Para o arquivo mestre in*:*

```
MFN 1
3 «Field 3 occ 1. »
3 «Field 3 occ 2 .»
```

```
7 «Field 7/1; Field 7/2; Field 7/3.»
MFN 2
...
```

Cria *out* com:

```
MFN 1
3 «Field 3 occ 1»
3 «Field 3 occ 2»
7 «Field 7/1»
7 «Field 7/2»
7 «Field 7/3»
MFN 2
...
```

Produz as seguintes mensagens:

```
*** mfn 1 tag=3/1 -> rejected char
*** mfn 1 tag=3/1 -> rejected char
*** mfn 1 tag=3/1 . -> rejected char
*** mfn 1 tag=3/2 -> rejected char
*** mfn 1 tag=3/2 . -> rejected char
*** mfn 1 tag=3/2 -> rejected char
*** mfn 1 tag=7/1 -> rejected char
```

MSRT - Programa

O programa MSRT ordena um arquivo mestre de forma ascendente, seguindo as chaves de ordenação geradas por determinada especificação de formato.

Depois que um arquivo mestre é ordenado, o registro com mfn=1 contém a chave de ordenação mais baixa, o registro com mfn=2 a segunda chave de ordenação mais baixa, e assim sucessivamente.

MSRT - Apresentação

O comando:

```
msrt \cisis\cds\cds 60 mhu,v24
```

ordena o arquivo mestre cds localizado no diretório \cisis\cds de acordo com os primeiros *60* caracteres do campo *24* (transformado apropriadamente para maiúsculas).

O comando:

```
msrt mxtb_out 9 v998
```

ordena o arquivo mestre mxtb_out de acordo com os valores armazenados no campo 998 (presumindo que é de tamanho fixo e com zeros à esquerda).

Enquanto que o comando:

```
msrt mxtb_out 9 f(999999999-val(v999),9,0)
```

ordena o arquivo mestre mxtb_out em ordem descendente, de acordo com o valor numérico armazenado no campo 999.

MSRT - Sintaxe

```
msrt <dbname> <keylen> <keyfmt> [-mfn] [tell=<n>]
```

Os Parâmetros presentes devem estar na ordem estipulado, isto é <dbname>, <keylen>, depois <keyfmt> e, por último, as opções, do contrário será produzido um erro.

Parâmetros obrigatórios:

<dbname> <keylen> <keyfmt>

Nome do arquivo mestre de entrada

<dbname>

Arquivo mestre a ser ordenado em seu próprio master.

Tamanho máximo da chave

<keylen>

Quantidade máxima de caracteres a comparar.

Geração da chave

<keyfmt> | tag=<n>

Formato que, aplicado aos registros, gera as chaves de ordenação.

Para especificar um campo de dados como chave de ordenação, sem a necessidade de executar um formato, o parâmetro <keyfmt> deve ser **tag=<n>**.

Parâmetros opcionais

-mfn

+del

tell=<n>

O parâmetro tell é explicado detalhadamente no Apêndice Parâmetros de uso geral.

Manter MFNs originais

-mfn

Conserva os *MFN* originais. Por default, os registros são renumerados, depois de ordenados.

Eliminar chaves iguais

+del

Elimina as chaves iguais de registros consecutivos, depois de ordenados.

MSRT - Saída

Quando um arquivo mestre é ordenado, o programa MSRT atualiza somente o .xrf, substituindo as entradas correspondentes de cada par de registros que estão sendo ordenados.

No final deste processo, a menos que se use a opção -mfn, o arquivo .mst é atualizado, atribuindo os números de registro ao campo mfn do leader do registro.

Esta atualização se realiza no mesmo lugar dos registros, o que significa que os registros são modificados em suas próprias localizações, portanto, a organização do arquivo mestre processado não muda. Desta maneira se poupa o tempo que se gasta com a gravação no arquivo mestre de todos os registros ordenados.

RETAG - Programa

O programa *RETAG* substitui a tag dos campos do arquivo mestre, segundo uma tabela de renumeração dada. Também pode realizar um desbloqueio (*unlock*) do arquivo mestre.

As operações que RETAG realiza são realizadas no mesmo registro, o que significa que os registros são modificados nas localizações originais e, portanto, a organização do arquivo mestre processado não muda.

RETAG - Apresentação

O comando:

```
retag \cisis\cds\cds xtable
```

processa todos os registros ativos do arquivo mestre cds localizado no diretório \cisis\cds, reescrevendo o segmento do diretório dos registros correspondentes, de acordo com uma tabela xtable de renumeração de tag localizada no diretório atual.

Uma tabela de renumeração é um arquivo seqüencial que tem uma linha para cada tag a ser renumerado, como indicado a seguir:

```
24 240
70 700
69 690
```

O comando anterior muda a tag 24 para 240, a tag 70 para 700, e a tag 69 para 690.

Neste exemplo, só são regravados os registros que contenham, pelo menos, uma ocorrência das tags 24, 70 ou 69.

Além disso, o programa *RETAG* pode ser usado para deixar sem efeito os bloqueios do arquivo mestre.

O comando:

```
retag \cisis\cds\cds unlock
```

em vez de especificar uma tabela de renumeração de etiquetas, especifica a palavra chave unlock, que deixa sem efeito os *exclusive write lock* ou *data entry locks* e todos os *record locks* existentes no arquivo mestre cds localizado no diretório \cisis\cds.

RETAG - Sintaxe

```
retag <dbname> {<retag.tab>|unlock} [<option> [...]]
```

Parâmetros obrigatórios

<dbname> <retag.tab>

Arquivo mestre de entrada

<dbname>

Arquivo mestre a ser reetiquetado ou desbloqueado (*unlocked*).

Tabela de renumeração

<retag.tab>

Nome de um arquivo seqüencial que provê uma tabela de renumeração, ou a palavra chave *unlock*.

Se <retag.tab> é *unlock*, isto indica ao programa que deve ser realizado um desbloqueio do arquivo mestre.

Uma tabela de renumeração tem o formato:

```
<tag> <new tag>
```

Pode conter até 5.461 entradas na versão padrão.

Parâmetros opcionais

from=<n>

to=<n>

tell=<n>

shift=<n>

Os parâmetros from, to, shift e tell *são vistos no Apêndice* Parâmetros de uso geral.

RETAG - Saída

As operações RETAG são realizadas no mesmo registro físico (*in-place*), não importando se foi solicitada uma reetiquetagem ou um desbloqueio.

Na operação de reetiquetagem, só se regrava o segmento do diretório dos registros processados.

Na operação de desbloqueio, só é regravado o registro de controle do arquivo mestre e o segmento *leader* dos registros processados.

CTLMFN - Programa

O programa CTLMFN exibe e atualiza o registro de controle de um arquivo mestre.

Deve ser usado antes de executar o programa MKXRF para restaurar todos os registros ativos em um arquivo mestre reinicializado logicamente, para estabelecer o número seguinte a ser atribuído no arquivo mestre e definir a quantidade de dados a serem analisados.

CTLMFN - Apresentação

O comando:

```
ctlmfn \cisis\bases\cds
```

lê o registro de controle do arquivo mestre \cisis\bases\cds.mst e permite que cada um dos campos deste registro seja atualizado. Estes campos são:

CAMPO	CONTENIDO
nxtmfn	MFN a ser atribuído ao próximo registro a ser criado na base de dados.
nxtmfb	Último número de bloco atribuído ao mestre (o primeiro bloco é 1).
nxtmfp	Próxima posição disponível no último bloco do arquivo mestre.
mftype	Para base de dados do usuário (1 para arquivos de mensagens do sistema).
reccnt	Reservado.
mfcxx1	Reservado.
mfcxx2	Quantidade de usuários no processo de entrada de dados (*Data Entry Lock*).
mfcxx3	*Exclusive Write Lock*.

Se for informado um valor inválido para *nxtmfb*, será apresentado o número total de blocos do arquivo mestre.

Os comandos

```
ctlmfn \cisis\bases\cds
```

e

```
mkxrf \cisis\bases\cds
```

permitem mudar um ou mais campos de controle em \cisis\bases\cds.mst (registros de dados) e então criar, reinicializar e gerar o arquivo de referências cruzadas \cisis\bases\cds.xrf.

CTLMFN - Sintaxe

ctlmfn <dbname> [ok]

Nome do arquivo mestre de entrada

<dbname>

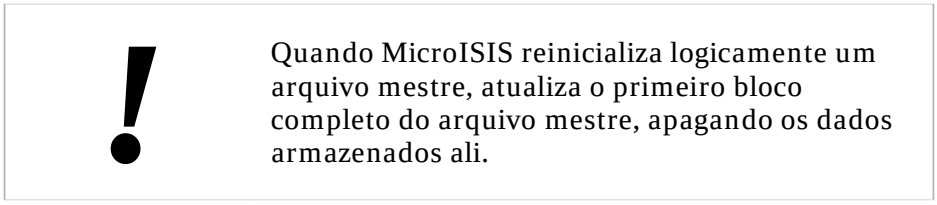
Parâmetro obrigatório que indica o arquivo mestre a ser processado.

Prompt de confirmação

ok

CTLMFN - Saída

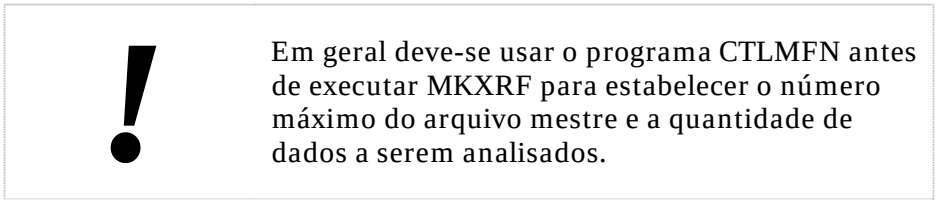
O programa CTLMFN grava somente os primeiros bytes (os bytes do registro de controle) de um arquivo .mst. Não são alterados os bytes restantes do primeiro bloco do arquivo mestre.



MKXRF - Programa

O programa MKXRF lê um arquivo .mst e cria seu arquivo .xrf correspondente.

Além disso, pode ser usado para restaurar todos os registros ativos em um arquivo mestre reinicializado logicamente.



MKXRF - Apresentação

O comando:

```
mkxrf \cisis\bases\cds
```

cria, reinicializa e grava o arquivo de referências cruzadas \cisis\bases\cds.xrf correspondente ao arquivo mestre \cisis\bases\cds.mst.

O programa MKXRF pode restabelecer um arquivo mestre reinicializado acidentalmente, sempre e quando seu registro de controle tenha os seguintes campos de controle ajustados em forma apropriada:

CAMPO	CONTENIDO
nxtmfn	MFN a ser atribuído ao próximo registro a ser criado na base de dados.
nxtmfb	Último número de bloco atribuído ao mestre (o primeiro bloco é 1).
nxtmfp	Próxima posição disponível no último bloco do arquivo mestre.

Estes valores podem ser estabelecidos usando o programa CTLMFN, como se mostra nos seguintes comandos:

```
ctlmfn \cisis\bases\cds
mkxrf \cisis\bases\cds
```

Estes comandos permitem substituir um ou mais campos de controle no arquivo \cisis\bases\cds.mst (registros de dados) para criar, reinicializar, e gerar o arquivo de referências cruzadas \cisis\bases\cds.xrf.

Descrição da execução de CTLMFN no exemplo:

a) Se *nxtmfn* é desconhecido, deverá ser estabelecido o mfn máximo *possível* (ver explicação mais abaixo); *nxtmfn* real poderá ser obtido posteriormente, executando o programa *MXF0* (campo 1013)
b) *nxtmfb* é a quantidade de blocos do arquivo mestre \cisis\bases\cds.mst que estão para serem analisados; se for escrito um valor inválido, será informado o número total de blocos.
c) *nxtmfp* é a última posição no bloco *nxtmfb* do arquivo mestre que está para ser analisado; um valor de 512 força *MKXRF* a processar todo o conteúdo do bloco nxtmfb do arquivo mestre.

MKXRF - Sintaxe

```
mkxrf <dbname>
```

Arquivo mestre de entrada

<dbname>

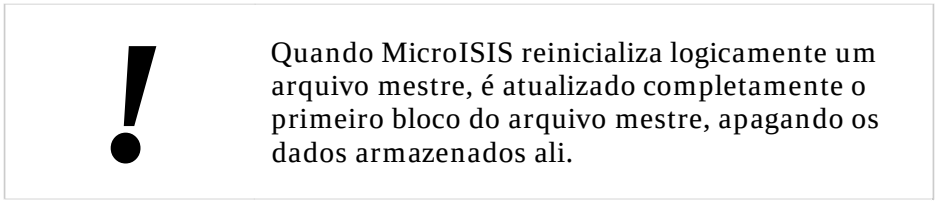
Nome do arquivo mestre a ser processado (Parâmetro obrigatório).

MKXRF – Saída

O programa MKXRF lê o registro de controle e examina o arquivo de entrada .mst até o local *nxtmfb* e *nxtmfp*; neste processo identifica se as seguintes condições de um registro do arquivo mestre se aplicam aos componentes do *leader*:

d) MFRmfn está no intervalo entre 1 e (nxtmfn-1).
e) MFRmfrl está no intervalo LEADER a MAXMFRL.
f) MFRbase é até MFRmfrl.
g) MFRmfbwb aponta para algum endereço prévio válido.
h) MFRmfbwp está no intervalo entre 0 e MSBSIZ.
i) MFRnvf é zero ou positivo.
j) MFRstatus é ACTIVE ou DELETED.
k) MFRbase é igual a LEADER + MFRnvf*sizeof(DIRSTRU).

