BIREME / OPAS / OMS

Centro Latino-Americano e do Caribe de Informação em Ciências da Saúde

Manual de Referência da Linguagem IsisScript

Versão 1.1

São Paulo - 2006

Manual de Referência da Linguagem IsisScript

Ficha Catalográfica

BIREME / OPAS / OMS (Brasil)

Manual de Referência da Linguagem IsisScript. / BIREME (org.). São Paulo : BIREME / OPAS / OMS, 2006.

80 p.

1. Manual do usuário. 2. Acesso à informação. 3. Sistemas de informação. 4. Gerenciamento de informação. 5. Saúde Pública. 6. Serviços de saúde. 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, citados ou não.

BIREME / OPAS / OMS

Centro Latino-Americano e do Caribe de Informação em Ciências da Saúde

Rua Botucatu 862 - Vila Clementino

Este documento foi produzido com a Metodologia para Normalização de Documentos (NorDoc) desenvolvida pela BIREME.

Sumário

Abreviaturas utilizadas

- **ANSI**. American National Standards Institute [Instituto Nacional Americano de Normas].
- **ASCII**. American Standard Code for Information Interchange [Código Padrão Americano para Intercâmbio de Informações].
- **BIREME**. Centro Latino-Americano e do Caribe de Informação em Ciências da Saúde.
- **BVS**. Biblioteca Virtual em Saúde.
- **CGI**. Common Gateway Interface [Interface Comum de Passagem].
- **FST**. Field Selection Table [Tabela de Seleção de Campo].
- **HTML**. HyperText Markup Language [Linguagem de Marcação de Hipertexto].
- **HTTP**. HyperText Transfer Protocol [Protocolo de Transferência de Hipertexto].

- **ISO**. International Organization for Standardization [Organização Internacional para Padronização].
- **MFN**. Master File Number [Número mestre de arquivo].
- **OMS**. Organização Mundial da Saúde.
- **OPAS**. Organização Pan-Americana de Saúde.
- **STW**. STop Word file [Arquivo de palavras proibidas].
- **UNESCO**. United Nations Educational, Scientific and Cultural Organization [Organização das Nações Unidas para a Educação, a Ciência e a Cultura].
- **URL**. Universal Resource Locator [Localizador Universal de Recurso].

Como usar este manual

Este manual está organizado em um único capítulo contendo todos os comandos, parâmetros e opções disponíveis na linguagem IsisScript.

Os comandos estão organizados no nível hierárquico 2 e seus elementos seguem a hierarquia até o nível 4.

Cada comando,. parâmetro ou opção conterá uma breve descrição de seu propósito. Informação adicional como atributos, sintaxe, elementos contidos e onde se aplica também são fornecidos, completando um exemplo de código para ilustrar o comando, opção ou parâmetro.

Os únicos pré-requisitos ao usuário são o conhecimento de linguagem de formato e o modelo e estrutura de uma base CDS/ISIS padrão.

Prefácio

Sobre a 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.

Referência do IsisScript

<IsisScript>

1. Pode Conter : <function> <section> <trace>
2. Atributos : name
3. Sintaxe : <IsisScript> ... </IsisScript>
   O elemento **<IsisScript>** é usado para indicar um grupo de instruções IsisScript.
   O atributo *name* é usado para nomear e documentar o grupo.

Exemplo

```
<IsisScript name=HelloWorld>
 <section>
 <display>Hello World!</display>
 </section>
</IsisScript>
```

IsisScript.name

1. Pode Ser Usado Em : <call> <function> <IsisScript> <section>
2. Sintaxe : name=...
   O atributo **name** é opcional, quando usado serve apenas para fins de documentação do IsisScript.

Exemplo

```
<IsisScript name=HelloWorld>
 <section>
 <display>Hello World!</display>
 </section>
</IsisScript>
```

<call>

1. Pode conter : <pft>
2. Pode ser usado em : <do> <function> <hl> <loop> <section> <update>
3. Atributos : name
4. Sintaxe : <call> ... </call>

   O elemento **<call>** indica a chamada de uma função. A função a ser chamada é especificada pelo atributo *name*.

   O argumento do elemento **<call>** é passado como parâmetro para a função.

Exemplo

```
<IsisScript>
 <function name="First">
 <display>FIRST </display>
 </function>
 <function name="Second">
 <display>SECOND </display>
 </function>
 <function name="ParamTest" action="replace" tag="1" split="occ">
 <display><pft>##'ParamTest'/</pft></display>
 <display><pft>ALL</pft></display>
 <return action="replace" tag="9999" split="occ"><pft>(v1/)</pft></return>
 </function>
 <section>
 <call name="First">now</call>
 <call name="Second">now</call>
 <call name="ParamTest"><pft>'One'/'Two'/</pft></call>
 <display><pft>ALL</pft></display>
 </section>
</IsisScript>
```

call.name

1. Pode ser usado em: <call> <function> <IsisScript> <parm> <section>
2. Sintaxe: name=...
   Nome da função a ser chamada.

Exemplo

```
<IsisScript>
 <function name="First">
 <display>FIRST </display>
 </function>
 <function name="Second">
 <display>SECOND </display>
 </function>
 <function name="ParamTest" action="replace" tag="1" split="occ">
 <display><pft>##'ParamTest'/</pft></display>
 <display><pft>ALL</pft></display>
 <return action="replace" tag="9999" split="occ"><pft>(v1/)</pft></return>
 </function>
 <section>
 <call name="First">now</call>
 <call name="Second">now</call>
 <call name="ParamTest"><pft>'One'/'Two'/</pft></call>
 <display><pft>ALL</pft></display>
 </section>
</IsisScript>
```

<cgitable>

1. Pode conter: <pft>
2. Pode ser usado em: <do> <function> <hl> <loop> <section> <update>
3. Sintaxe: <cgitable> ... </cgitable>
   Para cada linha do argumento especificado, adiciona ao campo especificado o conteúdo do CGI ("value") correspondente ao "name" especificado. Cada linha do argumento equivale à definição <field action="cgi" tag="...">*name*</field>

Exemplo

```
...
 <cgitable>
```

```
 <pft>'2001 db'/
 '2002 from'/
 ref(['CONFIG']1,(v1^t,x1,v1^n/))/
 </pft>
 </cgitable>
...
```

<define>

1. Pode conter: <pft>
2. Pode ser usado em: <do> <function> <hl> <loop> <section> <update>
3. Sintaxe: <define> ... </define>

   Define campos que serão usados dentro do elemento <*loop*>. Cada linha equivale à definição <field action="define" tag="...">*Isis_*...</field>

Exemplo

```
...
 <define>
 <pft>'1001 Isis_Current'/
 '2002 Isis_Total'/
 </pft>
 </define>
...
```

<display>

1. Pode conter: <htmlpft> <pft>
2. Pode ser usado em: <do> <function> <hl> <loop> <section> <update>
3. Sintaxe: <display> ... </display>

   Envia um texto para a saída corrente. O IsisScript começa tendo como saída corrente o "standard output" do sistema operacional.

   O redirecionamento da saída é informado através do elemento <*file*>.

Exemplo

```
<IsisScript>
<section>
 <display><pft>'Content-type: text/html'/#</pft></display>
 <display>Hello World!<br></display>
```

```
 <field action=cgi tag=100>^n^v</field>
 <display><pft>(|100=|v100|<br>|/)</pft></display>
</section>
</IsisScript>
```

<do>

1. Pode conter: <call> <cgitable> <define> <display> <do> <export> <extract> <field> <file> <flow> <hl> <label> <list> <loop> <parm> <proc> <return> <trace> <update>
2. Pode ser usado em: <do> <function> <hl> <loop> <section> <update>
3. Atributos: task
4. Sintaxe: <do> ... </do>

   O elemento **<do>** informa o início de uma tarefa IsisScript. As tarefas possíveis são: intervalo de registros, intervalo de chaves, pesquisa, repetição, lista, importação de registros, ordenação de bases de dados, carga de arquivo invertido e geração completa de arquivo invertido. A tarefa de lista pode ser de freqüencia, ordenação ou simples.

   O atributo *task* é usado para informar o tipo de tarefa a ser executada.

   Os parâmetros necessários para a execução da tarefa são informados através do elemento <*parm*>.

   O elemento <*loop*> informa o bloco de instruções a serem executadas para cada item da tarefa.

Exemplo

```
...
 <do task="search">
 <parm name="db">CDS</parm>
 <parm name="expression">plants * water</parm>
 <loop>
 <display><pft>mfn/</pft></display>
 </loop>
 </do>
...
```

do.task

1. Opções: fullinvertion import invertedload keyrange list mastersort mfnrange search update
2. Pode ser usado em: <do>
3. Sintaxe: task=...

Informa a tarefa geradora de registros que serão utilizados pelo elemento *<loop>*.

A ausência do atributo *task* indica uma repetição independente de tarefa.

Exemplo

```
...
 <do>
 <parm name=to>500</field>
 <field action=define tag=1001>Isis_Current</field>
 <field action=define tag=1002>Isis_Total</field>
 <loop>
 <display>
 <pft>newline('<br>'),v1001,'/',v1002/</pft>
 </display>
 </loop>
 </do>
...
```

do.task=fullinvertion

Faz a geração completa do arquivo invertido.

A base de dados é informada pelo elemento *<parm name=db>*.

A FST (Field Select Table) é informada pelo elemento *<parm name=fst>*.

Exemplo

```
...
 <do task=fullinvertion>
 <parm name=db><pft>v2001</pft></parm>
 <parm name=fst><pft>v2061</pft></parm>
 <field action=define tag=1102>Isis_Status</field>
 <display><pft>'Full invertion: ',v2001/</pft></display>
 <loop></loop>
 <display><pft>'Finished.'/</pft></display>
 <display><pft>'Lock status = 'v1102/</pft></display>
 </do>
...
```

do.task=import

Acessa dados de um arquivo texto e carrega na forma de registros de base de dados.

