

BIREME / OPS / OMS

Centro Latinoamericano y del Caribe de Información en Ciencias de la Salud

Referencia del lenguaje IsisScript

Versión 1.1

São Paulo - 2006

Copyright © 2006 - BIREME / OPS / OMS

Referencia del lenguaje IsisScript

Se concede permiso para copiar, distribuir y/o modificar este documento bajo los términos de la Licencia de Documentación Libre de GNU, Versión 1.2 o cualquier otra versión posterior publicada por la Free Software Foundation; sin Secciones Invariantes ni Textos de Cubierta Delantera ni Textos de Cubierta Trasera. Una copia de la licencia está incluida en la sección titulada GNU Free Documentation License.

Ficha Catalográfica

BIREME / OPS / OMS (Brasil)

Referencia del lenguaje IsisScript. / BIREME (org.). São Paulo : BIREME / OPS / OMS, 2006.

76 p.

1. Manual del usuario. 2. Acceso a la información. 3. Sistemas de información. 4. Gerenciamento de información. 5. Salud Pública. 6. Servicios de salud . I. BIREME II. Título

Advertencia - La mención a las compañías y/o instituciones específicas o a ciertos productos no implica que estos sean apoyados o recomendados por BIREME / OPS / OMS, y no significa que haya preferencia en relación a otros de naturaleza similar, citados o no.

BIREME / OPS / OMS

Centro Latinoamericano y del Caribe de Información en Ciencias de la Salud

Rua Botucatu, 862 - V. Clementino

Este documento fue producido con la Metodología para la Normalización de Documentos (NorDoc) desarrollada por BIREME.

Tabla de contenido

Abreviaturas utilizadas	V
Como usar este manual	VII
Prefacio	1
Sobre BIREME	1
La Biblioteca Virtual en Salud (BVS)	2
Referencia del IsisScript	5
<IsisScript>	5
<i>IsisScript.name</i>	5
<call>	6
<i>Call.name</i>	6
<cgitable>	7
<define>	8
<display>	8
<do>	9
<i>Do.task</i>	9
Do.task=fullinversion	10
Do.task=import	10
Do.task=invertedload	11
Do.task=keyrange	11
do.task=list	12
do.task=mastersort	13
do.task=mfnrange	14
do.task=search	14
do.task=update	15
<export>	15
<extract>	16
<field>	16
<i>field.action</i>	17
field.action=add	17
field.action=cgi	18

field.action=define	18
field.action=delete	18
field.action=export	19
field.action=hl	19
field.action=import	19
field.action=occ	20
field.action=replace	20
field.action=statusdb	20
field.action=statusfile	21
<i>field.from</i>	21
<i>field.previous</i>	22
field.previous=add	22
field.previous=delete	22
<i>field.split</i>	22
field.split=flddir	23
field.split=occ	23
<i>field.tag</i>	23
field.tag=list	24
<i>field.type</i>	24
field.type=flag	24
<file>	25
<i>file.action</i>	25
file.action=append	25
file.action=close	26
file.action=create	26
file.action=delete	26
file.action=unlock	26
<i>file.type</i>	27
file.type=database	27
file.type=file	27
file.type=inverted	27
file.type=master	28
file.type=output	28
file.type=tempfile	28
<flow>	28
<i>flow.action</i>	29
flow.action=exit	29
flow.action=jump	30
flow.action=skip	30
<function>	30
<i>function.action</i>	31
<i>function.from</i>	31
<i>function.name</i>	32
<i>function.split</i>	32
<i>function.tag</i>	32
<hl>	33
<htmlpft>	33
<i>htmlpft.action</i>	33
htmlpft.action=convert	34
<i>htmlpft.type</i>	34
<label>	35

<list>	35
<i>list.action</i>	35
list.action=delete	36
list.action=load	36
<i>list.type</i>	36
list.type=freq	36
list.type=list	37
list.type=sort	37
<loop>	37
<parm>	38
<i>parm.name</i>	38
parm.name=actab	39
parm.name=bufferize	40
parm.name=cipar	40
parm.name=count	40
parm.name=db	41
parm.name=decod	41
parm.name=delimiter	42
parm.name=expire	42
parm.name=expression	42
parm.name=file	43
parm.name=freqsum	43
parm.name=from	43
parm.name=fst	44
parm.name=gizmo	44
parm.name=indexlist	45
parm.name=key	45
parm.name=keyfield	46
parm.name=keylength	46
parm.name=keys	47
parm.name=keyfdb	48
parm.name=lockid	48
parm.name=maxlk	49
parm.name=mfn	49
parm.name=posting	50
parm.name=posttag	50
parm.name=prefix	51
parm.name=reset	51
parm.name=reverse	52
parm.name=sort	52
parm.name=stw	53
parm.name=suffix	53
parm.name=task	53
parm.name=to	54
parm.name=type	54
parm.name=uctab	55
<i>parm.tag</i>	55
<i>parm.type</i>	56
parm.type=check	56
<pft>	56
<i>pft.type</i>	57