A localização de cada registro do arquivo mestre identificado é escrito no arquivo .xrf que é criado, e inclui as versões de registros velhos.



Restaurando uma base de dados corrompida

A seguir são detalhados procedimentos para restaurar uma base de dados de acordo com seus possíveis estados de problemas:

Estado			Ação
MST	**XRF**	**Informação**	
OK	corrompido	MaxMFN desconhecido	Usar MKXRF.
Reinicializado	OK	Tamanho do XRF	Estimar maxMFN e usar CTLMFN.
Bloqueada	OK		Usar CTLMFN, informar zeros nos campos mfcxx2, mfcxx3

Cálculo aproximado do MaxMFN

Uma estimação do valor máximo de registros (*maxMFN*) se obtém de:

```
(tamanho_em_bytes_de_XRF/512) * 127
```

Se o valor de *maxMFN* estabelecido é maior do que o número real de registros, não serão gerados registros nulos no .mst, nem no .xrf. O arquivo .xrf é gravado somente para os registros que podem ser lidos no .mst. A base poderá ser lida sem problemas, mas não poderá ser atualizada (gerará o erro *fatal: recxref/read*). O *maxMFN* deverá ser corrigido seguindo uma destas duas formas:

Tomar o valor "x recs" que é indicado no final da execução de *MKXRF* e em seguida colocar x + 1 em *nxtmfn* através do *CTLMFN*.

Criar outro arquivo mestre (com *create*, não com *copy*) a partir do que se tem, a fim de corrigir o registro de controle no novo arquivo mestre.

ID2I - Programa

O programa I2ID recebe um arquivo ASCII, com determinada estrutura e gera um arquivo mestre.

ID2I - Apresentação

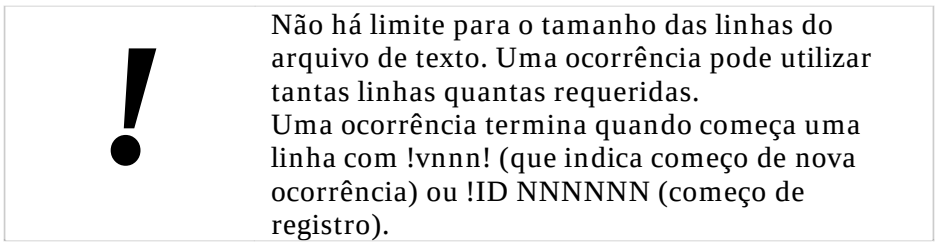
O programa recebe um arquivo ASCII e devolve um arquivo mestre.

Estrutura do arquivo ASCII:

- !ID nnnnnn Marca de começo de registro com mfn=nnnnnn
- !vnnn Marca de começo de uma ocorrência do campo com tag nnn.

O arquivo terá a forma:

```
!ID nnnnnn
!vXXX!...conteúdo da tag XXX.............
!vYYY!...conteúdo da tag YYY.............
...
!ID nnnnnj
!vXXQ!...conteúdo da tag XXQ.............
!vYYQ!...conteúdo da tag YYQ.............
...
```



Não há limite para o tamanho das linhas do arquivo de texto. Uma ocorrência pode utilizar tantas linhas quantas requeridas.
Uma ocorrência termina quando começa uma linha com !vnnn! (que indica começo de nova ocorrência) ou !ID NNNNNN (começo de registro).

Exemplo:

```
id2i x.txt create=newcds
```

A idéia é utilizar o *ID2I* junto com o *I2ID*, que pega um arquivo mestre e devolve um arquivo de texto (editável e modificável).

Através do *ID2I* o arquivo de texto modificado é convertido em um arquivo mestre:

```
I2ID cds >x.txt
```

Edit x.txt (Edita o conteúdo permitindo criar, modificar e apagar registros e ocorrências)

```
ID2I x.txt create=cds
```

I2ID é tratado detalhadamente no Capítulo 16.

ID2I - Sintaxe

```
id2i <filein> [create[/app]=]<dbout> [option [option] ... ]
options: {from|to|loop|count|offset|tell|id}=<n>
```

Parâmetros obrigatórios

<filein> <dbout>

Arquivo ASCII de entrada

<filein>

É um arquivo ASCII com a estrutura vista na apresentação.

Nome do arquivo mestre de saída

<dbout>

Arquivo mestre de saída, onde serão copiados os registros provenientes de *filein*.

Se *dbout* não existir, será produzido um erro; para criá-lo deve-se utilizar *create*. Se *dbout* existir, serão perdidos todos os seus dados, já que é reinicializada antes de copiar os registros.

Criar arquivo mestre de saída

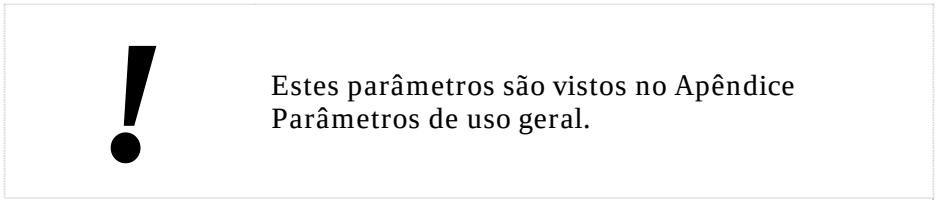
create/app=<dbnout>

Cria o arquivo mestre de saída; se *dbout* existir, serão perdidos todos os seus dados, já que é reinicializado antes de copiar os registros. A opção **/app** significa que a numeração é atribuída seqüencialmente

Além disso, não considera a informação de MFN nas marcas de começo de registros (!ID nnnnnn) e aceita marcas !ID 000000.

Parâmetros opcionais

{from|to|loop|count|tell|offset|id}=<n>



Estes parâmetros são vistos no Apêndice Parâmetros de uso geral.

I2ID - Programa

O programa I2ID recebe um arquivo mestre e gera um arquivo texto.

I2ID - Apresentação

O programa recebe um arquivo mestre e devolve um arquivo texto com a seguinte estrutura:

```
!ID nnnnnn      Marca de começo de registro com mfn=nnnnnn
!vnnn           Marca de começo de uma ocorrência do campo com tag nnn.
```

Arquivo de texto devolvido por *I2ID*:

```
!ID nnnnnn
!vXXX!...conteúdo da tag XXX.............
!vYYY!...conteúdo da tag YYY.............
...
!ID nnnnnj
!vXXQ!...conteúdo da tag XXQ.............
!vYYQ!...conteúdo da tag YYQ.............
...
```

Não há limite para o tamanho das linhas do arquivo texto. Uma ocorrência pode utilizar tantas linhas quantas requeridas.

Uma ocorrência termina quando começa uma linha com !vnnn! (indica começo de nova ocorrência) ou !ID NNNNN (começo de registro).

Exemplo:

```
I2ID cds >x.txt
```

A idéia é utilizar o *I2ID* junto com o *ID2I*. O primeiro devolve um arquivo texto (que pode ser editado e modificado) e, em seguida, por intermédio do *ID2I*, volta a converter o arquivo texto em um arquivo mestre.

```
I2ID cds >x.txt
```

edit x.txt (Edita o conteúdo, podendo-se criar, modificar e apagar registros e ocorrências)

```
ID2I x.txt create=cds
```

ID2D é tratado detalhadamente no Capítulo 15.

I2ID - Sintaxe

```
i2id <dbn> [option [option] ... ]
options: {from|to|loop|count|offset|tell}=<n>
          1record=<tag>=<value>
```

Parâmetros obrigatórios:

<dbn>

Arquivo mestre de entrada

<dbn>

Arquivo mestre a ser convertido.

Parâmetros opcionais

{from|to|loop|count|offset|tell}=<n>

Estes parâmetros são vistos no Apêndice Parâmetros de uso geral.

CRUNCHMF - Sintaxe

```
crunchmf <dbn> <target_dbn> [<option> [...]]
```

Converte o arquivo mestre (.mst e .xrf) de um sistema operacional para outro, devido à estrutura em binária não ser compatível para os diferentes sistemas. O programa detecta automaticamente o sistema em que está operando.

Opções:

```
{from|to|loop|count|tell}=<n>
target={pc|linux|hpux|sum|alpha|vax|umisys|mpe|cdc|same} default: linux
format={isis|cisisX}                                     default: isis
mstxl={0|1|2|4}                                          default: as <dbn>
```

Utilitários do arquivo invertido

IFKEYS - Programa

O programa IFKEYS exibe os termos do arquivo invertido e os *postings* totais correspondentes, opcionalmente desmembrados pelas etiquetas das quais foram extraídos.

IFKEYS aceita um intervalo de termos como parâmetro e pode ser usado para exibir um conjunto de termos do arquivo invertido.

IFKEYS - Apresentação

O comando:

```
ifkeys \cisis\bases\cds from=plant to=plants
```

lê o arquivo invertido cds localizado no diretório \cisis\bases começando no termo *plant* até o termo plants e exibe estes termos precedidos por seus números de *postings*:

```
8|PLANT
```

```
4|PLANT ECOLOGY
1|PLANT EVAPOTRANSPIRATION
1|PLANT PHOTOSYNTHESIS
20|PLANT PHYSIOLOGY
6|PLANT TRANSPIRATION
8|PLANTS
```

enquanto que o comando:

```
ifkeys \cisis\bases\cds from=plant to=plants +tags
```

produz a mesma informação e, adicionalmente, (a) inclui as tags (*Field ID*) de onde foram gerados estes termos e, (b) gera uma linha diferente para cada combinação de termo e tag, como indicado a seguir:

```
8| 24|PLANT
4| 69|PLANT ECOLOGY
1| 69|PLANT EVAPOTRANSPIRATION
1| 69|PLANT PHOTOSYNTHESIS
20| 69|PLANT PHYSIOLOGY
6| 69|PLANT TRANSPIRATION
6| 24|PLANTS
2| 69|PLANTS
```

Os parâmetros from to não distinguem maiúsculas de minúsculas, from=plant produzirá a mesma saída que from=PLANT.

IFKEYS - Sintaxe

```
ifkeys <dbname> [from=<key>] [to=<key>] [+tags] [tell=<n>]
```

Arquivo invertido de entrada

<dbname>

Arquivo invertido a ser processado.

Parâmetros opcionais

from=<term>

to=<term>

+tags

tell=<n>

O parâmetro tell é explicado detalhadamente no Apêndice Parâmetros de uso geral.

Primeiro termo a ser listado

from=<term>

Começa a listagem a partir do termo <term>.

Último termo a ser listado

to=<term>

Termina a listagem no termo <term>.

Mostrar informação sobre etiquetas (tags)

+tags

Acrescenta à listagem a tag da qual o termo foi extraído.

IFKEYS - Saída

Se for usada a opção *from=<term>* e o termo especificado de começo não existir, a listagem começará no termo seguinte do arquivo invertido. Se for empregada a opção *to=<term>* e o termo final especificado não existir, a listagem parará no termo anterior do arquivo invertido.

A saída de IFKEYS pode ser redirecionada a um arquivo para ser processado posteriormente pelo programa MX. Por exemplo, os comandos:

```
ifkeys cds +tags >x
mx seq=x "pft=if val(v1)=1 and val(v2)=24 then v3/ fi" now
```

exibem as palavras do título (campo 24 na base *cds*) que ocorrem exatamente uma vez.

IFLOAD - Programa

O programa IFLOAD carrega um arquivo invertido, usando arquivos de ligações ordenados de acordo com as opções de processamento. Sâo aceitos outros formatos, além do formato de arquivo de ligações padrão do MicroISIS.

Também permite que seja criado só o arquivo invertido do dicionário.

O programa IFLOAD pode ser usado também para criar e reinicializar um arquivo invertido.

IFLOAD – P

Apresentação

O comando:

```
ifload \cisis\bases\cds \isis\work\cds.lk1 \isis\work\cds.lk2 tell=99
```

carrega o arquivo invertido cds localizado no diretório \cisis\bases, usando os arquivos de ligações de chaves curtas e chaves longas cds.lk1 e cds.lk2, localizados no diretório \isis\work.

A opção *tell=99* produz uma mensagem progressiva a cada 99 registros de ligações processados, mostrando a chave atual que está sendo carregada.

Supõe-se que os arquivos de ligações anteriores estão no formato de arquivo de ligações padrão de MicroISIS, que têm a seguinte apresentação:

```
102 24 1 1 ABOUT
42 24 1 9 ABSENCE
6 24 1 10 ABSORPTION
87 24 1 5 ACCOUNT
136 69 1 1 ACCOUNTING
40 24 1 6 ACID
101 24 1 5 ACTION
49 24 1 6 ACTIVITIES
130 24 1 7 ACTIVITIES
23 24 1 5 ACTUAL
```

É um conjunto de registros de tamanho variável, que contêm campos que identificam a origem da chave e a própria chave.

Os primeiros 4 campos são os componentes do *posting*.