O nome do arquivo é informado através do elemento <*parm name=file*>.

O elemento <*parm name=type*> pode ser usado para informar o tipo do arquivo texto; arquivo texto *ISO 2709* é assumido se o tipo não for informado.

Exemplo

```
...
 <do task=import>
 <parm name=file><pft>v2041</pft></parm>
 <parm name=type><pft>v2042</pft></parm>
 <loop>
 <display><pft>ALL</pft></display>
 </loop>
 </do>
...
```

do.task=invertedload

Faz a carga do arquivo invertido através de uma base de dados que contém as chaves ordenadas.

A base de dados a ser carregada é informada pelo elemento <*parm name=db*>.

A base de dados que contém as chaves ordenadas é informada pelo elemento <*parm name=keysdb*>.

Exemplo

```
...
 <do task=invertedload>
 <parm name=db>TEST</parm>
 <parm name=keysdb>TESTKEYS</parm>
 <field action=define tag=1>Isis_Posting</field>
 <field action=define tag=2>Isis_Key</field>
 <field action=define tag=1102>Isis_Status</field>
 <display><pft>'Inverted load ...'/</pft></display>
 <loop></loop>
 <display><pft>'Lock status = 'v1102/</pft></display>
 <flow action=exit>
 <pft>if val(v1102) <> 0 then v1102 fi</pft>
```

```
 </flow>
 <display><pft>'Inverted load: TEST loaded.'/</pft></display>
 </do>
...
```

do.task=keyrange

Percorre uma lista de chaves do arquivo invertido de uma base de dados.

A base de dados é informada pelo elemento <*parm name=db*>.

A chave inicial é informada pelo elemento <*parm name=from*>.

A chave final é informada pelo elemento <*parm name=to*>.

A quantidade de chaves pode ser limitada pelo elemento <*parm name=count*>.

O elemento <*parm name=reverse*> pode ser usado para percorrer as chaves em ordem reversa.

O elemento <*parm name=posting*> pode ser usado para percorrer também a lista de posting de cada chave.

O elemento <*parm name=posttag*> pode ser usado para percorrer a lista de postings de cada chave somente para o tag especificado.

Exemplo

```
...
 <do task=keyrange>
 <parm name=db><pft>v2001</pft></parm>
 <parm name=from>
 <pft>v2002,if a(v2002) then v2101 fi</pft>
 </parm>
 <parm name=count><pft>v2003,"20"n2003</pft></parm>
 <parm name=reverse><pft>v2004</pft></parm>
 <parm name=to><pft>v2006</pft></parm>
 <field action=define tag=1001>Isis_Current</field>
 <field action=define tag=1>Isis_Key</field>
 <field action=define tag=2>Isis_Postings</field>
 <display><pft>'##) POSTINGS',c15,'KEY'/#</pft></display>
 <loop>
 <display>
 <pft>f(val(v1001),2,0),') ',v2,c15,v1/</pft>
 </display>
 <field action=export tag=1031>
 <pft>if val(v1001) = 1 then '1' fi</pft>
```

```
 </field>
 <field action=export tag=1032>1</field>
 </loop>
 <display>
 <pft>'***************'/,
 f(val(v1001),2,0),') ',v1031,' / ',v1032/
 </pft>
 </display>
 </do>
...
```

do.task=list

Percorre a lista de itens previamente carregados através do elemento <*list action=load type=...*>.

A lista pode ser ordenada através do elemento <*parm name=sort*>.

O item inicial é informado pelo elemento <*parm name=from*>.

O item final é informado pelo elemento <*parm name=to*>.

A quantidade de registros pode ser limitada pelo elemento <*parm name=count*>.

O elemento <*parm name=reverse*> pode ser usado para percorrer a lista em ordem reversa.

Exemplo

```
...
 <list action=load type=sort><pft>'1'/,'2'/,'3'/</pft></list>
 <list action=load type=sort><pft>'9'/,'8'/,</pft></list>
 <list action=load type=sort><pft>'F'/,'Z'/,'A'/</pft></list>
 <do task=list>
 <field action=define tag=1001>Isis_Current</field>
 <field action=define tag=1002>Isis_Items</field>
 <field action=define tag=1>Isis_Item</field>
 <loop>
 <display>
 <pft>v1001,'/',v1002,c10,v1/)</pft>
 </display>
 </loop>
 </do>
...
```

do.task=mastersort

Ordena os registros de uma base de dados.
A base de dados é informada pelo elemento <*parm name=db*>.
A chave para ordenação é informada pelo elemento <*parm name=key*>.
O tamanho da chave é informado pelo elemento <*parm name=keylength*>.

Exemplo

```
...
 <do task=mastersort>
 <parm name=db>CDS</parm>
 <parm name=key><pft>v24</pft></parm>
 <parm name=keylength>100</parm>
 <field action=define tag=1102>Isis_Status</field>
 <display><pft>'Key sort ...'/</pft></display>
 <loop></loop>
 <display><pft>'Lock status = 'v1102/</pft></display>
 <flow action=exit>
 <pft>if val(v1102) <> 0 then v1102 fi</pft>
 </flow>
 <display><pft>'Key sort: CDS sorted.'/</pft></display>
 </do>
...
```

do.task=mfnrange

Percorre uma lista de registros de uma base de dados.
A base de dados é informada pelo elemento <*parm name=db*>.
O MFN inicial é informado pelo elemento <*parm name=from*>.
O MFN final é informado pelo elemento <*parm name=to*>.
A quantidade de MFNs pode ser limitada pelo elemento <*parm name=count*>.
O elemento <*parm name=reverse*> pode ser usado para percorrer os MFNs em ordem reversa.

Exemplo

```
...
 <do task=mfnrange>
 <parm name=db>CDS</parm>
 <parm name=from>25</parm>
 <parm name=count>10</parm>
```

```
 <loop>
 <display><pft>ALL</pft></display>
 </loop>
 </do>
...
```

do.task=search

Pesquisa uma base de dados e percorre a lista de registros encontrados.

A base de dados é informada pelo elemento <*parm name=db*>.

A expressão de pesquisa é informada pelo elemento <*parm name=expression*>.

O registro inicial é informado pelo elemento <*parm name=from*>.

O registro final é informado pelo elemento <*parm name=to*>.

A quantidade de registros pode ser limitada pelo elemento <*parm name=count*>.

O elemento <*parm name=reverse*> pode ser usado para percorrer os registros em ordem reversa.

Exemplo

```
...
 <do task="search">
 <parm name="db">CDS</parm>
 <parm name="expression">plants * water</parm>
 <loop>
 <display><pft>mfn/</pft></display>
 </loop>
 </do>
...
```

do.task=update

Atualiza ou acrescenta um registro na base de dados.

A base de dados é informada através do elemento <*parm name=db*>.

O elemento <*parm name=mfn*> é usado para informar o MFN a ser atualizado ou se é para ser acrescentado um novo registro.

O elemento <*parm name=lockid*> é usado para identificar o "dono" do registro.

O elemento <*parm name=expire*> pode ser usado para informar que o "dono" do registro teve seu tempo, para permanecer com o registro bloqueado, expirado.

Exemplo

```
...
 <do task=update>
 <parm name=db>CDS</parm>
 <parm name=mfn>New</parm>
 <field action=define tag=1102>Isis_Status</field>
 <update>
 <field action=replace tag=20>
 <pft>date</pft>
 </field>
 <write>Unlock</write>
 <display><pft>ALL</pft></display>
 </update>
 </do>
...
```

<export>

1. Pode conter: <pft>
2. Pode ser usado em: <do> <function> <hl> <loop> <section> <update>
3. Sintaxe: <export> ... </export>

   Adiciona o registro corrente em um arquivo de texto.

   O nome do arquivo deve ter sido previamente informado através do elemento <*parm file*=...>.

   O tipo do arquivo padrão é o ISO2709; para gerar outro tipo de arquivo exportação informe previamente com o elemento <*parm type*=...>. Os tipos de arquivo de exportação são: *ISO2709*, *HLine* e *VLine*.

Exemplo

```
...
 <do task=mfnrange>
 <parm name=db>CDS</parm>
 <parm name=file>CDS.ISO</parm>
 <loop>
 <export>this</export>
 </loop>
 </do>
...
```

<extract>

1. Pode conter: <pft>
2. Pode ser usado em: <do> <function> <hl> <loop> <section> <update>
3. Sintaxe: <extract> ... </extract>

   Adiciona as chaves extraídas do registro corrente na base de dados de chaves.

   A FST (Field Select Table) deve ser previamente informada através do elemento <*parm name=fst*>.

   A base de dados que conterá as chaves deve ser previamente informada através do elemento <*parm name=keysdb*>.

   O campo que conterá as chaves deve ser previamente informado através do elemento <*field action=define tag=...*>*Isis_Key*</*field*>.

   O campo que conterá os dados sobre o posting deve ser previamente informado através do elemento <*field action=define tag=...*>*Isis_Posting*</*field*>.

Exemplo

```
...
 <do task=mfnrange>
 <parm name=fst>1 0 v1</parm>
 <parm name=keysdb>tmp1</parm>
 <field action=define tag=1>Isis_Posting</field>
 <field action=define tag=2>Isis_Key</field>
 <loop>
 <extract>this</extract>
 </loop>
 </do>
...
```

<field>

1. Pode conter: <pft>
2. Pode ser usado em: <do> <function> <hl> <loop> <section> <update>
3. Atributos: action from previous split tag type
4. Sintaxe: <field> ... </field>

   O elemento **<field>** é usado para adicionar, modificar, deletar, importar, exportar e definir campos.

   O atributo *action* informa o uso do elemento.

   O atributo *tag* informa o campo a ser afetado.

   O atributo *split* pode ser usado para informar que cada linha será uma nova ocorrência do campo.