Campo	Conteúdo
MFN	Número de registro (master file record number).
TAG	Identificador de campo, atribuído pela FST (field identifier).
OCC	Número de ocorrência do campo
CNT	Número seqüencial do termo no campo

Para permitir o uso de programas de ordenação (*sort*) padrão, IFLOAD aceita arquivos de ligações com registros de tamanho fixo, como indicado a seguir:

```
ABOUT        102  24  1  1
ABSENCE      42   24  1  9
ABSORPTION   6    24  1  10
ACCOUNT      87   24  1  5
ACCOUNTING   136  69  1  1
ACID         40   24  1  6
ACTION       101  24  1  5
ACTIVITIES   49   24  1  6
ACTIVITIES   130  24  1  7
ACTUAL       23   24  1  5
```

O comando seguinte carrega o arquivo invertido cds localizado no diretório \cisis\bases, usando os arquivos de ligações de tamanho fixo cds.lk1 e cds.lk2 localizados no diretório \isis\work:

```
ifload \cisis\bases\cds \isis\work\cds.lk1 \isis\work\cds.lk2 +fix
```

Um procedimento para gerar um arquivo invertido **x** usando a Tabela de Seleção de Campos (FST) default e o arquivo *Stopword* é:

```
mx x fst=@ stw=@ ln1=x.ln1 ln2=x.ln2 +fix tell=100
del *.$$$
mys 37 x.ln1 x.lk1
del x.ln1
del *.$$$
mys 57 x.ln2 x.lk2
del x.ln2
ifload x x.lk1 x.lk2 +fix tell=1000
del x.lk1
del x.lk2
```

Quando são criados vários arquivos invertidos para um dado arquivo mestre, e não é necessária a operação de busca por proximidade, só é necessário o componente MFN do *posting*. Os arquivos de ligações poderiam estar no formato:

```
ABOUT          102
ABSENCE        42
ABSORPTION     6
ACCOUNT        87
ACCOUNTING     136
ACID           40
ACTION         101
ACTIVITIES     49
ACTIVITIES     130
ACTUAL         23
```

e carregados com o comando:

```
ifload \cisis\bases\cds \isis\work\cds.lk1 \isis\work\cds.lk2 +fix/m
```

O programa IFLOAD permite carregar somente o dicionário do arquivo invertido, usando a opção *-posts*:

```
ifload authority x.lk1 x.lk2 -posts
```

Depois que se carrega um arquivo invertido, por default, o alerta de *I/F update is pending* é reinicializado em todos os registros associados do arquivo mestre.

Quando vários arquivos invertidos são criados para um dado arquivo mestre, deve-se especificar que o alerta de *I/F update is pending* deve ser mantido, como indicado a seguir:

```
ifload au au.lk1 au.lk2 reset=0
ifload ti ti.lk1 ti.lk2 reset=0
```

Se o alerta de *I/F update is pending* devesse ser reinicializado e o nome do arquivo invertido a ser carregado fora diferente do arquivo mestre associado, deverá ser usada a opção *master*

```
Ifload kw kw.lk1 kw.lk2 master=\cisis\bases\cds
```

IFLOAD - Sintaxe

```
ifload <dbname> {<file_lk1>|void} {<file_lk2>|void} [<option> [...]]
```

Parâmetros obrigatórios

<dbname>

<file_lk1>

<file_lk2>

Arquivo invertido de entrada

<dbname>

Arquivo invertido a ser carregado o arquivo invertido <*dbname*> é criado e reinicializado inclusivo se este já existe.

Arquivo de ligações de chaves curtas

<file_lk1|void>

Arquivo de ligações de chaves curtas, ordenado.

Arquivo de ligações de chaves longas

<file_lk2|void>

Arquivo de ligações de chaves longas, ordenado.

Parâmetros obrigatórios

master=<name>

- {reset | posts | balan}

tell=<n>

+fix[/m]

Reinicializar marca de atualização pendente

master=<mst_name>

Reinicializa o alerta de *I/F update is pending* no arquivo mestre <*mst_name*>;
por default se processa para o arquivo mestre <*dbname*>.

Atualização pendente

-reset

Não reinicializa o alerta de *I/F update is pending* no arquivo mestre <dbname> ou <*mst_name*>.

Não balancear dicionário

-balan

Não rebalanceia o dicionário (as árvores B*)

Não Carregar postings

-posts

Carrega somente o dicionário. Não carrega os *postings* no arquivo *.ifp*.

Informação sobre a execução do processo

tell=<n>

Produz uma mensagem progressiva no *Standard error* cada <n> registros de ligações carregados.

Arquivos de ligações de tamanho fixo

+fix[/m]

Carregar arquivos de tamanho fixo

+fix

Carrega arquivos de ligações com registros de tamanho fixo com o formato:

```
KEY MFN TAG OCC CNT
```

Carrega arquivo de ligações com formato reduzido

+fix/m

Carrega arquivos de ligações com registros de tamanho fixo com o formato:

```
KEY MFN
```

IFLOAD - Saída

O arquivo invertido consiste de seis arquivos físicos, cinco dos quais contêm o dicionário de termos recuperáveis (organizados como uma árvore B*) e o sexto

contém a lista de postings associados com cada termo. Para efeitos de otimizar o armazenamento em disco, são mantidos duas árvores B* (estrutura de dados que permite armazenar informação classificada) separadas, uma para termos de até 10 caracteres e outro para termos de miss de 10 caracteres e até um máximo de 30 caracteres. Ambas as árvores B* estão estruturadas como páginas de tamanho fixo, com chaves completadas com espaços em branco à direita.

Pode-se obter uma otimização de espaço de disco adicional, usando o Programa Utilitário do CISIS MYZ (capítulo 21), que comprime o dicionário do arquivo invertido para cada uma das árvores B*, como indicado a seguir:

```
myz ifn 1 ifn_z tell=10
myz ifn 2 ifn_z tell=10
```

O dicionário do arquivo invertido resultante ifn_z está composto pelos arquivos *ifn_z.cnt, ifn_z.n01, ifn_z.l01, ifn_z.n02 y ifn_z.l02*.

Depois de executados os comandos o arquivo ifn.ifp deverá ser renomeado como *ifn_z.ifp*, onde ifn_z é a versão comprimida do arquivo invertido inf.

IFUPD - Programa

O programa IFUPD atualiza um arquivo invertido, de acordo com o estabelecido em:

- A Tabela de Seleção de Campos (FST);
- Stopwords;
- Os registros do arquivo mestre;
- As opções de processamento.

O programa IFUPD pode também ser usado para criar e reinicializar um arquivo invertido.

IFUPD - Apresentação

O comando:

```
ifupd \cisis\bases\cds fst=@ stw=@
```

lê o arquivo mestre cds localizado no diretório \cisis\bases e atualiza o arquivo invertido correspondente, de acordo com a Tabela de Seleção de Campos default (FST) e com os arquivos *Stopword - cds.fst* e *cds.stw* - localizados no mesmo diretório.

O comando seguinte usa uma Tabela de Seleção de Campos diferente e especifica que as chaves do arquivo invertido a serem extraídas não usam *Stopwords*:

```
ifupd \cisis\bases\cds fst=@\cisis\bases\outra.fst
```

IFUPD aceita especificações na linha de entrada, tais como:

```
ifupd \cisis\bases\cds "fst=70 0 (v70/); 69 2 v69"
```

IFUPD permite que sejam atualizados vários arquivos invertidos para um dado arquivo mestre. Neste caso, todas as atualizações (exceto a última) devem especificar que o alerta *I/F update is pending* deve ser mantido, como indicado a seguir:

```
ifupd au fst=@au.fst master=\cisis\bases\cds reset=0
ifupd ti fst=@ti.fst master=\cisis\bases\cds reset=0
ifupd kw fst=@kw.fst master=\cisis\bases\cds
```

Na última linha do exemplo não aparece reset=0 devido a que serão atualizados todos os arquivos invertidos, por tanto não é necessário manter o alerta de atualização pendente.

IFUPD - Sintaxe

```
ifupd [mono|full] [create=]<ifname> [<option> [...]]

[mono|full]                          → single/multi user operation
[create=]                            → to delete+create <ifname>
<ifname>                             → output inverted file

options:

 fst=<fstspec>|@[fstfile]            → field select table
 stw=<stwspec>|@[stwfile]            → stop words
```

```
-posts                    → (init and) do not load .ifp
master=<name>             → alternate master file
actab=<file>              → alphabetic chars table
uctab=<file>              → upper case chars table
from=<n>                  → initial mfn
to=<n>                    → final mfn
count=<n>                 → max mfns
tell=<n>                  → tell <n>% loaded
```

Parâmetros obrigatórios

<ifname>

Arquivo invertido a ser atualizado

<dbname>

Arquivo invertido a ser atualizado.

Se for especificado *create=<dbname>*, é criado o arquivo invertido *<dbname>* e reinicializado, inclusive se este já existe.

Tabela de Seleção de Campos

fst=<fstspec>|@[fstfile]

FST por default

fst=@

Usa a Tabela de Seleção de Campos (FST) por default.

Carregar FST a partir de um arquivo externo

fst=@<file>

A Tabela de Seleção de Campos (FST) é provida no arquivo <file>.

Especificação de FST em linha

fst=<fstspec>

A Tabela de Seleção de Campos (FST) é provida na linha de entrada como uma lista separada por ponto e linha.

Arquivo de palavras não significativas

stw=<stwspec>|@[stwfile]

Arquivo STW por default

stw=@

Usa o arquivo *Stopword* por default.

Carregar lista STW a partir de um arquivo externo

stw=@<stwfile>

A lista de palavras não significativas (*stopwords*) é provida no arquivo <file>.

Manter marca de atualização pendente

reset=0

Mantém o alerta de *I/F update is pending*; por default reinicializa todos os registros processados.

Não carregar postings

-posts

Atualiza só o dicionário. Não carrega os postings no arquivo *.ifp*.

Arquivo mestre alternativo

master=<name>

Por default assume como arquivo mestre um que tenha o mesmo nome que o arquivo invertido que está sendo processado, mas se for indicado *master=<name>*, *ifupd* utilizará um arquivo mestre chamado <name>.

IFUPD – Saída

Somente são processados os registros do arquivo mestre que têm o alerta *I/F update is pending*.

O parâmetro *reset=0* permite que os registros do arquivo mestre processados possam ser usados posteriormente para criar ou atualizar outro arquivo invertido.

MYS - Sintaxe

```
mys {link1|link2|<preclen>} <fileln> <filelk> [<option> [...]]

options:  tell=<n>
          +fix/m

Ex: mys 37 x.ln1 x.lk1
     mys link1 x.ln1 x.lk1 tell=0

Ex: mys 57 x.ln2 x.lk2
     mys link2 x.ln2 x.lk2 tell=0
```

MYS – Saída

Realiza uma ordenação (*sort*) do arquivo de ligações (*links*) para criar o arquivo invertido

IFMERGE - Sintaxe

```
ifmerge <out> <if1>[,n1] <if2>[,n2] [...] [<option> [...]]
 <out>                    → output inverted file
 <if1>[,n1]               → input inverted file #1, maxmfn#1
 <if2>[,n2]               → input inverted file #2, maxmfn#2

options:
+mstxrf                   → create <out> M/F and its mstxrf <out>.pft
-posts                    → do not load .ifp
-balan                    → do not rebalance the dict
tell=<n>                  → tell <n> keys has been loaded
```

IFMERGE – Saída

Combina vários arquivos invertidos, de diferentes arquivos master, em um único conjunto de arquivos invertidos, com um procedimento para recuperar os registros dos arquivos master fontes.

MKIY0 - Sintaxe

```
mkiy0 <dbn> [-ifp] [-v] [blksize=32768]
```

MKIY0 - Saída

Combina os seis arquivos que compõem o arquivo invertido em um único arquivo físico.

CRUNCHIF - Sintaxe

```
crunchif <dbn> <target_dbn> [<option> [...]]

options:
-ifp                          → don't crunch .ifp file
/ifp                          → crunch .ifp file if needed
tell=<n>                      → tell <n> records processed

target={linux|hpux|sum|alpha|vax|umisys|mpe|cdc|pc}  default: linux
```

CRUNCHIF - Saída

Converte o arquivo invertido de um sistema operacional para outro, por exemplo de Windows para Linux

Referências bibliográficas

1. UNESCO. *Mini-micro CDS/ISIS:* Reference manual (version 2.3). Organized by Giampaolo Del Bigio. Paris: Umited Nations Educational, Scientific and Cultural Organization, 1989. 286 p. ISBN 92-3-102-605-5.

2. BUXTON, Andrew, HOPKINSON, Alan. *The CDS/ISIS for Windows Handbook* [online]. Paris: Umited Nations Educational, Scientific and Cultural Organization, 2001 [cited 30 August 2006]. 164 p. Available from internet: <http://bvsmodelo.bvs.br/download/winisis/winisis-handbook-en.pdf>.

3. SUTER, Tito. “Prehistoria” e historia del MicroISIS [online]. In: *Manual para instructores de Winisis*. Buenos Aires: Centro Atómico Constituyentes (CAC), Comisión Nacional de Energía Atómica (CNEA), 1999 [citado el 30 Agosto 2006]. p. 21-26. Disponible en internet: <http://www.cnea.gov.ar/cac/ci/isis/isidams.htm>.