O atributo *previous* informa se o conteúdo anterior será eliminado ou não.
O atributo *type* informa o tipo de campo a ser acessado.
O atributo *from* informa o número do campo a ser acessado.

Exemplo

```
...
 <field action=replace tag=2><pft>v400^b</pft></field>
...
```

field.action

1. Opções: add cgi define delete export hl import occ replace statusdb statusfile
2. Pode ser usado em: <field> <file> <flow> <function> <htmlpft> <list> <return>
3. Sintaxe: action=...
   Indica a ação que o elemento <*field*> irá executar.

Exemplo

```
...
 <field action=replace tag=2><pft>v400^b</pft></field>
...
```

field.action=add

Acrescenta uma nova occorrência ao campo especificado pelo atributo **tag**.
O argumento do elemento contém o dado a ser adicionado.

Exemplo

```
...
 <field action=add tag=2>A</field>
 <field action=add tag=2>B</field>
 <field action=add tag=2>C</field>
...
```

field.action=cgi

Adiciona o conteúdo do campo especificado pelo atributo **tag** com o valor correspondente do CGI.
O argumento do elemento <*field*> indica qual é o item do CGI cujo valor será adicionado.

Exemplo

```
...
 <field action=cgi tag=2>phone</field>
...
```

field.action=define

Define o número do campo especificado pelo atributo **tag** como sendo de atualização automática do IsisScript.

O argumento do elemento <*field*> indica que tipo de informação será armazenada no campo.

Os seguintes argumentos podem ser definidos:

Isis_Current - Índice da execução corrente do <*loop*>

Isis_Total - Total de vezes possível para o <*loop*>

Isis_From - Parâmetro *from* do <*loop*>

Isis_To - Parâmetro *to* do <*loop*>

Isis_Lock - Campo de armazenamento do controle de bloqueio de registros

Isis_Status - Armazenameto do estado da execução da tarefa

Isis_Item - Item da lista

Isis_Value - Quantidade de itens calculados pela contagem de freqüencia

Isis_Key - Chave corrente

Isis_Posting - Dados sobre o posting corrente

Isis_Postings - Total de postings da chave corrente

Isis_Items - Total de itens na lista

Isis_ErrorInfo - Apontador de erro na pesquisa

Isis_Keys - Chaves para destacar texto

Isis_MFN - Número do campo de armazenamento do MFN para exportação ou importação de registros

Isis_RecordStatus - Número do campo que conterá o estado do registro

Exemplo

```
... <field action=define tag=1001>Isis_Current</field>...
```

field.action=delete

Elimina uma ou todas as ocorrências do campo especificado pelo atributo **tag**.

O argumento do elemento <*field*> indica a ocorrência a ser eliminada, use *ALL* no argumento para indicar todas as ocorrências.

Exemplo

```
...
 <field action=delete tag=5>ALL</field>
 <field action=delete tag=6>1</field>
```

```
...
```

field.action=export

Modifica o conteúdo de um ou mais campos no escopo anterior do IsisScript.

O campo é especificado pelo atributo **tag**.

Use *tag=list* para indicar que o argumento contém a lista de campos a ser exportada.

Exemplo

```
...
 <field action=export tag=2205>5</field>
 <field action=export tag=list>6,7,21/29</field>
...
```

field.action=hl

Substitui o conteúdo do campo especificado pelo atributo **tag** com um novo valor utilizando esquema de destacar texto.

Exemplo

```
...
 <parm name=prefix><b></parm>
 <parm name=suffix></b></parm>
 <parm name=keys><pft>(v1022/)</pft></parm>
 <field action=hl tag=70><pft>(v70/)</pft></field>
...
```

field.action=import

Modifica o conteúdo de um ou mais campos no escopo corrente do IsisScript copiando os campos do escopo anterior.

O campo é especificado pelo atributo **tag**.

Use *tag=list* para indicar que o argumento contém a lista de campos a ser importada.

Exemplo

```
...
 <field action=import tag=2070>70</field>
 <field action=import tag=list>201,206,301/314,321</field>
...
```

field.action=occ

Modifica o conteúdo do campo especificado pelo atributo *tag* com o conteúdo do campo especificado pelo atributo *from*, somente com a occorrência especificada pelo argumento do elemento.

Exemplo

```
...
 <do>
 <parm name=to><pft>f(nocc(v70),1,0)</pft></parm>
 <field action=define tag=1001>Isis_Current</field>
 <loop>
 <field action=import tag=70>70</field>
 <field action=occ tag=1 from=70><pft>v1001</pft></field>
 ...
 </loop>
 </do>
...
```

field.action=replace

Substitui o conteúdo do campo especificado pelo atributo **tag**.

O argumento contém o novo conteúdo.

Exemplo

```
...
 <field action=replace tag=2><pft>v400^b</pft></field>
...
```

field.action=statusdb

Substitui o conteúdo do campo especificado pelo atributo **tag** com o estado da base de dados informada pelo argumento do elemento <*field*>.

Se existir o master ou o arquivo invertido ou ambos, o campo será criado contendo o subcampo **s** (^s). O subcampo **s** conterá o caracter *m*, se o master existir, e o caracter *i*, se o invertido existir.

Se o master existir, o campo conterá também o subcampo **n** (^n) com o total de registros na base de dados mais um, o subcampo **d** (^d) com o número de bloqueio de entrada de dados (Data Entry Lock), e o subcampo **e** (^e) com o número de bloqueio exclusivo (Exclusive Write Lock).

Exemplo

```
...
 <field action=statusdb tag=1091><pft>v2001</pft></field>
 <flow action=jump>
 <pft>if v1091^s : 'm' then 'LABEL_OK' fi</pft>
 </flow>
...
```

field.action=statusfile

Substitui o conteúdo do campo especificado pelo atributo **tag** com o estado do arquivo informado pelo argumento do elemento <*field*>.

Após a execução, o campo conterá o subcampo **s** (^s) com o caracter "*e*" se o arquivo existir, caso contrário o campo ficará ausente.

Exemplo

```
...
 <field action=statusfile tag=1091>C:\AUTOEXEC.BAT</field>
 <flow action=jump>
 <pft>if v1091^s : 'e' then 'LABEL_OK' fi</pft>
 </flow>
...
```

field.from

1. Pode ser usado em: <field>
2. Sintaxe: from=...

   Especifica o número do campo que será acessado pelo atributo *action=occ*.

Exemplo

```
...
 <do>
 <parm name=to><pft>f(nocc(v70),1,0)</pft></parm>
 <field action=define tag=1001>Isis_Current</field>
 <loop>
 <field action=import tag=70>70</field>
 <field action=occ tag=1 from=70><pft>v1001</pft></field>
 ...
 </loop>
 </do>
...
```

field.previous

1. Opções: add delete
2. Pode ser usado em: <field>
3. Sintaxe: previous=...

   Especifica se o campo sendo importado ou exportado terá seu conteúdo anterior eleiminado ou se novas ocorrências serão adicionadas.

   A ausência deste atributo significa que o conteúdo anterior será eliminado.

Exemplo

```
...
 <field action=import tag=1 previous=add>200</field>
...
```

field.previous=add

Adiciona novas ocorrências.

Exemplo

```
...
 <field action=export tag=200 previous=add>1</field>
...
```

field.previous=delete

Informa que é para eliminar as ocorrências anteriores.

Exemplo

```
...
 <field action=export tag=4001 previous=delete>1</field>
...
```

field.split

1. Opções: flddir occ
2. Pode ser usado em: <field>
3. Sintaxe: split=...

   Indica a forma que o dado será armazenado.

Exemplo

```
...
```

```
<field action=replace tag=1 split=occ><pft>(v200/)</pft></field>
...
```

field.split=flddir

O texto a ser armazenado no campo especificado pelo atributo **tag** é o diretório de campos do registro com os respectivos conteúdos.

Cada linha contém o número do campo (5 dígitos), um espaço em branco e o conteúdo do campo.

Exemplo

```
...
<do task="mfnrange">
<parm name="db">CDS</parm>
<parm name="count">5</parm>
<loop>
<field action="replace" tag="1" split="flddir">ALL</field>
<display><pft>ALL</pft></display>
</loop>
</do>
...
```

field.split=occ

Indica que cada linha do argumento do elemento <*field*> será armazenada em uma nova ocorrência do campo especificado pelo atributo *tag*.

Exemplo

```
...
<field action=replace tag=1 split=occ><pft>(v200/)</pft></field>
...
```

field.tag

1. Opções: list
2. Pode ser usado em: <field> <parm>
3. Sintaxe: tag=...

   O atributo **tag** é usado para especificar o número do campo.

   Use **tag=list** para informar que a lista de campos será passada pelo argumento do elemento <*field*>.

Exemplo

```
...
 <field action=replace tag=2><pft>v400^b</pft></field>
...
```

field.tag=list

Informa que a lista de campos será passada pelo argumento do elemento <*field*>.

Use a vírgula "**,**" para separar a lista de campos. Use a barra "**/**" para indicar um intervalo de campos. Use o abre colchete "**[**" para informar que o campo será armazenado com o número de campo especificado depois do caracter dois pontos "**:**" e antes do fecha colchete "**]**".

Exemplo

```
...
 <field action=import tag=list>1,2,3,11/19,[30:20]</field>
...
```

field.type

1. Opções: flag
2. Pode ser usado em: <field> <file> <htmlpft> <list> <pft>
3. Sintaxe: type=...

   Informa o tipo do dado que está sendo acessado.

Exemplo

```
...
 <field action=cgi tag=2011 type=flag>trace</field>
...
```

field.type=flag

Informa que o campo que está sendo acessado do CGI é do tipo presente / ausente.

Se estiver presente e vazio o conteúdo do campo será armazenado com *On*, senão o campo será armazenado com o valor passado.

Este atributo somente tem efeito com o atributo *action=cgi*.

Exemplo

```
...
```

```
 <field action=cgi tag=2011 type=flag>trace</field>
...
```

<file>

1. Pode conter: <pft>
2. Pode ser usado em: <do> <function> <hl> <loop> <section> <update>
3. Atributos: action type
4. Sintaxe: <file> ... </file>

   O elemento **<file>** pode ser usado para criar, desbloquear e fechar bases de dados, criar arquivos temporários, deletar arquivos e mudar a saída padrão do IsisScript.

   O atributo *action* informa a ação.

   O atributo *type* informa o tipo de arquivo que sofrerá a ação.

Exemplo

```
...
 <file action=create type=database>TESTX</file>
...
```

file.action

1. Opções: append close create delete unlock
2. Pode ser usado em: <field> <file> <flow> <function> <htmlpft> <list> <return>
3. Sintaxe: action=...

   Informa a ação a ser executada pelo elemento <*file*>.

Exemplo

```
...
 <file action=create type=database>TESTX</file>
...
```

file.action=append

Abre um arquivo de saída para adiconar os textos no final.

Exemplo

```
...
 <file action=append type=output>TEST.LOG</file>
...
```

file.action=close

Fecha o arquivo de saída quando o atributo *type=output* ou uma base de dados quando o atributo *type=database*.

Exemplo

```
...
 <file action=close type=output>TEST.LOG</file>
...
```

file.action=create

Combinado com o atributo *type=output* a ação é criar um novo arquivo de saída. Combinado com o atributo *type=database* a ação é inicializar uma base de dados.

Exemplo

```
...
 <file action=create type=database>TESTX</file>
...
```

file.action=delete

Deleta um arquivo.

Exemplo

```
...
 <file action=delete type=file>TESTX</file>
...
```

file.action=unlock

Desbloqueia uma base de dados.

Os valores do bloqueio de entrada de dados (Data Entry Lock) e o bloqueio exclusivo (Exclusive Write Lock) são zerados.

Os registros permanecem inalterados.

Exemplo

```
...
 <file action=unlock type=database>CDS</file>
...
```

file.type

1. Opções: database file inverted master output tempfile
2. Pode ser usado em: <field> <file> <htmlpft> <list> <pft>
3. Sintaxe: type=...
   Informa o tipo de arquivo alvo da ação.

Exemplo

```
...
 <file action=create type=database>TESTX</file>
...
```

file.type=database

Informa que a ação atua sobre uma base de dados.

Exemplo

```
...
 <file action=create type=database>TESTX</file>
...
```

file.type=file

A ação incide sobre um arquivo.
Válido somente para a ação *action=delete*.

Exemplo

```
...
 <file action=delete type=file>TEST.LOG</file>
...
```

file.type=inverted

Informa que a ação atua sobre o arquivo invertido de uma base de dados.
Válido somente para a ação *action=create*.

Exemplo

```
...
 <file action=create type=inverted>TESTX</file>
...
```

file.type=master

Informa que a ação atua sobre o arquivo master de uma base de dados.
Válido somente para a ação *action=create*.

Exemplo

```
...
 <file action=create type=master>TESTX</file>
...
```

file.type=output

Informa que a ação atua sobre o arquivo de saída.

Exemplo

```
...
 <file action=create type=output>TEST.LOG</file>
...
```

file.type=tempfile

Informa que a ação atua sobre um arquivo temporário.
Válido somente para a ação *action=create*.
O argumento informa o número da campo em que será armazenado o nome do arquivo temporário único retornado pelo sistema operacional.

Exemplo

```
...
 <file action=create type=tempfile>4001</file>
...
```

<flow>

1. Pode conter: <pft>
2. Pode ser usado em: <do> <function> <hl> <loop> <section> <update>
3. Atributos: action

4. Sintaxe: <flow> ... </flow>

   O elemento **<flow>** é usado para desviar a seqüência de execução das instruções do IsisScript.

   O atributo *action* informa a ação.

Exemplo

```
...
 <flow action=jump><pft>if p(v1) then 'GO' fi</pft></flow>
 <display>Field 1 absent</display>
 <flow action=exit>1</flow>
 <label>GO</label>
 <display>Field 1 present, continue</display>
 ...
...
```

flow.action

1. Opções: exit jump skip
2. Pode ser usado em: <field> <field> <flow> <function> <htmlpft> <list> <return>
3. Sintaxe: action=...

   Informa a ação que o elemento <*flow*> deverá seguir.

Exemplo

```
...
 <flow action=jump><pft>if p(v1) then 'GO' fi</pft></flow>
 <display>Field 1 absent</display>
 <flow action=exit>1</flow>
 <label>GO</label>
 <display>Field 1 present, continue</display>
 ...
...
```

flow.action=exit

Termina a execução do IsisScript corrente.

O argumento do elemento <*flow*> informa o código a ser retornado para o sistema operacional.

Exemplo

```
...
```

```
 <flow action=jump><pft>if p(v1) then 'GO' fi</pft></flow>
 <display>Field 1 absent</display>
 <flow action=exit>1</flow>
 <label>GO</label>
 <display>Field 1 present, continue</display>
 ...
...
```

flow.action=jump

Desvia a execução do IsisScript para o elemento <*label*> correspondente.
O argumento da instrução <*flow*> informa o destino.

Exemplo

```
...
 <flow action=jump><pft>if p(v1) then 'GO' fi</pft></flow>
 <display>Field 1 absent</display>
 <flow action=exit>1</flow>
 <label>GO</label>
 <display>Field 1 present, continue</display>
 ...
...
```

flow.action=skip

Desvia a execução do IsisScript para o início do <*loop*> do escopo corrente ou abandona o escopo corrente e volta para o escopo anterior.
O argumento do elemento <*flow*> deve ser *Next* ou *Quit* respectivamente.

Exemplo

```
...
 <do>
 <parm name=db>CDS</db>
 <loop>
 <flow action=skip>
 <pft>if a(v24) then 'Next'
 else if val(v26^c) > 1989 then 'Quit' fi
 fi
 </pft>
 </flow>
```

```
<display><pft>@CDS.PFT</pft></display>
</loop>
</do>
...
```

<function>

1. Pode conter: <call> <cgitable> <define> <display> <do> <export> <extract> <field> <file> <flow> <hl> <label> <list> <parm> <proc> <return> <trace>
2. Pode ser usado em: <IsisScript>
3. Atributos: action from name split tag
4. Sintaxe: <function> ... </function>

   O elemento **<function>** inicia um bloco de declaração de uma função.

   Use os atributos *action*, *tag* e *split* para receber parâmetros conforme descrito para o elemento <*field*>.

Exemplo

```
...
<function name="Test" action="replace" tag="1">
<display>Inside Test function<br></display>
<display><pft>'Field 1 = ',v1</pft></display>
</function>
...
```

function.action

Veja: <field action=...>

Passagem de parâmetros para a função.

Tem a mesma funcionalidade do atributo *action* do elemento <*field*>.

Exemplo

```
...
<function name="Test" action="replace" tag="1">
<display>Inside Test function<br></display>
<display><pft>'Field 1 = ',v1</pft></display>
</function>
...
```

function.from

Veja: <field from=...>
Passagem de parâmetros para a função.
Tem a mesma funcionalidade do atributo *from* do elemento <*field*>.

Exemplo

```
...
 <function name="Test" action="replace" tag="1">
 <display>Inside Test function<br></display>
 <display><pft>'Field 1 = ',v1</pft></display>
 </function>
...
```

function.name

1. Pode ser usado em: <call> <function> <IsisScript> <section>
2. Sintaxe: name=...
   O atributo **name** identifica a função que está sendo declarada.
   Este nome será utilizado pelo elemento <*call*> para chamar a função.

Exemplo

```
...
 <function name="Test" action="replace" tag="1">
 <display>Inside Test function<br></display>
 <display><pft>'Field 1 = ',v1</pft></display>
 </function>
...
```

function.split

Veja: <field split=...>
Passagem de parâmetros para a função.
Tem a mesma funcionalidade do atributo *split* do elemento <*field*>.

Exemplo

```
...
 <function name="Test" action="replace" tag="1">
 <display>Inside Test function<br></display>
 <display><pft>'Field 1 = ',v1</pft></display>
```

```
 </function>
...
```

function.tag

Veja: <field tag=...>
Passagem de parâmetros para a função.
Tem a mesma funcionalidade do atributo *tag* do elemento <*field*>.

Exemplo

```
...
 <function name="Test" action="replace" tag="1">
 <display>Inside Test function<br></display>
 <display><pft>'Field 1 = ',v1</pft></display>
 </function>
...
```

<hl>

1. Pode conter: <call> <cgitable> <define> <display> <do> <export> <extract> <field> <file> <flow> <label> <list> <parm> <proc> <trace>
2. Pode ser usado em: <do> <function> <loop> <section> <update>
3. Sintaxe: <loop> ... </loop>
   O elemento **<hl>** inicia um bloco de instruções para destacar texto.

Exemplo

```
...
 <hl>
 <parm name="prefix"><b></parm>
 <parm name="suffix"></b></parm>
 <parm name="keys"><pft>(v1022/)</pft></parm>
 <field action="hl" tag="18"><pft>v18</pft></field>
 <display><pft>ALL</pft></display>
 </hl>
...
```

<htmlpft>

1. Pode conter: <pft>
2. Pode ser usado em: <display>
3. Atributos: action type
4. Sintaxe: <htmlpft> ... </htmlpft>
   Interpreta e formata um arquivo HTML que contenha instruções da linguagem de formato.

Exemplo

```
...
 <display>
 <htmlpft><pft>cat('Test.htm')</pft></htmlpft>
 </display>
...
```

htmlpft.action

1. Opções: convert
2. Pode ser usado em: <field> <file> <flow> <function> <htmlpft> <list> <return>
3. Sintaxe: action=...
   Especifica uma ação diferente da ação padrão do elemento <*htmlpft*>.