Glossário

- **Arquivo**. Em computação, um conjunto de dados que se pode gravar em algum dispositivo de armazenamento. Os arquivos de dados são criados por aplicações, como por exemplo, um processador de textos.

- **Arquivo invertido**. Conjunto de seis arquivos físicos, cinco dos quais contêm os termos de busca do dicionário (organizados como uma árvore B*) e o sexto contém a lista de apontadores associados a cada termo. A fim de otimizar o armazenamento em disco, são mantidas duas árvores B* em separado: uma para os termos de até 10 caracteres (armazenados nos arquivos *.N01* e *.L01*) e outra para os termos de mais de 10 caracteres (armazenados nos arquivos *.N02* e *.L02*). O arquivo *.CNT* contém campos de controle para ambas as árvores B*). Em cada arquivo árvore B* o arquivo *.N0x* contém os nodos da árvore e o arquivo *.L0x* contém as folhas. Os registros das folhas apontam para o lugar onde se encontram os apontadores que contêm a informação para localizar os registros (postings) na base de dados. Este arquivo é identificado com a extensão *.IFP*.

- **Backup**. Procedimento no qual um ou mais arquivos e/ou diretórios são duplicados para outro dispositivo de armazenamento (fita ou disco), para produzir uma copia de segurança, que pode ser restaurada no caso de algum dado seja apagado acidentalmente ou se ocorreu dano físico dos dados originais.

- **Base de dados**. Coleção de dados estruturados para que seja possível acessá-los e manipulá-los facilmente. É formada por unidades denominadas registros, cujos diversos atributos são representados por campos e subcampos. Por exemplo, em um arquivo "cadastro de clientes", cada cliente representa um registro, que possui vários campos, como "NOME", "CÓDIGO DO CLIENTE", "TELEFONE", etc.

- **Bases de dados bibliográficos**. Versão eletrônica de um catálogo ou índice bibliográfico.

- **Campo**. Elemento de um registro que permite armazenar informação específica. Ver *Base de dados*.

- **CDS/ISIS – MicroISIS**. Software desenvolvido e mantido pela UNESCO para o tratamento de dados bibliográficos.

- **Chave**. Expressão que identifica uma ou mais informações de determinada classe ou tipo e que pode ser usada na busca.

- **Formato de apresentação**. Conjunto de comandos que determinam como deve ser a saída de dados de uma base de dados ISIS.

- **Formato eletrônico**. Qualquer forma de armazenamento, recuperação e apresentação de informação passível de transmissão online ou gravação em meios magnéticos ou óticos.

- **Formato ISO (de intercâmbio de dados)**. Padrão estabelecido pela ISO para intercâmbio de dados entre instituições, redes e usuários. Refere-se à norma ISO 2709.

- **Formato LILACS**. Formato de descrição bibliográfica estabelecido por BIREME, baseado no UNISIST Reference Manual for Machine-readable Bibliographic Descriptions.

- **Indexação**. Procedimento de identificar e descrever o conteúdo de um documento com termos que representam os assuntos correspondentes desse documento, com o objetivo de recuperá-lo posteriormente.

- **Posting**. Consiste do endereço de uma chave extraída do arquivo mestre.

- **Registro**. Conjunto estruturado de dados que permite armazenar determinado assunto. Ver *Base de dados*.

- **Subcampo**. Elemento que contém a menor parte de informação de um campo, cujo sentido pode não ser claro se não for analisado em conjunto com os outros elementos relacionados.

- **UNISIST**. Programa intergovernamental relativo às cooperações no campo da informação científica e tecnológica.

Apêndice I - Parâmetros de uso geral

Relativos à saída padrão:

now, tell, -all, >, >>

Desabilitar *prompt* entre registros

now (nowait)

Este parâmetro desabilita o prompt que MX apresenta entre registros.

O parâmetro *now* ou *nowait* permite que sejam processados todos os registros sem intervenção do operador e suprime a saída do indicador *(prompt)* do programa.

Agora é obtida a saída completa de forma imediata, sem que haja parada entre os registros com o indicador de dois pontos .. (*prompt)*:

```
mx cds from=24 to=50 pft=mfn/ now
```

Informar a cada n registros

tell=<n>

O parâmetro *tell=n* produz uma breve mensagem a cada <n> registros no dispositivo de saída de mensagens de erro (*stderr*, normalmente a tela).

Mesmo que o processo seja redirecionado para uma saída impressa ou para um arquivo, as mensagens produzidas por *tell=n* não podem ser redirecionadas, continuarão saindo na tela.

Recomenda-se usar este parâmetro junto com o parâmetro *-all* que suprime a listagem de saída dos registros na tela.

Exemplo:

```
mx cds from=1 to=150 tell=30 now -all
```

Produzirá na tela do computador só as seguintes mensagens:

```
C:\>mx cds from=1 to=150 tell=30 now -all
+++ 30
+++ 60
+++ 90
+++ 120
+++ 150
C:\>
```

Desabilitar fluxo *(dump)* de informação em tela

-all

O parâmetro *-all* faz com que não seja enviado nenhum tipo de informação para a tela, salvo mensagens de erro ou informação gerada pelo parâmetro *tell* (que utiliza o *Standard error* para enviar suas mensagens).

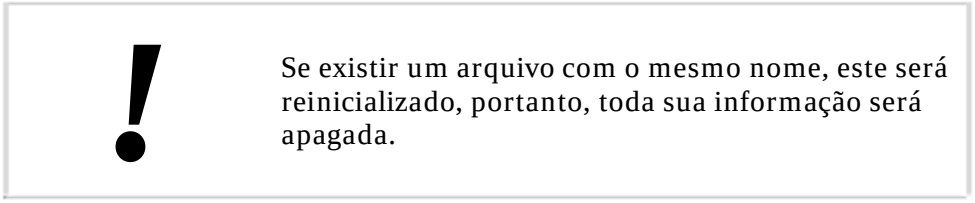
Redirecionar saída padrão

> <file> | >> <file>

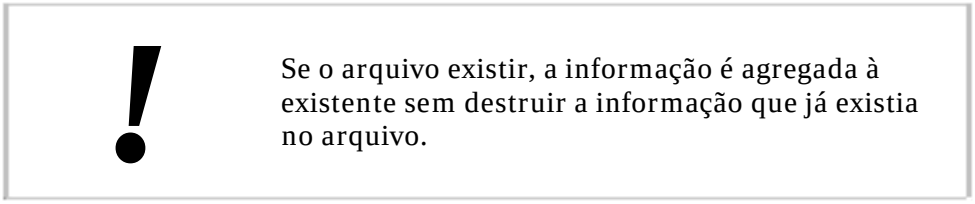
Toda a informação apresentada na tela (saída padrão por default, *Standard output*), pode ser redirecionada para um arquivo, impressora, etc.

É possível dirigir a saída para um arquivo ou para uma impressora, usando os atributos de redirecionamento do sistema operacional: > ou >>.

Se no final da linha de comandos for acrescentada a instrução **>*file***, MX cria um arquivo com nome ***file*** e guarda nele toda a informação que estava direcionada para a saída padrão (tela).



Se for colocado **>>*file*** na chamada do MX, este abre o arquivo com nome ***file*** e acrescenta toda a informação que foi dirigida à saída padrão (tela), se o arquivo não existir é criado.



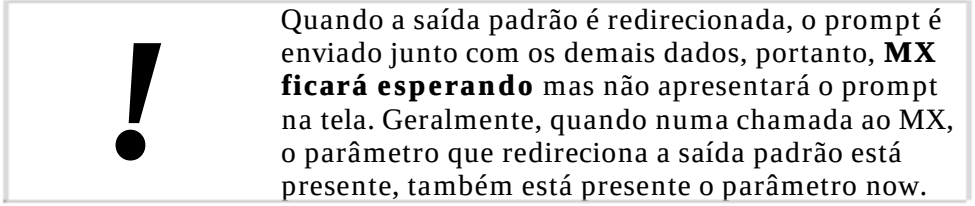
Exemplos:

- Enviar a listagem do resultado de uma busca para a impressora:

```
mx cds plants pft=@cds.pft now > LPT1
```

- Enviar a listagem de um intervalo de registros para um arquivo:

```
mx cds from=10 to=30 pft=@otro.pft lw=35 now > arquivo.txt
```



Considere que se for colocado o parâmetro ***-all*** não haverá saída, então o arquivo ao qual se direcionou a saída ficará vazio.

Relativos à seleção de registros:

<from> <to> <count> <loop>

Iniciar no registro n

from=<n>

O processo começa no registro <n> da base de dados de entrada; se não for especificado, o processo começa no primeiro registro do arquivo mestre.

Finalizar no registro n

to=<n>

O processo finaliza no registro <n> da base de entrada; se não for especificado, finaliza no último registro do arquivo mestre.

Processar um registro a cada n

loop=<n>

Pula <n> registros para cada registro processado.

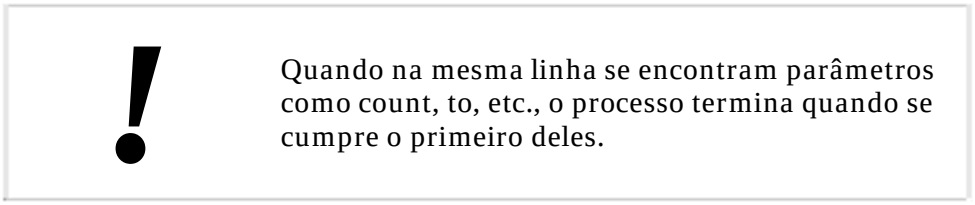
Por exemplo, se este parâmetro está presente e o valor precedido pelo sinal de igual é 5, serão processados os registros: 1, 6, 11, etc.

Selecionar *n* registros

count=<n>

O parâmetro *count=n* seleciona exatamente *n* registros a partir de um inicio dado. Se não for indicado registro de inicio, começa a partir do primeiro registro.

Exemplo com MX	**Saída**
`mx cds from=24 to=50 loop=5 pft=mfn/ now`	000024 000029 000034 000039 000044 000049
`mx cds from=24 to=50 count=3 loop=5 pft=mfn/ now`	000024 000029 000034
`mx cds from=24 to=50 count=9 loop=5 pft=mfn/ now`	000024 000029 000034 000039 000044 000049



Quando na mesma linha se encontram parâmetros como count, to, etc., o processo termina quando se cumpre o primeiro deles.

Relativos aos registros de Saída:

<offset>

Somar n aos números de registro

offset=<n>

Soma <n> ao MFN; assim o MFN que é guardado em *dbout* é o *MFN* do registro de entrada mais *offset*.

```
mxcp cds newcds offset=1000
```

No exemplo, ao indicar offset=1000, serão ingressados os registros da base cds na base newcds com mfn 1000, 1001, 1002, etc.

Substituição global de padrões

gizmo

O parâmetro *gizmo*= permite realizar substituições globais do conteúdo dos campos de uma base CDS/ISIS, converter uma cadeia de caracteres em outra, e assim realizar modificações, codificação/decodificação, compressão de dados, etc.

Estas substituições podem ser realizadas em todos os registros da base ou em um conjunto de registros (selecionados através de uma busca, um intervalo, etc.). As substituições, por sua vez, podem abranger o registro como um todo ou só alguns campos. Por exemplo, para substituir os sinais < > por / /, ou a cadeia de caracteres *plants* por *PLANTAS*, etc.

Para efetuar estas substituições é necessário dispor de um arquivo mestre *gizmo*. Este arquivo mestre tem em princípio dois campos: o campo 1 contém o dado a substituir, e o campo 2 o novo valor. Cada par de dados será um registro da base *gizmo*.

Cada registro de entrada é submetido ao procedimento de substituição estabelecido no arquivo *gizmo*. Ao começar a execução do MX os dados do arquivo gizmo são carregados como uma tabela na memória e são ordenados alfabeticamente pelo valor do campo 1 e pelo seu tamanho (desta maneira as cadeias de caracteres mais longas são convertidas antes que as curtas).