Exemplo

```
...
 <file action=create type=output>TEST.PFT</file>
 <display>
 <htmlpft action=convert>
 <pft>cat('TEST.HTML')</pft>
 </htmlpft>
 </display>
 <file action=close type=output>now</file>
...
```

htmlpft.action=convert

Converte um HTML com especificações de formato para um formato, porém não interpreta o formato convertido, mostra a especificação gerada.

Exemplo

```
...
 <file action=create type=output>TEST.PFT</file>
 <display>
 <htmlpft action=convert>
 <pft>cat('TEST.HTML')</pft>
 </htmlpft>
 </display>
 <file action=close type=output>now</file>
...
```

htmlpft.type

Veja: <pft type=...>

Tipo de ação a ser tomada para execução do formato.

Exemplo

```
...
 <display>
 <htmlpft type=reload>
 <pft>cat('Test.htm')</pft>
 </htmlpft>
 </display>
...
```

<label>

1. Pode ser usado em: <do> <function> <hl> <loop> <section> <update>
2. Sintaxe: <label> ... </label>

   O elemento **<label>** indica um ponto para qual o IsisScript poderá desviar a seqüencia de execução das instruções através do elemento <*flow action=jump*>.

Exemplo

```
...
 <field action=cgi tag=2001>db</field>
 <flow action=jump><pft>if p(v2001) then 'OK' fi</flow>
 <display>db parameter absent, must exit</display>
 <flow action=exit>0</exit>
```

```
<label>OK</label>
<display>db parameter present, continue</display>
...
```

<list>

1. Pode conter: <pft>
2. Pode ser usado em: <do> <function> <hl> <loop> <section> <update>
3. Atributos: action type
4. Sintaxe: <list> ... </list>

   O elemento **<list>** altera a lista interna do IsisScript. As opções são: adicionar itens à lista ou eliminar todos os itens da lista.

   O atributo *action* indica a opção.

   O atributo *type* indica o tipo de lista.

Exemplo

```
...
<list action=load type=freq><pft>(v66/)</pft></list>
...
```

list.action

1. Opções: delete load
2. Pode ser usado em: <field> <file> <flow> <function> <htmlpft> <list> <return>
3. Sintaxe: action=...

   Informa a ação a ser executada na lista do IsisScript.

Exemplo

```
...
<list action=load type=freq><pft>(v66/)</pft></list>
...
```

list.action=delete

Elimina todos os itens da lista.

Exemplo

```
...
<list action=delete>now</list>
```

```
...
```

list.action=load

Adiciona novos itens à lista.
Cada linha informada pelo argumento será um novo item na lista.

Exemplo

```
...
 <list action=load type=freq><pft>(v66/)</pft></list>
...
```

list.type

1. Opções: freq list sort
2. Pode ser usado em: <field> <file> <htmlpft> <list> <pft>

Informa o tipo de armazenamento na lista.

Exemplo

```
...
 <list action=load type=freq><pft>(v66/)</pft></list>
...
```

list.type=freq

Lista de contagem de freqüência de itens.
Um item repetido não é inserido na lista, é adicionado no total de ocorrências daquele item.

Exemplo

```
...
 <list action=load type=freq><pft>(v66/)</pft></list>
...
```

list.type=list

A lista de itens sem ordenação.

Exemplo

```
...
 <list action=load type=list><pft>(v66/)</pft></list>
```

```
...
```

list.type=sort

A lista é ordenada pelo conteúdo do item.

Exemplo

```
...
 <list action=load type=sort><pft>(v66/)</pft></list>
...
```

<loop>

1. Pode conter: <call> <cgitable> <define> <display> <do> <export> <extract> <field> <file> <flow> <hl> <label> <list> <parm> <proc> <return> <trace>
2. Pode ser usado em: <do>
3. Sintaxe: <loop> ... </loop>
   O elemento **<loop>** indica um grupo de instruções que serão repetidas para todos os dados encontrados de acordo com o tipo de tarefa especificada no elemento <*do*> correspondente.

Exemplo

```
...
<do task=search>
 <parm name=db> <pft>v2001</pft></parm>
 <parm name=expression><pft>v2005</pft></parm>
 <loop>
 <display><pft>mfn/</pft></display>
 </loop>
</do>
...
```

<parm>

1. Pode conter: <pft>
2. Pode ser usado em: <do> <function> <hl> <loop> <section> <update>
3. Atributos: name tag type
4. Sintaxe: <parm> ... </parm>

O elemento **<parm>** informa um parâmetro para o bloco de instruções ao qual pertence.

Exemplo

```
...
<do task=search>
 <parm name=db> <pft>v2001</pft></parm>
 <parm name=expression><pft>v2005</pft></parm>
 <loop>
 <display><pft>mfn/</pft></display>
 </loop>
</do>
...
```

parm.name

1. Opções: actab buffersize cipar count db decod delimiter expire expression file freqsum from fst gizmo indexlist key keyfield keylength keys keysdb lockid maxlk mfn posting posttag prefix reset reverse sort stw suffix task to type uctab
2. Pode ser usado em: <call> <function> <IsisScript> <parm> <section>
3. Sintaxe: name=...

   Nome do parâmetro.

Exemplo

```
<IsisScript>
 <section>
 <parm name=actab><pft>cat('ISISAC.TAB')</pft></parm>
 <parm name=uctab><pft>cat('ISISUC.TAB')</pft></parm>
 <parm cipar><pft>'CDS.*=/bases/cds/cds.*'/,
 'ACTAB=/isis/menu/isisac.tab'/</pft>
 'UCTAB=/isis/menu/isisuc.tab'/</pft>
 </parm>
 <do task=search>
 <parm name=db>CDS</parm>
 <parm name=expression>plants*water</parm>
 <parm name=to>10</parm>
 <loop>
 ...
 </loop>
```

```
 </do>
 ...
 </section>
</IsisScript>
```

parm.name=actab

Muda a tabela de caracteres alfabéticos do IsisScript durante a seção corrente.

A tabela de caracteres alfabéticos informa, para a atualização do arquivo invertido e para a extração de chaves, quais caracteres serão considerados como alfabéticos.

Os caracteres que não estiverem na tabela serão considerados como delimitadores.

Em uma seção sem a opção <*parm name=actab*> o IsisScript assume a tabela ANSI.

Exemplo

```
<IsisScript>
 <section>
 <parm cipar><pft>'CDS.*=/bases/cds/cds.*'/,
 'ACTAB=/isis/menu/isisac.tab'/</pft>
 'UCTAB=/isis/menu/isisuc.tab'/</pft>
 </parm>
 <parm name=actab><pft>cat('ACTAB')</pft></parm>
 <parm name=uctab><pft>cat('UCTAB')</pft></parm>
 <do task=search>
 <parm name=db>CDS</parm>
 <parm name=expression>plants*water</parm>
 <parm name=to>10</parm>
 <loop>
 <display><pft>mpu,v24,mpl</pft></display>
 </loop>
 </do>
 ...
 </section>
</IsisScript>
```

parm.name=buffersize

Permite alterar o tamanho do "buffer" interno (em bytes) do WXIS que é usado para armazenar o resultado da formatação.

Exemplo

```
...
 <parm name="buffersize">90000</parm>
...
```

parm.name=cipar

Ativa uma tabela de assinalamentos de nomes lógicos com nomes físicos de arquivos para a seção corrente.

Cada linha contém um assinalamento, à esquerda do caracter = (igual) fica o nome lógico e à direita o nome físico do arquivo.

O caracter * (asterisco) indica que o assinalamento vale para qualquer arquivo de base de dados.

Exemplo

```
...
 <parm name=cipar>
 <pft>
 'CDS.ISO=/bases/cds/cds.iso'/
 'CDS.*=/bases/cds/cds.*'/
 'TEST.PFT=/bases/cds/test.pft'/
 </pft>
 </parm>
...
```

parm.name=count

Limita a quantidade de vezes que o grupo de instruções do elemento <*loop*> será executado.

Exemplo

```
...
 <do task=search>
 <parm name=db> <pft>v2001</pft></parm>
 <parm name=expression><pft>v2005</pft></parm>
 <parm name=from> <pft>v2002</pft></parm>
 <parm name=count>10</parm>
 <loop>
 <display><pft>mfn/</pft></display>
 </loop>
```

```
 </do>
 ...
```

parm.name=db

Informa a base de dados a ser acessada nas seguintes tarefas:
task=mfnrange
task=keyrange
task=search
task=update
task=fullinvertion
task=mastersort
task=invertedload

Exemplo

```
...
<do task=search>
 <parm name=db> <pft>v2001</pft></parm>
 <parm name=expression><pft>v2005</pft></parm>
 <loop>
 <display><pft>mfn/</pft></display>
 </loop>
</do>
...
```

parm.name=decod

Informa a base de dados de parâmetros de expansão de campos decodificados.

Exemplo

```
...
<do task=search>
 <parm name=db> <pft>v2001</pft></parm>
 <parm name=decod> <pft>v2101</pft></parm>
 <parm name=expression><pft>v2005</pft></parm>
 <loop>
 <display><pft>mfn/</pft></display>
 </loop>
</do>
...
```

parm.name=delimiter

Informa o separador de campos para importação de registros na opção *RLine*.
Na ausência deste parâmetro é assumido o caracter | (pipe).

Exemplo

```
...
 <do task="import">
 <parm name="file"><pft>v2041</pft></parm>
 <parm name="type"><pft>v2042</pft></parm>
 <parm name="delimiter"><pft>v2043</pft></parm>
 <loop>
 <display><pft>ALL</pft></display>
 </loop>
 </do>
...
```

parm.name=expire

Informa o tempo máximo que o registro permanecerá bloqueado.

Após esse período o registro pode ser bloqueado por outro usuário (outra identificação).

Exemplo

```
...
<do task=update>
 <field action=cgi tag=2001>db</field>
 <field action=cgi tag=2002>mfn</field>
 <parm name=db><pft>v2001</pft></parm>
 <parm name=mfn><pft>v2002</pft></parm>
 <parm name=expire>14400</parm>
 <field action=define tag=1101>Isis_Lock</field>
 <field action=define tag=1102>Isis_Status</field>
 <update>
 <write>Unlock</write>
 <display><pft>ALL</pft></display>
 <display><pft>'*** LOCK STATUS: 'v1102/</pft></display>
 </update>
</do>
```

```
...
```

parm.name=expression

Informa a expressão de pesquisa.

Exemplo

```
...
<do task=search>
 <parm name=db> <pft>v2001</pft></parm>
 <parm name=expression><pft>v2005</pft></parm>
 <loop>
 <display><pft>mfn/</pft></display>
 </loop>
</do>
...
```

parm.name=file

Informa o nome do arquivo que será importado ou exportado.

Exemplo

```
...
 <do task=import>
 <parm name=file><pft>v2041</pft></parm>
 <parm name=type><pft>v2042</pft></parm>
 <loop>
 <display><pft>ALL</pft></display>
 </loop>
 </do>
...
```

parm.name=freqsum

Informa o valor a ser somado na adição de novos itens na lista de freqüência.

Exemplo

```
...
 <loop>
 <!-- field 1 = Product
```

```
 field 2 = Quantity -->
 <parm name=freqsum><pft>v2</pft></parm>
 <list action=load type=freq><pft>v1</pft></list>
 </loop>
...
```

parm.name=from

Informa qual é o primeiro item a ser acessado pelo grupo do elemento *<loop>*.

Exemplo

```
...
<do task=search>
 <parm name=db> <pft>v2001</pft></parm>
 <parm name=expression><pft>v2005</pft></parm>
 <parm name=from> <pft>v2002</pft></parm>
 <parm name=count> <pft>v2003</pft></parm>
 <loop>
 <display><pft>mfn/</pft></display>
 </loop>
</do>
...
```

parm.name=fst

Informa a FST (Field Select Table) que será usada para atualização do arquivo invertido ou para extração de chaves.

Exemplo

```
...
 <do task=update>
 <parm name=db><pft>v2001</pft></parm>
 <parm name=mfn>New</parm>
 <parm name=fst>1 0 v1</parm>
 <field action=define tag=1102>Isis_Status</field>
 <update>
 <field action=cgi tag=1>Name</field>
 <field action=cgi tag=2>Phone</field>
 <write>Unlock</write>
```

```
 <display><pft>ALL</pft></display>
 </update>
 </do>
...
```

parm.name=gizmo

Informa a base de dados de parâmetros de conversão de conteúdo.

Exemplo

```
...
 <file action=create type=database>GIZMO_DIAC</file>
 <do task=update>
 <parm name=db>GIZMO_DIAC</parm>
 <parm name=mfn>New</parm>
 <field action=define tag=1101>Isis_Status</field>
 <update>
 <field action=replace tag=1>á</field>
 <field action=replace tag=2>á</field>
 <write>Unlock</write>
 </update>
 </do>
 <do task=mfnrange>
 <parm name=db>CDS</parm>
 <parm name=gizmo>GIZMO_DIAC</parm>
 <loop>
 <display><pft>ALL</pft></display>
 </loop>
 </do>
...
```

parm.name=indexlist

Informa a lista de índices para a pesquisa na base de dados.

Exemplo

```
...
 <do task=search>
 <parm name=db>cds</parm>
```

```
 <parm name=indexlist><pft>
 '^p*^ycds^m** '/,
 '^pAU ^ycdsaut^mAU '/,
 '^pTI ^ycdstit^mTI '/
 </pft></parm>
 <parm name=expression>
 Au Mag$ or ([Ti] plants AND water)
 </parm>
 <loop>
 <display><pft>ALL</pft></display>
 </loop>
 </do>
...
```

parm.name=key

Chave para ordenação da base de dados.

Exemplo

```
...
 <do task=mastersort>
 <parm name=db>CDS</parm>
 <parm name=key><pft>v24</pft></parm>
 <parm name=keylength>100</parm>
 <field action=define tag=1102>Isis_Status</field>
 <display><pft>'Sort ...'/</pft></display>
 <loop></loop>
 <display><pft>'Lock status = 'v1102/</pft></display>
 <flow action=exit>
 <pft>if val(v1102) <> 0 then v1102 fi</pft>
 </flow>
 <display><pft>'Sort: CDS sorted.'/</pft></display>
 </do>
...
```

parm.name=keyfield

Especifica o campo que é a chave de ordenação da base de dados.

Exemplo

```
...
 <do task="mastersort">
 <parm name="db">CDS</parm>
 <parm name="keyfield">24</parm>
 <parm name="keylength">200</parm>
 <field action="define" tag="1102">Isis_Status</field>
 <display><pft>'Key sort ...'/</pft></display>
 <loop></loop>
 <display>
 <pft>'Lock status = 'v1102/</pft>
 </display>
 <flow action="exit">
 <pft>if val(v1102) <> 0 then v1102 fi</pft>
 </flow>
 <display>
 <pft>'Key sort: ',v2001,' sorted.'/</pft>
 </display>
 </do>
...
```

parm.name=keylength

Tamanho da chave de ordenação da base de dados.

Exemplo

```
...
 <do task=mastersort>
 <parm name=db>CDS</parm>
 <parm name=key><pft>v24</pft></parm>
 <parm name=keylength>100</parm>
 <field action=define tag=1102>Isis_Status</field>
 <display><pft>'Sort ...'/</pft></display>
 <loop></loop>
 <display><pft>'Lock status = 'v1102/</pft></display>
 <flow action=exit>
 <pft>if val(v1102) <> 0 then v1102 fi</pft>
 </flow>
```

```
 <display><pft>'Sort: CDS sorted.'/</pft></display>
 </do>
...
```

parm.name=keys

Informa a lista de chaves para o esquema de destaque de texto.

Exemplo

```
...
 <do task=search>
 <parm name=db>cds</parm>
 <parm name=expression><pft>v2005</pft></parm>
 <parm name=from><pft>v2002,"1"n2002</pft></parm>
 <parm name=count>10</parm>
 <field action=define tag=1001>Isis_Current</field>
 <field action=define tag=1002>Isis_Total</field>
 <field action=define tag=1022>Isis_Keys</field>
 <loop>
 <hl>
  <parm name=prefix><b></parm>
  <parm name=suffix></b></parm>
  <parm name=keys><pft>(v1022/)</pft></parm>
  <field action=hl tag=24><pft>v24</pft></field>
  <field action=hl tag=70 split=occ><pft>(v70/)</pft></field>
 <display><pft>ALL</pft></display>
 </hl>
 </loop>
 <display><pft>
 if val(v1002) = 0 then 'No record found!' fi
 </pft></display>
 </do>
...
```

parm.name=keysdb

Base de dados que conterá as chaves que serão extraídas, se usado como parâmetro para o elemento <*extract*>.

Base de dados com as chaves ordenadas para carga do arquivo invertido se usado como parâmetro do elemento <*do task=invertedload*>.

Exemplo

```
...
 <do task=invertedload>
 <parm name=db><pft>v2001</pft></parm>
 <parm name=keysdb><pft>v2064</pft></parm>
 <field action=define tag=1>Isis_Posting</field>
 <field action=define tag=2>Isis_Key</field>
 <field action=define tag=1102>Isis_Status</field>
 <display><pft>'Inverted load ...'/</pft></display>
 <loop></loop>
 <display><pft>'Lock status = 'v1102/</pft></display>
 <flow action=exit><pft>
 if val(v1102) <> 0 then v1102 fi
 </pft></flow>
 <display><pft>
 'Inverted load: ',v2001,' loaded.'/
 </pft></display>
 </do>
...
```

parm.name=lockid

Identificador de bloqueio de registro.

Exemplo

```
...
 <do task=update>
 <parm name=db><pft>v2023</pft></parm>
 <parm name=mfn><pft>mfn</pft></parm>
 <field action=define tag=1101>Isis_Lock</field>
 <parm name=lockid>
 <pft>getenv('REMOTE_ADDR'),x1,s(date).8</pft>
 </parm>
 <field action=define tag=1102>Isis_Status</field>
 <update>
 <field action=cgi tag=1>phone</field>
```

```
 <field action=replace tag=1><pft>v1</pft></field>
 <write>Unlock</write>
 <display><pft>ALL</pft></display>
 <display>
 <pft>'*** LOCK STATUS: 'v1102/</pft>
 </display>
 </update>
 </do>
...
```

parm.name=maxlk

Número máximo de chaves (por registro) na extração via FST (Field Select Table).

Na ausência deste parâmetro o IsisScript assume 1024.

Exemplo

```
...
 <do task=fullinvertion>
 <parm name=db><pft>v2001</pft></parm>
 <parm name=fst><pft>v2061</pft></parm>
 <parm name=maxlk>5000</parm>
 <field action=define tag=1102>Isis_Status</field>
 <display><pft>'Full invertion: ',v2001/</pft></display>
 <loop></loop>
 <display><pft>'Finished.'/</pft></display>
 <display><pft>'Lock status = 'v1102/</pft></display>
 </do>
...
```

parm.name=mfn

Número do registro a ser atualizado, ou *New* para indicar que será um novo registro da base de dados, ou *GetNew* para indicar que será um novo registro da base de dados contento os campos importados do registro do escopo anterior do IsisScript.

Exemplo

```
...
 <do task=update>
```

```
<parm name=db><pft>v2023</pft></parm>
<parm name=mfn><pft>mfn</pft></parm>
<field action=define tag=1101>Isis_Lock</field>
<parm name=lockid>
<pft>getenv('REMOTE_ADDR'),x1,s(date).8</pft>
</parm>
<field action=define tag=1102>Isis_Status</field>
<update>
<field action=cgi tag=1>phone</field>
<field action=replace tag=1><pft>v1</pft></field>
<write>Unlock</write>
<display><pft>ALL</pft></display>
<display>
<pft>'*** LOCK STATUS: 'v1102/</pft>
</display>
</update>
</do>
...
```

parm.name=posting

Quantidade de postings para cada chave.
Use *All* para indicar "todos os postings".