Exemplos:

Cria-se uma base de dados chamada TESTE usando os parâmetros de MX conhecidos e ingressam-se os dados diretamente a partir do teclado:

```
mx seq=con create=teste -all now
<|/
>|/
plants|PLANTAS
```

```
<ctrl>Z (ou <Fu6>)
```

Obtendo os registros seguintes:

```
mfn= 1
1 «<»
2 «/»
mfn= 2
1 «>»
2 «/»
mfn= 3
1 «plants»
2 «PLANTAS»
```

O conteúdo dos campos título e descritores do registro MFN=1 é:

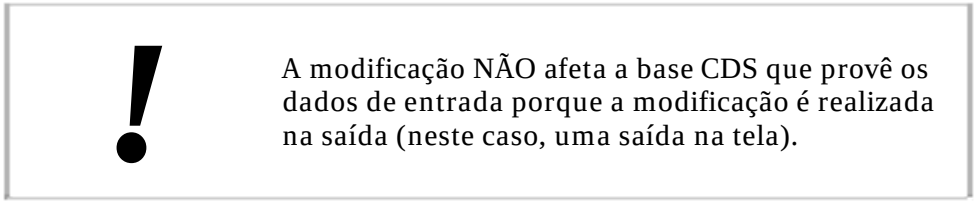
```
mx cds to=1 "pft=mfn/v24/v69"
000001
Techniques for the measurement of transpiration of individual plants
Paper on: <plant physiology><plant transpiration><measurement and instruments>
```

Se ao seguinte exemplo for aplicado o parâmetro gizmo:

```
mx cds gizmo=teste to=1 "pft=mfn/v24/v69"
```

Dará como resultado:

```
000001
Techniques for the measurement of transpiration of individual PLANTAS
Paper on: /plant physiology//plant transpiration//measurement and instruments/
```



Para modificar realmente os registros, o resultado do processo deveria ser enviado para o mesmo arquivo mestre, como no exemplo seguintes:

```
mx cds gizmo=teste to=1 copy=cds –all now
```

Se, por outro lado, se deseja gerar uma nova base de dados é necessário criá-la:

```
mx cds gizmo=teste to=1 create=Saída –all now
```

É possível restringir a modificação a um campo específico do registro, indicando após o parâmetro *gizmo* a etiqueta ou etiquetas nas quais se realizará a substituição. É possível indicar um intervalo de etiquetas separadas por /.

```
mx cds gizmo=teste,69,24 to=1 create=Saída -all now
mx cds gizmo=teste,35/56 to=1 create=Saída -all now
```

Apêndice II - Arquivo CIPAR

A interface de programação CISIS provê uma ferramenta chamada CIPAR, que faz às vezes do SYSPAR.PAR e dos <dbn>.par do CDS/ISIS padrão, além de muitas outras funções próprias do cisis. O CIPAR é um arquivo ASCII puro que ativa mecanismos para a busca de nomes de arquivos lógicos, nomes de bases de dados, estabelece parâmetros do ambiente de trabalho, etc.

Um arquivo de Parâmetros CIPAR consiste de linhas de comandos, um comando por linha, com instruções de equivalências lógicas. Uma equivalência lógica é uma sentença de atribuição, onde o valor da esquerda do sinal = representa o conjunto de valores da direita.

CISIS lê as linhas do arquivo CIPAR e as armazena na memória como uma tabela de referências. Cada linha é lida até que se chegue ao final do arquivo ou até que se encontre a combinação de /* (barra, asterisco), dependendo do que ocorre primeiro. O texto que segue após o /* é considerado como comentário e documentação sobre o CIPAR.

Exemplos:

Supondo que o arquivo se chama DADOS.CIP e contenha as seguintes linhas:

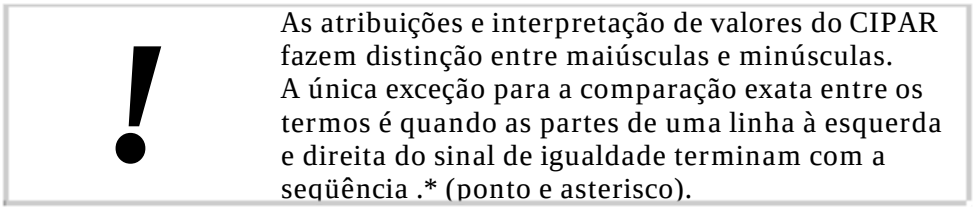
```
CDS.*=\cisis\bases\cds.*
CDS1.PFT=\cisis\bases\cds1.pft
```

então a instrução

```
mx cipar=dados.cip CDS pft=@CDS1.PFT from=10 ...
```

è equivalente a:

```
mx \cisis\bases\cds pft=\cisis\bases\cds1.pft from=10 ...
```



No seguinte exemplo:

```
dbxtrace=y
14=1
cds.*=c:\cisis\bases\cds.*
cds1.pft=c:\cisis\bases\cds1.pft
lilacs.xrf=c:\lilacsok.xrf
lilacs.*=d:\bases\lilacs\lilacs.*
```

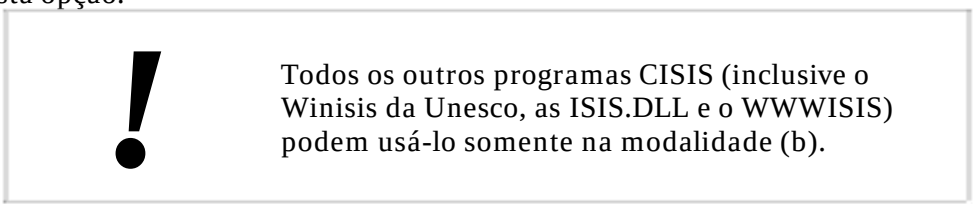
Os termos da esquerda serão ou não convertidos para os termos da direita dependendo se atendem ou não à comparação exata. Se o programa MX encontra uma seqüência de caracteres (que poderia ser o nome de uma base de dados, de um formato, etc.), buscará nas linhas do CIPAR e, se encontrar a linha de equivalência, realizará as transformações correspondentes.

Encontra	Converte para:
dbxTRACE	Não converte
dbxtrace	y
14	1
cds.mst	c:\cisis\bases\cds.mst
lilacs.xxx	d:\bases\lilacs\lilacs.xxx
lilacs.xrf	c:\lilacsok.xrf
LILACS.xxx	Não converte

Além da definição de nomes lógicos de arquivos é possível atribuir valores de variáveis de ambiente (*environment*) de forma global às aplicações.

Há duas maneiras de ativar o CIPAR:

a) Usando-o diretamente dentro da linha de comandos do MX, através do parâmetro *cipar=<file>*, onde *<file>* especifica o nome do arquivo que será usado como CIPAR durante a execução do programa. Só o programa MX pode usar esta opção.



b) Como variável de ambiente do sistema operacional. Para isto será usado o comando *set=CIPAR=<Nome>* onde *<Nome>* será o arquivo CIPAR que será usado em todas as execuções posteriores dos programas CISIS.

Exemplo:

```
c:\>set cipar=\cipar.par
```

Esta instrução criará uma variável de ambiente (uma variável global do sistema operacional) chamada CIPAR, cujo conteúdo é c:\cipar.par. Todas as aplicações CISIS que são executadas a partir deste momento tomarão este arquivo como CIPAR, não importa a partir de onde os programas são rodados.

É possível substituir o parâmetro por outro arquivo reatribuindo a variável de ambiente novamente com uma instrução set=<novo arquivo>.

```
c:\> set cipar=\outro.par
```

Para eliminar essa variável de ambiente do sistema operacional, basta atribuir-lhe um valor nulo:

```
c:\> set cipar=
```

A capacidade de atribuir valores globais do sistema operacional não é exclusivo do MS-DOS; outros sistemas operacionais, tais como Unix, também dispõem desta possibilidade, embora usem outros comandos:

- Sun Os e UNIX/CSH usam
  `setenv cipar=x`
- HP-UX, UNISYS 6000 e UNIX/ksh usam
  `cipar=x; export cipar`

Se existe CIPAR como variável de ambiente e, além disto, se declara o parâmetro *cipar=* na linha de comandos do MX, este último prevalecerá para a execução específica.

Exemplo:

```
C:\> set cipar=c:\cipar.par
C:\> mx cipar=outro.par CDS from=10 to= ... etc.
```

A execução deste comando do MX usará as definições do arquivo **outro.par**.

Parâmetros que podem ser incluídos no CIPAR

Parâmetros só para MX

[cgitag=<tag>] [cgipfx=<pfx>] cgi={<fmt>|mx}

Recebe parâmetros de uma chamada para um CGI

Armazena os dados do CGI em um campo repetitivo de etiqueta definida por *cgitag*, por default o campo 2000. Cada par de dados nome/valor é guardado nos subcampos ^n ^v. MX lerá cada linha mediante um formato (<fmt> ou mx.pft) que deverá ser incluído como parâmetro adicional.

cgitag=<tag>	Etiqueta de campo para os subcampo n, v (por default 2000)
cgipfx=<pfx>	Prefixo do nome da etiqueta de campo (por default **tag**)
cgi=<fmt>	Especifica os parâmetros do mx através de um formato
cgi=mx	Especifica os parâmetros do mx pelos nomes correspondentes

Exemplos:

- Na linha de comando são definidas as variáveis REQUEST_METHOD=GET e REQUEST_METHOD=GET, então se chama o mx indicando o formato a ser interpretado no modo CGI.
  - Para Linux e Windows:

    ```
    set REQUEST_METHOD=GET
    ```

  - Para Windows

    ```
    set "QUERY_STRING=db~cds&count~2&now&btell~0&bool~plants*water&pft~mfn/"
    ```

  - Para Linux

    ```
    set QUERY_STRING="db~cds&count~2&now&btell~0&bool~plants*water&pft~mfn/"
    ```

  - Para Linux e Windows:

    ```
    mx cgi=mx
    000004
    000011
    ```

Para que o formato interprete a variável *QUERY_STRING*, deve-se utilizar como separador de ocorrências o caráter reservado *&* e como separador de subcampos o caráter ~. O formato especificado deve tratar os subcampos *n* (anterior a ~ no caso de existir) e *v* (posterior a ~ no caso de existir) da etiqueta de definição de campo (tag).

- Na linha de comando são definidas as variáveis REQUEST_METHOD=GET, QUERY_STRING, então é chamado mx, forçando a leitura de dados em um campo determinado (11000 no exemplo) e indicando o formato a ser interpretado pelo modo CGI.
  - Para Linux e Windows:

    ```
    set REQUEST_METHOD=GET
    ```

  - Para Windows

    ```
    set "QUERY_STRING=db~cds&count~2&now&btell~0&bool~plants*water&pft~mfn/"
    ```

  - Para Linux

    ```
    set QUERY_STRING="db~cds&count~2&now&btell~0&bool~plants*water&pft~mfn/"
    ```

  - Para Linux e Windows:

    ```
    mx cgitag=11000   “cgi=(if v11000^n=’db’ then v11000^n,’=’,v11000^v/ fi)”
    mfn=1
     24 «Techniques for the measurement of transpiration of individual plants»
     26  «^aParis^bUmesco^c-1965»
     30  «^ap. 211-224^billus.»
     70  «Magalhaes, A.C»
     70  «Franco, C.M.»
     44  «Methodology of plant eco-physiology: proceedings of the Montpellier
    Symposium»
     ...
    ```

- Neste exemplo os parâmetros são passados pelo ambiente CGI e não pelas variáveis de ambiente. Em um servidor web, com o programa mx e o formato mx.pft gravados na área de CGI, monta-se a seguinte página.

```
<html>
   <head>
      <title>Exemplo de chamada de MX em ambiente CGI</title>
   </head>
<body>
   <form method="post"  action="/cgi-bin/mx.exe/cgi=mx" >
      Executar MX com  'bool=' e 'count='  atribuindo os seguintes valores:
<hr>
      Expressão de busca (bool=): <input type="text" name="bool"
value="$"></input> <br/ >
      Num. de registros a exibir (count=): <input type="text" name="count"
value="10">
```

```
            </input> <br/ >
          <input type="hidden" name="db" value="d:\httpd\bases\cds\cds”></input>
          <input type="hidden" name="now"></input>
          <input type="hidden" name="btell" value="0"></input>
          <input type="hidden" name="pft"

    value="lw(8000),newline('<br>'),@d:\httpd\bases\cds\cds”.pft"></input>
          <input type="submit" value="search & display"></input>
       </form>
    </body>
    </html>
```

No exemplo foi usado o formato padrão *mx.pft*, como pode ser observado na sentença *action* da etiqueta <form>

cipar: ci_tempdir=<path>

Define o drive e/ou diretório onde serão criados os arquivos temporários de trabalho. Se não for definido o parâmetro ci_tempdir, os arquivos temporários serão criados no diretório indicado pela variável de ambiente TEMP ou TMP.

```
echo ci_tempdir=c:\work >xcip
mx cipar=xcip cdromdb

@set ci_tempdir=c:\work
mx cdromdb
```

cipar: maxmfrl=<nbytes>

Especifica o tamanho do MFRmfrl (master file record length). Por default terá um tamanho de 32767 bytes que é máximo MFRmfrl padrão.

```
echo maxmfrl=32767 >xcip
mx cipar=xcip bigrecsdb
```

cipar: ci_fststrip=<maxlen>

Elimina qualquer marca do tipo <text> ou </text> até um máximo de <maxlen> caracteres de tamanho para os campos de dados no início de uma execução de FST (os campos de dados são preenchidos com espaços em branco).

```
echo ci_fststrip=21 >xcip
```

```
mx cds
mfn= 1
69 «Paper on: <plant physiology><plant transpiration><measurement and
instruments>»
mx cipar=xcip cds "fst=690 4 v69"
mfn= 1
69 «Paper on: <measurement and instruments> »
690 «PAPER^m1^o1^c1^l1»
690 «ON^m1^o1^c2^l1»
690 «MEASUREMENT^m1^o1^c3^l2»
690 «AND^m1^o1^c4^l1»
690 «INSTRUMENTS^m1^o1^c5^l2»
```

Parâmetros para MX e aplicações programadas para ambiente multiusuário

{netws | 14} | <dbn>.net ={0 | single | MONONETS} |

{1 | full | FULLNETS} |{ 2 | master | MASTNETS }

Os valores do parâmetro correspondem aos valores 0, 1, 2, do parâmetro 14 do SYSPAR.PAR. O valor por default é MONONETS. Também é possível atribuir o parâmetro de rede a uma base de dados específica através de <*dbn*>.*net*=<*n*>

Exemplos:

```
14=1
netws=MASTNETS
cds.net=FULLNET
```

Parâmetro maxmfrl

maxmfrl=<n>

Realiza a mesma especificação que o parâmetro *mfrl*=<*n*> da linha de comandos do programa MX.