Exemplo

```
...
<do task=keyrange>
<parm name=db>CDS</parm>
<parm name=from>PLANTS</parm>
<parm name=count>20</parm>
<parm name=posting>All</parm>
<field action=define tag=1001>Isis_Current</field>
<field action=define tag=1>Isis_Key</field>
<field action=define tag=2>Isis_Postings</field>
<field action=define tag=3>Isis_Posting</field>
<display><pft>
' POSTINGS',c15,'KEY',c46,'POSTING DETAIL'/#
</pft></display>
```

```
 <loop>
 <display><pft>
 f(val(v1001),2,0),') ',v2,c15,v1,c46,v3/
 </pft></display>
 </loop>
 </do>
...
```

parm.name=posttag

Informa o número do campo do posting a ser acessaso no intervalo de chaves.

Exemplo

```
...
 <do task=keyrange>
 <parm name=db>CDS</parm>
 <parm name=from>B</parm>
 <parm name=count>20</parm>
 <parm name=posttag>70</parm>
 <field action=define tag=1001>Isis_Current</field>
 <field action=define tag=1>Isis_Key</field>
 <field action=define tag=2>Isis_Postings</field>
 <field action=define tag=3>Isis_Posting</field>
 <display><pft>
 ' POSTINGS',c15,'KEY',c46,'POSTING DETAIL'/#
 </pft></display>
 <loop>
 <display><pft>
 f(val(v1001),2,0),') ',v2,c15,v1,c46,v3/
 </pft></display>
 </loop>
 </do>
...
```

parm.name=prefix

Prefixo a ser inserido no esquema de destaque de texto, ou prefixo a ser usado pelo elemento <*htmlpft*>.

Exemplo

```
...
 <parm name=prefix>[pft]</prefix>
 <parm name=suffix>[/pft]</suffix>
 <htmlpft><pft>cat('TEST.HTM')</pft></htmlpft>
...
```

parm.name=reset

Permite que a atualização do arquivo invertido não altere a informação de registro do arquivo mestre com inversão pendente. Aplicável para bases de dados com múltiplos arquivo invertidos.

Exemplo

```
...
 <do task="fullinvertion">
 <parm name="db">CDS</parm>
 <parm name="fst">CDS.FST</parm>
 <parm name="reset">Off</parm>
 <field action="define" tag="1102">Isis_Status</field>
 <display><pft>'Full invertion: CDS'/</pft></display>
 <loop></loop>
 <display><pft>'Finished.'/</pft></display>
 <display><pft>'Lock status = 'v1102/</pft></display>
 </do>
...
```

parm.name=reverse

Informa que os registros resultado da tarefa especificada no elemento <*do*> serão acessados em ordem reversa.

Exemplo

```
...
 <do task=search>
 <parm name=db><pft>v2001</pft></parm>
 <parm name=expression><pft>v2005</pft></parm>
 <parm name=reverse>On</parm>
 <loop>
 <display><pft>mfn/</pft></display>
 </loop>
```

```
 </do>
...
```

parm.name=sort

Informa o formato de ordenação da lista interna do IsisScript.

Exemplo

```
...
 <list action=load type=list>
 <pft>'5.00'/,'1.50'/,'10.00'/,'8'/,'3.75'/,'14.20'/,'0.40'/
 </pft></list>
 <do task=list>
 <field action=define tag=1001>Isis_Current</field>
 <field action=define tag=1002>Isis_Itens</field>
 <field action=define tag=1>Isis_Item</field>
 <parm name=sort><pft>f(val(v1),10,2)</pft></parm>
 <loop>
 <display><pft>v1001,'/',v1002,c10,v1/</pft></display>
 </loop>
 </do>
...
```

parm.name=stw

Informa o arquivo com a tabela de palavras que não devem ser incluídas no arquivo invertido (Stop Word Table) que será usada para atualização do arquivo invertido ou na extração de chaves.

Exemplo

```
...
 <do task=fullinvertion>
 <parm name=db><pft>v2001</pft></parm>
 <parm name=fst><pft>v2061</pft></parm>
 <parm name=stw><pft>v2001,'.stw'</pft></parm>
 <field action=define tag=1102>Isis_Status</field>
 <display><pft>'Full invertion: ',v2001/</pft></display>
 <loop></loop>
 <display><pft>'Finished.'/</pft></display>
 <display><pft>'Lock status = 'v1102/</pft></display>
```

```
 </do>
...
```

parm.name=suffix

Sufixo a ser inserido no esquema de destaque de texto, ou sufixo a ser usado pelo elemento <*htmlpft*>.

Exemplo

```
...
 <parm name=prefix>[pft]</prefix>
 <parm name=suffix>[/pft]</suffix>
 <htmlpft><pft>cat('TEST.HTM')</pft></htmlpft>
...
```

parm.name=task

Indica o tipo de tarefa que será utilizada pelo elemento <*loop*>. Tem efeito se não for especificado o atributo *task* do elemento <*do*>.

Exemplo

```
...
 <do>
 <field action="cgi" tag="2081">dotask</field>
 <parm name="task"><pft>v2081</pft>
 <loop>
 ...
 </loop>
 </do>
...
```

parm.name=to

Informa qual é o último item a ser acessado pelo grupo do elemento <*loop*>.

Exemplo

```
...
<do task=keyrange>
 <parm name=db> <pft>v2001</pft></parm>
 <parm name=from><pft>v2002</pft></parm>
```

```
 <parm name=to> <pft>v2002,'ZZZZZZZZZ'</pft></parm>
 <loop>
 <display><pft>v1/</pft></display>
 </loop>
</do>
...
```

parm.name=type

Informa o tipo de arquivo para exportação ou importação.

Os tipos possíveis são: *ISO2709*, *HLine*, *RLine* e *VLine*.

ISO2709 é uma norma ISO (International Standards Organization) mas limita o número de identificação dos campos em 3 dígitos.

HLine é o mais eficiente, utiliza o comando H do elemento <*proc*>.

RLine é usado somente para importação, onde cada linha de um arquivo sequencial corresponde a um registro.

VLine é o recomendado para permitir modificação via editor de texto.

Exemplo

```
...
 <do task=import>
 <parm name=file><pft>v2041</pft></parm>
 <parm name=type><pft>v2042</pft></parm>
 <loop>
 <display><pft>ALL</pft></display>
 </loop>
 </do>
...
```

parm.name=uctab

Muda a tabela de conversão de caracteres para maiúscula do IsisScript durante a seção corrente.

Esta tabela informa, para a atualização do arquivo invertido, para a extração de chaves e para a opção mode da linguagem de formato, todos os caracteres correspondentes de minúsculo ou maiúsculo com acentuação para o correspondente maiúsculo sem acento.

Em uma seção sem o elemento <*parm name=uctab*> o IsisScript assume a tabela ANSI.

Exemplo

```
<IsisScript>
```

```
<section>
<parm cipar><pft>'CDS.*=/bases/cds/cds.*'/,
'ACTAB=/isis/menu/isisac.tab'/</pft>
'UCTAB=/isis/menu/isisuc.tab'/</pft>
</parm>
<parm name=actab><pft>cat('ACTAB')</pft></parm>
<parm name=uctab><pft>cat('UCTAB')</pft></parm>
<do task=search>
<parm name=db>CDS</parm>
<parm name=expression>plants*water</parm>
<parm name=to>10</parm>
<loop>
<display><pft>mpu,v24,mpl</pft></display>
</loop>
</do>
...
</section>
</IsisScript>
```

parm.tag

1. Pode ser usado em: <field> <parm>
2. Sintaxe: tag=...
   O atributo **tag** é usado para especificar o número do campo.

Exemplo

```
...
<parm name="fst" type="check" tag="1">
<pft>cat(v2065)</pft>
</parm>
...
```

parm.type

1. Opções: check
2. Pode ser usado em: <do>
3. Sintaxe: type=check
   Especifica o tipo da parâmetro.

Exemplo

```
...
 <parm name="fst" type="check" tag="1">
 <pft>cat(v2065)</pft>
 </parm>
...
```

parm.type=check

Permite a verificação da sintaxe de uma FST (Field Select Table). Atualiza o campo especificado pelo atributo tag com o código de erro (5 dígitos), um espaço e o ponto em que foi detectado o erro de sintaxe, ou 00000 caso não haja erro de sintaxe.

Exemplo

```
...
 <field action="cgi" tag="2065">fst</field>
 <parm name="fst" type="check" tag="1"><pft>cat(v2065)</pft></parm>
 <display><pft>ALL</pft></display>
...
```

<pft>

1. Pode conter: <pft>
2. Pode ser usado em: <call> <cgitable> <define> <display> <export> <extract> <field> <file> <flow> <htmlpft> <label> <list> <parm> <pft> <proc> <return> <trace> <write>
3. Atributos: type
4. Sintaxe: <pft> ... </pft>
   Formata o registro corrente.

Exemplo

```
...
 <display><pft>("Autor: "v70+|; |)</pft></display>
 <display><pft>@CDS.PFT</pft></display>
 <display><pft>cat('C:\AUTOEXEC.BAT')</pft></display>
 <display><pft>ref(['CONFIG']1,v500/)</pft></display>
...
```

pft.type

1. Opções: check reload
2. Pode ser usado em: <field> <file> <htmlpft> <list> <pft>
3. Sintaxe: type=...

   Tipo de ação a ser tomada para execução do formato.

Exemplo

```
...
 <do>
 <parm name=to>10</parm>
 <loop>
 <display>
 <pft type=reload>
 <pft>ref(['CONFIG']val(v1),v500/)</pft>
 </pft>
 </display>
 </loop>
 </do>
...
```

pft.type=check

Permite a verificação da sintaxe de um formato. Retorna o código de erro (5 dígitos), um espaço e o ponto em que foi detectado o erro de sintaxe, ou caso não haja erro de sintaxe.