Parâmetro mstxl no CIPAR

mstxl=<n>

mstxl | <dbn>/mstxl=<n>

n=0 ... 4 (0 é o valor por default, predefinido na Interface CISIS)

Determina o tamanho máximo que o arquivo .MST pode ter. Este parâmetro é lido e considerado somente durante a criação da base de dados.

<n>	Máximo .MST	Registro em bytes múltiplo de
0	512 MB	2 (valor por default)
1	1 GB	2
2	2 GB	4
3 ou 4	4 GB	8

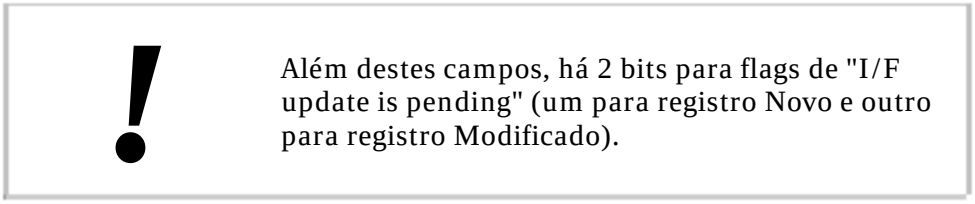
Exemplo:

```
cds/mstxl=3
```

Estes parâmetros também se aplicam ao WinISIS, ISIS.DLL e WWWISIS e a todas as aplicações desenvolvidas com a Interface CISIS.

Como superar o limite de 512 MB para o arquivo mestre:

O ponteiro de cada registro do .MST está armazenado no .XRF. Este ponteiro se divide em dois campos: o campo número do bloco e o campo deslocamento (*offset*) dentro do bloco. Estes dois campos ocupam respectivamente 20 e 9 bits.



O ponteiro tem 4 bytes e é tratado como inteiro com sinal (um valor negativo indica fisicamente/logicamente apagado).

Desta forma o limite é de 512 MB, como indicado mais abaixo:

```
2**20-1 = 1048675 blocos de 512 (offset a partir de 0 até 2**9-1 = 511)
```

A implementação do MSTXL na Interface CISIS usa *offset* para blocos de 512 bytes em unidades de 2, 4 e 8 bytes (usando por tanto 8, 7 e 6 bits para armazenar o offset) e número de bloco em 21, 22, e 24 bits, respectivamente, para mstxl = 1, 2, e 3 (ou 4).

A indicação de um arquivo mestre em formato MSTXL está armazenada no byte mais à esquerda do campo MFTYPE do registro de controle do .MST, que é reinicializado quando é carregado na memória para processamento. Esta indicação é "recordada" por CISIS durante as leituras/gravações posteriores até que esse arquivo mestre seja fechado

É suficiente criar um arquivo mestre com um CIPAR que contenha *mstxl*=<*n*> ou <*dbn*>/*mstxl*=<*n*> para que os processos subseqüentes respeitem essa inicialização independentemente do CIPAR.

Parâmetro dbxtrace=y

dbxtrace=y

EXIBE mensagens na saída padrão (*stdout - standard output*):

```
dbxopen - <dbn><.ext> fd=<n> [R]
dbxopen - <dbn><.ext> fd=<n> [RW]
```

<n> número de descritor de arquivo
[R] arquivo foi aberto para leitura (*read*)
[RW] arquivo foi aberto para leitura/gravação (*read/write*)

dbxtrace=y idem "trace=dbx" do MX (como trace=rec, trm, giz, b40, par, mul, etc)

Parâmetro mstload=<n>

mstload=<n>

Carrega na memória e fecha os arquivos mestres (default=0).

```
mstload=<n>    idem "load=<n>" do MX, para M/F
```

Parâmetro invload=<n>

invload=<n>

Carrega na memória e fecha os arquivos invertidos (default=0).

```
invload=<n>    idem "load=<n>" del MX, para I/F
```

Parâmetro mcose={y|n}

mcose={y|n}

Fecha todos os arquivos mestres (default=n).

```
mcose=y só permanece aberto um arquivo mestre
```

Parâmetro iflush={y|n}

iflush={y|n}

Esvazia todos os arquivos invertidos (default=n).

```
iflush=y      só permanece aberto um arquivo invertido com seu "data base
              descriptor" na memória
```

Parâmetro mflush={y|n}

mflush={y|n}

Esvazia todos os arquivos mestres (default=n).

```
mflush=y        só permanece aberto um arquivo mestre com seu "data base
                descriptor" na memória. (data base descriptor inclui buffers de
                e/s e gizmos)
```

Parâmetro what={y|n}

what={y|n}

Exibe a versão CISIS (default=n).

```
what=y     idem "what" do MX (chamado como único parâmetro do MX)
```

Exemplos

- Criar um arquivo mestre sem registros ou reinicializar um existente

```
mx seq=NUL create=NOVO
```

Se existir uma base de dados com esse nome, MX não avisará que esta será eliminada irreversivelmente.

- Compactar uma base de dados

Este procedimento recuperará o espaço perdido no MST pelas sucessivas atualizações de registros.

```
mx DADOS -all now create=DBN_AUX tell=100
xcopy DBN_AUX.* DADOS.*
del DBN_AUX.*
FULLINV DADOS DADOS.FST DADOS
```

- Verificar elementos duplicados

Desejando-se verificar no catálogo de livros que não existem números de inventário duplicados. Os inventários são registrados no campo 7 e são indexados com técnica 0 com o prefixo INV=, isto é: |INV=|v7.

```
mx LIVROS "pft=(if npost(|INV=|v7) > 1 then mfn,x3,v7/ fi)" tell=100 now >
duplic.lst
```

O procedimento requer que esteja presente o arquivo invertido.

- Fazer uma análise rápida dos campos usados em uma base de dados

São necessários os programas MX.EXE e MXF0.EXE e estes dois arquivos de formatos:

```
DMXF0A.PFT
'Análise dos dados da base ',v1001/#
" 1001 = input master file name ................. "v1001/,
" 1003 = date & time stamp ...................... "v1003/,
" 1009 = total number of records ................ "v1009/,
" 1010 = number of active records ............... "v1010/,
```

```
" 1011 = number of logically deleted records .... "v1011/,
" 1012 = number of phisically deleted records ... "v1012/,
DMXF0B.PFT
/#
"TAG DOCS OCCS"d1020/
"---------------------"d1020/
,(v1020^t,v1020^d,v1020^o/),
Com este se prepara um arquivo .bat, como o siguinte:
MYSCAN.BAT
REM %1 = <dbn_name> %2 = <nro regs estimados>
REM
mxf0 %1 create=lista %2 tell=100
mx lista pft=@dmxf0a.pft now > %1.lst
mx lista pft=@dmxf0b.pft now >> %1.lst
```

A execução de MYSCAN.BAT requer dois parâmetros de entrada, a base de dados com seu path e a quantidade estimada de registros que contem.

Como resultado será gerado um arquivo com o nome da base e extensão .lst.

Exemplo:

```
myscan c:\dbisis\cds\cds 150
```

- Eliminar termos duplicados em um campo repetitivo

Supondo que os descritores são registrados no campo v87, como campo repetitivo.

```
mx DADOS fmtl=20000 proc=@LIMPO from=1 to=100 now -all create=OUT
```

O arquivo LIMPO tem a seguinte especificação de formato:

```
proc('d870d871'),
( if v870[1] : s(v87|~|)
then
else proc('D870A870|'v870[1],v87'~|','A871|'v87'|')
fi ),
proc('d870'),
proc('d87d871',|A87~|v871|~|),
```

- Controle de qualidade dos dados

É oferecido um modelo simplificado de um arquivo .bat para realizar vários controles sobre uma base de dados. Para este exemplo, dispõe-se de uma base de dados chamada

TEST para a qual são realizados os controles indicados no menu, e uma base THES (tesauro) contra a qual são validados os descritores de TEST.

```
CHK.BAT
@echo off
:BEGIN
cls
echo ------------------------------------------------------------
echo QUALITY CONTROL
echo ------------------------------------------------------------
echo.
echo E - Delete invalid characters (clean records)
echo O - Check mandatory fields
echo D - Check Descriptors
echo X - Exit
echo.
choice /c:EODX /N Select one option:
if errorlevel 4 goto END
if errorlevel 3 goto DESCRIPTORES
if errorlevel 2 goto OBLIGATORIOS
if errorlevel 1 goto CLEAN
:CONTINUA
echo.
echo (Press any key to continue...)
pause > nul
goto BEGIN
:CLEAN
call OPC_CLN. BAT
goto CONTINUA
:OBLIGATORIOS
call OPC_OBL.BAT
goto CONTINUA
:DESCRIPTORES
call OPC_DES.BAT
goto CONTINUA
:END
```

O procedimento é completado com uma série de arquivos .bat que realizam cada uma das opções apresentadas.

Os arquivos são denominados OPC_xxx, onde xxx representa cada uma das opções do menu. Cada OPC_xxx ativa um programa CISIS que chama um arquivo in=<text_file> que contem os parâmetros necessários para a função de validação. Associado a este <text_file> há um arquivo de formato com o mesmo nome e com extensão .pft, que será usado para a validação dos dados.

Se a validação é realizada contra um arquivo invertido, o in=<text_file> deverá chamar um procedimento de jchk.

```
OPC_CLN.BAT
@echo off
mxcp \dbisis\TEST \dbisis\TEST clean > garbage
OPC_OBL.BAT
@echo off
echo Wait..
mx in=chk00 >> errores.tmp
CHK00
\dbisis\test
pft=@chk00.pft
-all
now
CHK00.PFT (exemplo de um .PFT de validación)
if a(v02) then mfn,c8,'02->FATAL ERROR: field #2 missing!' fi/
if a(v01) then mfn,c8,'01->ERROR: field #01 missing!' fi/
if a(v923) and p(v23) then mfn,c8,'923->ERROR: field #923 missing!' fi/
if v31<>'ENG' then mfn,c8,'31->ERROR: field #31 invalid!' fi/
OPC_DES.BAT
@echo off
echo Wait...
mx in=chk620 >> errores.tmp
 CHK620 (verifica el campo 620 contra um tesauro)
\dbisis\test
jchk=Thes=mhu,(v620/)
pft=@chk620.pft
-all
now
 CHK620.PFT
if p(v32001) then ( if a(v32001^m) then mfn,c8,|620->|v32001^k,c65,'*invalid'/
fi ) fi
```

- Índice especial de títulos

Ao indexar o campo de título (séries ou monografias) com técnica 0 (campo completo), a ordenação alfabética destes no dicionário não respeita as normas biblioteconômicas de não considerar as palavras tais como: A, An, El, Os, The, etc.

No CDS/ISIS é possível obter a saída impressa de forma correta se essas palavras estiverem entre <..>, mas esta solução não está disponível para o arquivo invertido. O seguinte exemplo resolve o problema.

Será indexado o campo v24 (título), entre os registros 15 e 20, de três maneiras para comparar os resultados. Os exemplos 1 e 2 mostram o que é obtido através do CDS/ISIS padrão, e o exemplo 3 mostra a solução do problema.

Usando-se a seguinte FST: 240 0 v24

```
mx CDS from=15 to=20 now -all "fst=240 0 v24" ifupd/create=cds
ifkeys cds +tags
1| 240|<A> METHOD OF DETERMINING EVAP
1| 240|<THE> HEAT RESISTANCE OF PLANT
1| 240|<THE> MEASUREMENT OF DROUGHT R
1| 240|<THE> ROLE OF DEW IN PINE SURV
1| 240|GAUGES FOR THE STUDY OF EVAPOT
1| 240|MEASUREMENT OF DROUGHT RESISTA
```

Todos os títulos que tem <...> aparecem no início do dicionário, pois o caráter < tem um valor ASCII menor do que a letra "A".

Usando-se a seguinte FST: 240 0 mhu,v24

```
mx CDS "fst=240 0 mhu,v24" ifupd/create=cds now -all from=15 to=20
```

O seguinte resultado é apresentado no dicionário:

```
Ifkeys CDS +tags
1| 240|A METHOD OF DETERMINING EVAPOT
1| 240|GAUGES FOR THE STUDY OF EVAPOT
1| 240|MEASUREMENT OF DROUGHT RESISTA
1| 240|THE HEAT RESISTANCE OF PLANTS,
1| 240|THE MEASUREMENT OF DROUGHT RES
1| 240|THE ROLE OF DEW IN PINE SURVIV
```

Devido ao uso de MHU, os <..> não aparecem no dicionário. A alfabetação considera as palavras não significativas.

A solução é obtida usando um *gizmo*= WORDS, cujo conteúdo é:

```
mfn= 1
1 «<The> »
mfn= 2
1 «<El> »
mfn= 3
1 «<An> »
mfn= 4
1 «<A> »
mfn= 5
1 «<L' >»
mfn= 6
1 «<LA> »
mfn= 7
1 «<La> »
mfn= 8
1 «<L'>»
..
```

Não há campo 2 no gizmo, porque a conversão transforma o dado do campo 1 em uma saída nula. Além disto, deixa-se um espaço em branco ao final de cada campo para que os campos de títulos, depois de aplicado o gizmo, não comecem com um espaço em branco.

```
mx CDS from=15 to=20 now -all "fst=240 0 v24" ifupd/create=cds gizmo=words,24
ifkeys CDS +tags
1| 240|GAUGES FOR THE STUDY OF EVAPOT
1| 240|HEAT RESISTANCE OF PLANTS, ITS
2| 240|MEASUREMENT OF DROUGHT RESISTA
1| 240|METHOD OF DETERMINING EVAPOTRA
1| 240|ROLE OF DEW IN PINE SURVIVAL I
```

Apêndice III - Estrutura dos registros de uma base ISIS

Um registro com estrutura ISIS tem duas características especiais que oferecem uma grande versatilidade para o tratamento de informação textual: campos repetitivos e de tamanho variável.

Considerando que os registros não têm um tamanho predeterminado, nem os campos têm um tamanho fixo, nem uma quantidade predeterminada de ocorrências, não é possível ter acesso direto a nenhuma porção de dados dentro da base.

O acesso ao registro é feito de modo indireto, através de ponteiros em um arquivo auxiliar com extensão .XRF, e dentro do registro os dados são acessados através de ponteiros em um diretório.

O arquivo .XRF contem toda a informação necessária para encontrar o ponto de início do registro solicitado dentro do .MST.

Para todos os exemplos que seguem será usado o registro MFN=1 da base CDS, cujo conteúdo completo é o seguinte:

\cisis\bases\cds	Base de dados
Nxtmfn nxtmfb nxtmfp t reccnt mfcxx1 mfcxx2 mfcxx3 RC 152 123 13 0 0 0 0 0 0	Registro de controle (control)
Mfn= 1|comb= 1|comp= 64| N| ... | 1+000=00000c40	xref

Mfn= 1|mfrl= 370|mfbwb= 0|mfbwp= 0|base= 66|nvf= 8|status= 0| 0	leader
Mfn= 1|dir= 1|tag= 44|pos= 0|len= 77 Mfn= 1|dir= 2|tag= 50|pos= 77|len= 11 Mfn= 1|dir= 3|tag= 69|pos= 88|len= 78 Mfn= 1|dir= 4|tag= 24|pos= 166|len= 68 Mfn= 1|dir= 5|tag= 26|pos= 234|len= 22 Mfn= 1|dir= 6|tag= 30|pos= 256|len= 20 Mfn= 1|dir= 7|tag= 70|pos= 276|len= 15 Mfn= 1|dir= 8|tag= 70|pos= 291|len= 12	dir
Mfn= 1 44 «Methodology of plant eco-physiology: proceedings of the Montpellier Symposium» 50 «Incl. bibl.» 69 «Paper on: <plant physiology><plant transpiration><measurement and instruments>» 24 «Techniques for the measurement of transpiration of individual plants» 26 «^aParis^bUmesco^c-1965» 30 «^ap. 211-224^billus.» 70 «Magalhaes, A.C.» 70 «Franco, C.M.» ..	fields

O Registro de CONTROLE

Uma base ISIS tem um registro especial no início (MFN=0) ao qual CDS/ISIS não provê acesso. Este registro tem uma estrutura diferente dos demais registros do arquivo mestre (MST).

A informação deste registro é mostrada com o parâmetro +control.

A estrutura é:

Estrutura do registro de CONTROLE	
Ctlmfn	Sempre 0. Este campo não aparece com o MX <dbn> +control.
Nxtmfn	MFN a ser atribuído no próximo registro.

Estrutura do registro de CONTROLE	
Nxtmfb	Último bloco atribuído no .MST. Os blocos são de 512 bytes.
Nxtmfp	Primeira posição livre dentro do último bloco atribuído. Um registro pode começar em qualquer posição livre entre 0-498 e estender-se por um ou mais blocos. Nenhum registro pode começar entre o byte 500 e 510.
Mftype	Tipo de base de dados: 0 base do usuário, 1 base de mensagens do sistema.
Reccnt	Reservado.
Mfcxx1	Reservado.
Mfcxx2	Bloqueio de entrada de dados, valores 0, 1...n, depende de quantos registros estão sendo editados em certo momento.
Mfcxx3	Bloqueio de leitura exclusiva, valores 0/1.

O Registro de XREF

O arquivo .XRF está organizado como uma tabela de ponteiros para o arquivo mestre (.MST). O primeiro ponteiro corresponde ao MFN=1, o segundo ao MFN=2, etc.

Cada ponteiro consiste de dois campos (4 bytes) que na tabela do exemplo anterior indicam que o MFN=1 começa no bloco 1 (*comb*) e dentro do bloco a partir do byte 65 (*comp*) e que não está pendente a atualização do arquivo invertido. O último valor é a referência real do ponteiro expressada em valor hexadecimal.

Cada bloco do .XRF é um arquivo de 512 bytes de tamanho e contem 127 ponteiros (dado importante para a reconstrução do .MST que é explicado no utilitário CTLMFN).

O Registro do Arquivo MST

Os registros do arquivo mestre são armazenados consecutivamente, um após outro, ocupando cada registro exatamente *MFRL* bytes. Cada arquivo é armazenado como blocos físicos de 512 bytes. Um registro pode começar em qualquer ponto entre a posição 0 e 498 e pode estender-se por um ou mais blocos.

O registro MST tem tamanho variável e consiste de três seções:

- Uma de tamanho fixo (*leader*);
- Um diretório;
- Os campos de dados de tamanho variável.

Estrutura do LEADER

O *leader* consiste de um bloco de tamanho fixo de 18 bytes.

Mfn	Número de registro
Mfrl	Tamanho total do registro, incluindo as três seções: *leader*, *diretório* e *área de dados*. Sempre é um número par.
Mfbwb	Ponteiro para a versão anterior do registro, número do bloco dentro do MST. Inicialmente está com 0, e também após a atualização do arquivo invertido.
Mfbwp	Ponteiro para a versão anterior do registro. Deslocamento dentro do bloco.
Base	Posição onde começa a área de dados dentro do registro. Este valor é a soma do tamanho do *leader* mais o tamanho do diretório.
Nvf	Quantidade de campos no registro, ou seja, a quantidade de entradas no diretório do registro.
Status	Indicador de apagado (0 = registro ativo; 1 = logicamente apagado)

Estrutura do DIRETÓRIO

O *diretório* do registro é um índice para os conteúdos do registro no segmento de dados. Este índice é constituído por tantas entradas conforme o número de campos (nvf), permitindo o acesso aos dados. Cada entrada no *diretório* é formada por três partes:

Tag	Identificador de campo ou etiqueta.
Pos	Endereço em bytes onde começa o primeiro caráter na área de dados correspondente a esta ocorrência do campo.
Len	Tamanho do campo em bytes.

Apêndice IV - Lista de arquivos TABs disponíveis

Para serem usados tanto no modo de apresentação como para inversão das bases

ASCII CODE PAGE 437 (CP437)

ac437.tab	Caracteres válidos do conjunto ASCII CP437
ac437n.tab	Caracteres válidos do conjunto ASCII CP437 incl. 0-9
ac437XT.tab	Caracteres válidos do conjunto ASCII CP437 incl. 0-9 e &'()*+,-./:
ma437.tab	Conversão dos caracteres para maiúsculas (com acento)
mi437.tab	Conversão dos caracteres para minúsculas (com acento)
na437.tab	Elimina acentos dos caracteres, mantendo maiúsculas/minúsculas
uc437.tab	Conversão dos caracteres para maiúsculas (sem acentos)
lc437.tab	Conversão dos caracteres para minúsculas (sem acentos)

ASCII CODE PAGE 850 (CP850)

ac850.tab	Caracteres válidos do conjunto ASCII CP850
ac850n.tab	Caracteres válidos do conjunto ASCII CP850 incl. 0-9
ac850XT.tab	Caracteres válidos do conjunto ASCII CP850 incl. 0-9 e &'()*+,-./:
ma850.tab	Conversão dos caracteres para maiúsculas (com acento)

mi850.tab	Conversão dos caracteres para minúsculas (com acento)
na850.tab	Elimina acentos dos caracteres, mantendo maiúsculas/minúsculas
uc850.tab	Conversão dos caracteres para maiúsculas (sem acentos)
lc850.tab	Conversão dos caracteres para minúsculas (sem acentos)

ANSI (Windows)

acans.tab	Caracteres válidos do conjunto ANSI
acansn.tab	Caracteres válidos do conjunto ANSI incl. 0-9
acansXT.tab	Caracteres válidos do conjunto ANSI incl. 0-9 e &'()*+,-./:
maans.tab	Conversão dos caracteres para maiúsculas (com acento)
mians.tab	Conversão dos caracteres para minúsculas (com acento)
naans.tab	Elimina acentos dos caracteres, mantendo maúsculas/minúsculas
ucans.tab	Conversão dos caracteres para maiúsculas (sem acentos)
lcans.tab	Conversão dos caracteres para minúsculas (sem acentos)

GIZMOs disponíveis para conversão do conteúdo de bases de dados

Conversão do conjunto de caracteres

g437ans	Conversão de ASCII CODE PAGE 437 para ANSI
gans437	Conversão de ANSI para ASCII CODE PAGE 437
g850ans	Conversão de ASCII CODE PAGE 850 para ANSI
gans850	Conversão de ANSI para ASCII CODE PAGE 850
gutf8ans	Conversão de UFT-8 para ANSI
gansutf8	Conversão de ANSI para UFT-8

ASCII CODE PAGE 437 (CP437)

g437ma	Conversão para maiúsculas acentuadas
g437mi	Conversão para minúsculas acentuadas
g437na	Retira acentos mantendo maiúsculas/minúsculas
g437uc	Conversão para maiúsculas sem acento (Upper case)
g437lc	Conversão para minúsculas sem acento (Lower case)

ASCII CODE PAGE 850 (CP850)

g850ma	Conversão para maiúsculas acentuadas
g850mi	Conversão para minúsculas acentuadas
g850na	Retira acentos mantendo maiúsculas/minúsculas
g850uc	Conversão para maiúsculas sem acento (Upper case)
g850lc	Conversão para minúsculas sem acento (Lower case)

ANSI (Windows)

gansma	Conversão para maiúsculas acentuadas
gansmi	Conversão para minúsculas acentuadas
gansna	Retira acentos mantendo maiúsculas/minúsculas
gansuc	Conversão para maiúsculas sem acento (Upper case)
ganslc	Conversão para minúsculas sem acento (Lower case)

Conversão auxiliar de caracteres de marcação

gentit	Conversão de caracteres pela entidade HTML correspondente ("&'<>)
gchar	Conversão para a entidade HTML pelos caracteres correspondentes ("&'<>)

Conversão auxiliar de e para entidades HTML

Ghtmlans	Converte entidades HTML em caracteres ANSI
ganshtml	Converte caracteres ANSI em entidades HTML
ghtml850	Converte entidades HTML em caracteres ASCII CP850
g850html	Converte caracteres ASCII CP850 em entidades HTML
ghtml437	Converte entidades HTML em caracteres ASCII CP437
g437html	Converte caracteres ASCII CP437 em entidades HTML

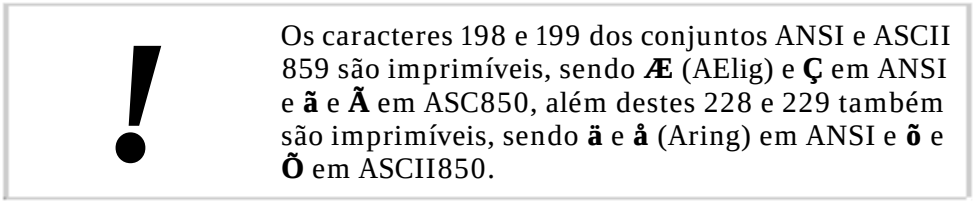
Como reconhecer o conjunto de caracteres em que está uma base CDS/ISIS

Se na distribuição de caracteres oferecida pelo `MXF0` houver AUSENCIA de caracteres com códigos entre 127(0x7f) e 159(0x9f) (inclusive), então pertencem ao conjunto ANSI.

Se na distribuição de caracteres oferecida pelo `MXF0` houver uma forte concentração entre os códigos 128(0x80) até 167(0xa7) (inclusive) é muito provável que seja Code Page 437.

Se na distribuição de caracteres oferecida pelo `MXF0` houver presença de elementos em algum dos seguintes códigos: 181(0xb5), 182, 183, 198, 199, 210, 211, 212, 214, 215, 216, 222, 224, 226, 227, 228, 229, 233, 234, 235, 236, 237 é muito provável que seja Code Page 859.

É importante a presença como caracteres discriminadores do código de página: o 198(0xc6) e 199(0xc6), que são **ã** e **Ã**, e também 228(0xe4) e 229(0xe5) que são **õ** e **Õ**.



Os caracteres 198 e 199 dos conjuntos ANSI e ASCII 859 são imprimíveis, sendo **Æ** (AElig) e **Ç** em ANSI e **ã** e **Ã** em ASC850, além destes 228 e 229 também são imprimíveis, sendo **ä** e **å** (Aring) em ANSI e **õ** e **Õ** em ASCII850.