Exemplo

```
...
 <field action="cgi" tag="2065">pft</field>
 <display>
 <pft type="check">
 <pft>v2065</pft>
 </pft>
 </display>
...
```

pft.type=reload

Use esta opção para informar ao IsisScript que o formato terá que ser recompilado cada vez que esta instrução for executada.

Exemplo

```
...
 <display>
 <pft type=reload>
 <pft>ref(['CONFIG']1,v500/)</pft>
 </pft>
 </display>
...
```

<proc>

1. Pode conter: <pft>
2. Pode ser usado em: <do> <function> <hl> <loop> <section> <update>
3. Sintaxe: <proc> ... </proc>
   Modifica o conteúdo do registro corrente.

Exemplo

```
...
 <proc><pft>'a',v2024,'~',v2027,'~'</pft></proc>
...
```

<return>

1. Pode conter: <pft>
2. Pode ser usado em: <function>
3. Atributos: action split tag
4. Sintaxe: <return> ... </return>
   Sai da funcão corrente.

Exemplo

```
...
<function name=ParamTest action=replace tag=1 split=occ>
 <display><pft>##'ParamTest'/</pft></display>
```

```
 <display><pft>ALL</pft></display>
 <return action=replace tag=9999 split=occ>
 <pft>(v1/)</pft>
 </return>
 <display>Parameter field 1 absent!</display>
</function>
...
```

return.action

Veja: <field action=...>

Retorno de parâmetros para quem chamou a função. Tem a mesma funcionalidade do atributo *action* do elemento <*field*>.

Exemplo

```
...
<function name=ParamTest action=replace tag=1 split=occ>
 <display><pft>##'ParamTest'/</pft></display>
 <display><pft>ALL</pft></display>
 <return action=replace tag=9999 split=occ>
 <pft>(v1/)</pft>
 </return>
 <display>Parameter field 1 absent!</display>
</function>
...
```

return.split

Veja: <field split=...>

Retorno de parâmetros para quem chamou a função. Tem a mesma funcionalidade do atributo **split** da instrução <*field*>.

Exemplo

```
...
<function name=ParamTest action=replace tag=1 split=occ>
 <display><pft>##'ParamTest'/</pft></display>
 <display><pft>ALL</pft></display>
 <return action=replace tag=9999 split=occ>
 <pft>(v1/)</pft>
```

```
 </return>
 <display>Parameter field 1 absent!</display>
</function>
...
```

return.tag

Veja: <field tag=...>

Retorno de parâmetros para quem chamou a função. Tem a mesma funcionalidade do atributo *tag* do elemento <*field*>.

Exemplo

```
...
<function name=ParamTest action=replace tag=1 split=occ>
 <display><pft>##'ParamTest'/</pft></display>
 <display><pft>ALL</pft></display>
 <return action=replace tag=9999 split=occ>
 <pft>(v1/)</pft>
 </return>
 <display>Parameter field 1 absent!</display>
</function>
...
```

return.tag

Veja: <field tag=...>

Retorno de parâmetros para quem chamou a função. Tem a mesma funcionalidade do atributo *tag* da instrução <*field*>.

Exemplo

```
...
<function name=ParamTest action=replace tag=1 split=occ>
 <display><pft>##'ParamTest'/</pft></display>
 <display><pft>ALL</pft></display>
 <return action=replace tag=9999 split=occ>
 <pft>(v1/)</pft>
 </return>
 <display>Parameter field 1 absent!</display>
</function>
```

```
...
```

<section>

1. Pode conter: <call> <cgitable> <define> <display> <do> <export> <extract> <field> <file> <flow> <hl> <label> <list> <parm> <proc> <trace>
2. Pode ser usado em: <IsisScript>
3. Atributos: name
4. Sintaxe: <section> ... </section>

   O elemento **<section>** é usado para iniciar uma seqüência de instruções que acessam campos comuns e utilizam-se de tabelas comuns.

   O atributo *name* pode ser utilizado para identificação.

Exemplo

```
<IsisScript name=Test>
 <section name=TestFirst>
 <display><pft>mpu,'Test'</pft></display>
 </section>
</IsisScript>
```

section.name

1. Pode ser usado em: <call> <function> <IsisScript> <section>
2. Sintaxe: name=...

   O atributo **name** é opcional, quando usado serve para identificação da seção.

Exemplo

```
<IsisScript name=Test>
 <section name=TestFirst>
 <display><pft>mpu,'Test'</pft></display>
 </section>
</IsisScript>
```

<trace>

1. Pode conter: <pft>
2. Pode ser usado em: <do> <function> <hl> <IsisScript> <loop> <section> <update>

3. Sintaxe: <trace> ... </trace>
   Ativa ou destiva a mostra da instrução que está sendo executada. Os modos possíveis são normal, linha a linha ou tabela, respectivamente: *On*, *BR* e *Table*.

Exemplo

```
...
 <trace>On</trace>
...
```

<update>

1. Pode conter: <call> <cgitable> <define> <display> <do> <export> <extract> <field> <file> <flow> <hl> <label> <list> <parm> <proc> <return> <trace> <write>
2. Pode ser usado em: <do>
3. Sintaxe: <update> ... </update>
   Inicia um bloco de instruções para modificação ou adição de um registro.

Exemplo

```
...
 <do task=update>
 <parm name=db>CDS</parm>
 <parm name=mfn>New</parm>
 <field action=define tag=1102>Isis_Status</field>
 <update>
 <field action=append tag=1>One more</field>
 <write>Unlock</write>
 <display><pft>ALL</pft></display>
 </update>
 </do>
...
```

<write>

1. Pode conter: <pft>
2. Pode ser usado em: <update>
3. Sintaxe: <write> ... </write>
   Elemento que grava a modificação do registro.

Se o elemento <*parm name=mfn*> indicar *New* ou *GetNew* então adiciona um novo registro, senão atualiza o mfn passado como argumento.

Se o argumento do elemento **<write>** for *Unlock* o registro será desbloqueado após ser gravado, se for *Lock* o registro será gravado e bloqueado, se for *NoUnlock* o registro permanecerá bloqueado e a informação de bloqueio permanecerá a mesma, se for *Delete* o registro será deletado.

Exemplo

```
...
 <do task=update>
 <parm name=db>CDS</parm>
 <parm name=mfn>New</parm>
 <field action=define tag=1102>Isis_Status</field>
 <update>
 <field action=add tag=1>Mais um</field>
 <write>Unlock</write>
 <display><pft>ALL</pft></display>
 </update>
 </do>
...
```

Referências bibliográficas

1. PACKER, Abel Laerte et al. WWWISIS : el camino hacia Internet. INFOISIS, Buenos Aires, v. 2, n. 3-4, p. 7-21, 1996.

2. SANTOS, Gildenir Carolino, PIETROSANTO, Ademir Giacomo. O acesso em base de dados em economia e educação, pela Internet através da ferramenta WWWISIS. Presented in Seminário Nacional de Bibliotecas Universitárias, 10., 1998, Fortaleza. (electronic version: diskette).

3. JAYAKANTH, F. Implementing WWWISIS for providing Web access to bibliographic databases. INSPEL, [Germany], v. 35, n.1, p. 42-54, 2001.

Glossário

- **Arquivo**. Em computação, um conjunto de dados que pode ser gravado em algum dispositivo de armazenamento. Os arquivos de dados são criados por aplicativos, como por exemplo um processador de textos.

- **Arquivo invertido**. Conjunto de seis arquivo físicos, cinco dos quais contém os termos de busca do dicionário (organizados como uma árvore B*) e o sexto contém a lista de apontadores associadas a cada termo. A fim de otimizar o armazenamento em disco, se mantém duas árvores B* em separado: uma para os termos com tamanho até 10 caracteres (armazenados nos arquivos .N01 e .L01) e outra para os termos com tamanho até 30 caracteres (armazenados nos arquivos .N02 e .L02). O arquivo .CNT contém os campos de controle para ambas árvores B*). Em cada arquivo da árvore B* o archivo .N0x contém os nós da árvore e o arquivo .L0x contém as folhas. Os registros das folhas apontam para o lugar onde se encontram os apontadores que contém a informação para localizar os registros (postings) na base de dados. Este arquivo se identifica 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), produzindo uma cópia de segurança que pode ser restaurada em caso de apagamento acidental ou dano físico dos dados originais.

- **Base de dados**. Coleção de dados estruturados para serem acessados e manipulados facilmente. É formada por unidades chamadas registros, cujos diversos atributos são representados por campos e subcampos. Por exemplo, num arquivo "cadastro de clientes", cada cliente representa um registro, que possui vários campos, como "NOME", "CÓDIGO DO CLIENTE", "TELEFONE" etc.

- **Bases de dados bibliográficas**. 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.

- **CGI**. É um padrão para conectar aplicações externas com os provedores de acesso a informação, tais como o HTTP ou os Web Services.

- **Chave**. Expressão que identifica uma ou mais informações de determinada classe ou tipo e que pode ser usada em uma 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 armazenagem, recuperação e apresentação de informação passível de transmissão online ou gravação em mídia magnética ou óptica.

- **Formato ISO (de arquivo)**. Padrão estabelecido pela ISO para intercâmbio de dados entre instituições, redes e usuários.

- **Formato LILACS**. Formato de descrição bibliográfica estabelecido pela BIREME, baseado na 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 a esse documento com o objetivo de recuperá-lo posteriormente.

- **Posting**. Consiste de um endereço de chave extraída do arquivo master.

- **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 outros elementos relacionados.

- **URL**. Padrão definido para endereçamento de conteúdos de dados via protocolo TCP/IP. Os navegadores de internet utilizam a URL para acessar páginas na web.