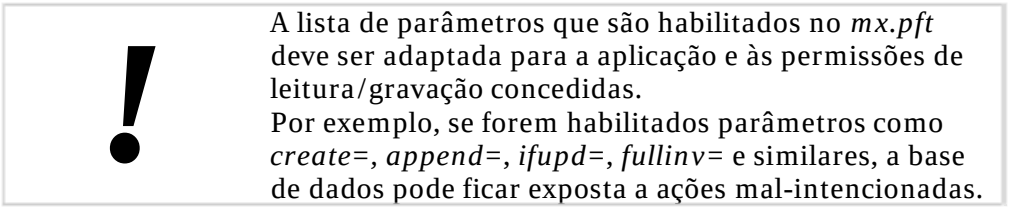
Elemento de complicação: Windows 95 e 98 utilizam codificação diferente de Windows 98SE e posteriores.

OBSERVAÇÕES

Os gizmos de conversão entre os conjuntos de caracteres: ANSI; ASCII CP437; e ASCII CP850, têm como objetivo obter caracteres gráficos similares.

Os gizmos de CONVERSÃO AUXILIAR abrangem só os caracteres acentuados.

Apêndice V - MX.PFT: Lista de parâmetros que extraem do ambiente CGI



A lista de parâmetros que são habilitados no *mx.pft* deve ser adaptada para a aplicação e às permissões de leitura/gravação concedidas.
Por exemplo, se forem habilitados parâmetros como *create=*, *append=*, *ifupd=*, *fullinv=* e similares, a base de dados pode ficar exposta a ações mal-intencionadas.

Parâmetro	Formato de extração
what	,(if v2000^n='what' then v2000^n/ fi),
prolog=	,(if v2000^n='prolog' then v2000^n'='v2000^v/ fi),
epilog=	,(if v2000^n='epilog' then v2000^n'='v2000^v/ fi),
in=	,(if v2000^n='in' then v2000^n'='v2000^v/ fi),
trace	,(if v2000^n='trace' then v2000^n/ fi),
trace=dbx	,(if v2000^n='trace=dbx' then v2000^n/ fi),
trace=rec	,(if v2000^n='trace=rec' then v2000^n/ fi),
trace=dec	,(if v2000^n='trace=dec' then v2000^n/ fi),
trace=trm	,(if v2000^n='trace=trm' then v2000^n/ fi),
trace=b50	,(if v2000^n='trace=b50' then v2000^n/ fi),
trace=b40	,(if v2000^n='trace=b40' then v2000^n/ fi),
trace=fmt	,(if v2000^n='trace=fmt' then v2000^n/ fi),
trace=fst	,(if v2000^n='trace=fst' then v2000^n/ fi),
trace=all	,(if v2000^n='trace=all' then v2000^n/ fi),
cipar=	,(if v2000^n='cipar' then v2000^n'='v2000^v/ fi),

Parâmetro	Formato de extração
mfrl=	,(if v2000^n='mfrl' then v2000^n'='v2000^v/ fi),
fmtl=	,(if v2000^n='fmtl' then v2000^n'='v2000^v/ fi),
load=	,(if v2000^n='load' then v2000^n'='v2000^v/ fi),
pages=	,(if v2000^n='pages' then v2000^n'='v2000^v/ fi),
uctab=	,(if v2000^n='uctab' then v2000^n'='v2000^v/ fi),
actab=	,(if v2000^n='actab' then v2000^n'='v2000^v/ fi),
-all	,(if v2000^n='-all' then v2000^n/ fi),
-control	,(if v2000^n='-control' then v2000^n/ fi),
-leader	,(if v2000^n='-leader' then v2000^n/ fi),
-xref	,(if v2000^n='-xref' then v2000^n/ fi),
-dir	,(if v2000^n='-dir' then v2000^n/ fi),
-fields	,(if v2000^n='-fields' then v2000^n/ fi),
+hits	,(if v2000^n='+hits' then v2000^n/ fi),
cgiplushits	,(if v2000^n='+hits''cgiplushits' then v2000^n'+hits'/ fi),
+fix	,(if v2000^n='+fix' then v2000^n/ fi),
+fix/m	,(if v2000^n='+fix/m' then v2000^n/ fi),
+all	,(if v2000^n='+all' then v2000^n/ fi),
+control	,(if v2000^n='+control' then v2000^n/ fi),
+leader	,(if v2000^n='+leader' then v2000^n/ fi),
+xref	,(if v2000^n='+xref' then v2000^n/ fi),
+dir	,(if v2000^n='+dir' then v2000^n/ fi),
+fields	,(if v2000^n='+fields' then v2000^n/ fi),
from=	,(if v2000^n='from' then v2000^n'='v2000^v/ fi),
to=	,(if v2000^n='to' then v2000^n'='v2000^v/ fi),
loop=	,(if v2000^n='loop' then v2000^n'='v2000^v/ fi),
count=	,(if v2000^n='count' then v2000^n'='v2000^v/ fi),
tell=	,(if v2000^n='tell' then v2000^n'='v2000^v/ fi),
b50=	,(if v2000^n='b50' then v2000^n'='v2000^v/ fi),
b40=	,(if v2000^n='b40' then v2000^n'='v2000^v/ fi),
nb1=	,(if v2000^n='nb1' then v2000^n'='v2000^v/ fi),
nbb=	,(if v2000^n='nbb' then v2000^n'='v2000^v/ fi),
nb0=	,(if v2000^n='nb0' then v2000^n'='v2000^v/ fi),
nb2=	,(if v2000^n='nb2' then v2000^n'='v2000^v/ fi),
btell=	,(if v2000^n='btell' then v2000^n'='v2000^v/ fi),
b50t=	,(if v2000^n='b50t' then v2000^n'='v2000^v/ fi),
b40t=	,(if v2000^n='b40t' then v2000^n'='v2000^v/ fi),
b40u=	,(if v2000^n='b40u' then v2000^n'='v2000^v/ fi),
dupr=	,(if v2000^n='dupr' then v2000^n'='v2000^v/ fi),
dupl=	,(if v2000^n='dupl' then v2000^n'='v2000^v/ fi),
p1=	,(if v2000^n='p1' then v2000^n'='v2000^v/ fi),
p2=	,(if v2000^n='p2' then v2000^n'='v2000^v/ fi),
nowait	,(if v2000^n='nowait' then v2000^n/ fi),
now	,(if v2000^n='now' then v2000^n/ fi),
stderr=off	,(if v2000^n='stderr=off' then v2000^n/ fi),
mono	,(if v2000^n='mono' then v2000^n/ fi),
mast	,(if v2000^n='mast' then v2000^n/ fi),
full	,(if v2000^n='full' then v2000^n/ fi),
db=	,(if v2000^n='db' then v2000^n'='v2000^v/ fi),
dbn=	,(if v2000^n='dbn' then v2000^n'='v2000^v/ fi),

Parâmetro	Formato de extração
dict=	,(if v2000^n='dict' then v2000^n'='v2000^v/ fi),
k1=	,(if v2000^n='k1' then v2000^n'='v2000^v/ fi),
k2=	,(if v2000^n='k2' then v2000^n'='v2000^v/ fi),
cgi=	,(if v2000^n='cgi' then v2000^n'='v2000^v/ fi),
null	,(if v2000^n='null' then v2000^n/ fi),
tmp	,(if v2000^n='tmp' then v2000^n/ fi),
seq=	,(if v2000^n='seq' then v2000^n'='v2000^v/ fi),
iso=	,(if v2000^n='iso' then v2000^n'='v2000^v/ fi),
isotag1=	,(if v2000^n='isotag1' then v2000^n'='v2000^v/ fi),
bool=	,(if v2000^n='bool' then if v2000^v>'' then v2000^n'='v2000^v/ fi fi), if v2000:'^nbool^v' then 'bool=' (if v2000^n='bool' then if v2000^v>'' then v2000^v fi fi),/ fi,
mfbw	,(if v2000^n='mfbw' then v2000^n/ fi),
tbin=	,(if v2000^n='tbin' then v2000^n'='v2000^v/ fi),
tb=	,(if v2000^n='tb' then v2000^n'='v2000^v/ fi),
join=	,(if v2000^n='join' then v2000^n'='v2000^v/ fi),
jchk=	,(if v2000^n='jchk' then v2000^n'='v2000^v/ fi),
jch0=	,(if v2000^n='jch0' then v2000^n'='v2000^v/ fi),
jch1=	,(if v2000^n='jch1' then v2000^n'='v2000^v/ fi),
jmax=	,(if v2000^n='jmax' then v2000^n'='v2000^v/ fi),
jtag=	,(if v2000^n='jtag' then v2000^n'='v2000^v/ fi),
proc=	,(if v2000^n='proc' then v2000^n'='v2000^v/ fi),
convert=	,(if v2000^n='convert' then v2000^n'='v2000^v/ fi),
decod=	,(if v2000^n='decod' then v2000^n'='v2000^v/ fi),
gizp=	,(if v2000^n='gizp' then v2000^n'='v2000^v/ fi),
gizmo=	,(if v2000^n='gizmo' then v2000^n'='v2000^v/ fi),
giz1=	,(if v2000^n='giz1' then v2000^n'='v2000^v/ fi),
giz2=	,(if v2000^n='giz2' then v2000^n'='v2000^v/ fi),
putdir=	,(if v2000^n='putdir' then v2000^n'='v2000^v/ fi),
getdir=	,(if v2000^n='getdir' then v2000^n'='v2000^v/ fi),
sys=	,(if v2000^n='sys' then v2000^n'='v2000^v/ fi),
sys/show=	,(if v2000^n='sys/show' then v2000^n'='v2000^v/ fi),
text=	,(if v2000^n='text' then v2000^n'='v2000^v/ fi),
text/show=	,(if v2000^n='text/show' then v2000^n'='v2000^v/ fi),
lw=	,(if v2000^n='lw' then v2000^n'='v2000^v/ fi),
pft=	,(if v2000^n='pft' then v2000^n'='v2000^v/ fi),
invx=	,(if v2000^n='invx' then v2000^n'='v2000^v/ fi),
iso=	,(if v2000^n='iso' then v2000^n'='v2000^v/ fi),
outiso=	,(if v2000^n='outiso' then v2000^n'='v2000^v/ fi),
outisotag1=	,(if v2000^n='outisotag1' then v2000^n'='v2000^v/ fi),
fix=	,(if v2000^n='fix' then v2000^n'='v2000^v/ fi),
ln1=	,(if v2000^n='ln1' then v2000^n'='v2000^v/ fi),
ln2=	,(if v2000^n='ln2' then v2000^n'='v2000^v/ fi),
fst=	,(if v2000^n='fst' then v2000^n'='v2000^v/ fi),
fst/h=	,(if v2000^n='fst/h' then v2000^n'='v2000^v/ fi),
stw=	,(if v2000^n='stw' then v2000^n'='v2000^v/ fi),
ifupd=	,(if v2000^n='ifupd' then v2000^n'='v2000^v/ fi),

Parâmetro	Formato de extração
ifupd/create=	,(if v2000^n='ifupd/create' then v2000^n'='v2000^v/ fi),
ifupd/dict=	,(if v2000^n='ifupd/dict' then v2000^n'='v2000^v/ fi),
ifupd/create/dict=	,(if v2000^n='ifupd/create/dict' then v2000^n'='v2000^v/ fi),
fullinv=	,(if v2000^n='fullinv' then v2000^n'='v2000^v/ fi),
fullinv/dict=	,(if v2000^n='fullinv/dict' then v2000^n'='v2000^v/ fi),
fullinv/keep=	,(if v2000^n='fullinv/keep' then v2000^n'='v2000^v/ fi),
fullinv/m=	,(if v2000^n='fullinv/m' then v2000^n'='v2000^v/ fi),
fullinv/ansi=	,(if v2000^n='fullinv/ansi' then v2000^n'='v2000^v/ fi),
fullinv/dict/keep=	,(if v2000^n='fullinv/dict/keep' then v2000^n'='v2000^v/ fi),
fullinv/dict/m=	,(if v2000^n='fullinv/dict/m' then v2000^n'='v2000^v/ fi),
fullinv/dict/ansi=	,(if v2000^n='fullinv/dict/ansi' then v2000^n'='v2000^v/ fi),
fullinv/dict/m/ansi=	,(if v2000^n='fullinv/dict/m/ansi' then v2000^n'='v2000^v/ fi),
fullinv/ansi=	,(if v2000^n='fullinv/ansi' then v2000^n'='v2000^v/ fi),
fullinv/m/ansi=	,(if v2000^n='fullinv/m/ansi' then v2000^n'='v2000^v/ fi),
fullinv/keep/ansi=	,(if v2000^n='fullinv/keep/ansi' then v2000^n'='v2000^v/ fi),
fullinv/keep/m/ansi=	,(if v2000^n='fullinv/keep/m/ansi' then v2000^n'='v2000^v/ fi),
maxlk1=	,(if v2000^n='maxlk1' then v2000^n'='v2000^v/ fi),
maxlk2=	,(if v2000^n='maxlk2' then v2000^n'='v2000^v/ fi),
copy=	,(if v2000^n='copy' then v2000^n'='v2000^v/ fi),
append=	,(if v2000^n='append' then v2000^n'='v2000^v/ fi),
updatf=	,(if v2000^n='updatf' then v2000^n'='v2000^v/ fi),
create=	,(if v2000^n='create' then v2000^n'='v2000^v/ fi),
merge=	,(if v2000^n='merge' then v2000^n'='v2000^v/ fi),