pft.type=check	57
pft.type=reload	58
<proc>	58
<return>	58
<i>return.action</i>	59
<i>return.split</i>	59
<i>return.tag</i>	60
<i>return.tag</i>	60
<section>	60
<i>section.name</i>	61
<trace>	61
<update>	62
<write>	62
Citas bibliográficas	64
Glosario	65

Abreviaturas utilizadas

- **ANSI.** American National Standards Institute [Instituto Nacional Americano de Normas].
- **ASCII.** American Standard Code for Information Interchange [Código Americano Normalizado para el Intercambio de Información].
- **BIREME.** Centro Latinoamericano y del Caribe de Información en Ciencias de la Salud.
- **BVS.** Biblioteca Virtual en Salud.
- **CGI.** Common Gateway Interface [Interfaz Común de Pasarela].
- **FST.** Field Selection Table [Tabla de Selección de Campo].
- **HTML.** HyperText Markup Language [Lenguaje de Marcación de Hipertexto].
- **HTTP.** HyperText Transfer Protocol [Protocolo de Transferencia de Hipertexto].

- ISO. International Organization for Standardization [Organización Internacional para la Normalización].
- MFN. Master file number [Número maestro de archivo].
- OMS. Organización Mundial de la Salud.
- OPS. Organización Panamericana de la Salud.
- STW. STop Word file [Archivo de palabras prohibidas].
- UNESCO. United Nations Educational, Scientific and Cultural Organization [Organización de las Naciones Unidas para la Educación, la Ciencia y la Cultura].
- URL. Universal Resource Locator [Localizador Universal de Recurso].

Como usar este manual

Este manual está organizado en un único capítulo conteniendo todos los comandos, sus parámetros y las opciones disponibles en el lenguaje IsisScript.

Los comandos están organizados en el nivel 2 de los títulos (Headings) y sus elementos siguen la jerarquía hasta el nivel 4.

Cada comando, parámetro o opción contiene una breve descripción de su propósito. Información adicional como atributos, sintax, elementos internos y donde es aplicable también están disponibles y son completadas con un código de ejemplo para demostrar el comando, la opción o el parámetro.

El lenguaje de formato de salida del modelo CDS/ISIS y la estructura de bases de datos son los únicos conocimientos requeridos del usuario.

Prefacio

Sobre BIREME

Año tras año, BIREME cumple su misión como centro especializado en información científica y técnica en salud para la región de América Latina y el Caribe. Establecida en Brasil en 1967, con el nombre de Biblioteca Regional de Medicina (que originó la sigla BIREME), atendió desde el inicio a la creciente demanda de literatura científica actualizada por parte de los sistemas nacionales de salud y las comunidades de investigadores, profesionales y estudiantes. Posteriormente, en 1982, pasó a llamarse Centro Latinoamericano y del Caribe de Información en Ciencias de la Salud, para mejor expresar sus funciones, orientadas al fortalecimiento y ampliación del flujo de información científica y técnica en salud en toda la región, pero conservó su sigla.

El trabajo en red, en base a la descentralización, orientado a desarrollar capacidades locales, compartir recursos de información, desarrollar productos y servicios cooperativos, elaborar metodologías comunes, siempre fue el fundamento del trabajo de cooperación técnica de BIREME. De esa forma el centro se consolida como un modelo internacional que promueve la capacitación de los profesionales de información a nivel gerencial y técnico, para que adopten los paradigmas de información y comunicación que mejor atiendan a las necesidades locales.

Los principales fundamentos que dan origen y soporte a la existencia de BIREME son los siguientes:

- el acceso a la información científico-técnica en salud es esencial al desarrollo de la salud;
- la necesidad de desarrollar la capacidad de los países de América Latina y el Caribe de operar las fuentes de información científico-técnica en salud de forma cooperativa y eficiente;
- la necesidad de promover el uso y de responder a las demandas de información científico-técnica en salud de los gobiernos, los sistemas de salud, las instituciones de enseñanza e investigación.

BIREME, como centro especializado de la Organización Panamericana de la Salud (OPAS)/Organización Mundial de la Salud (OMS), coordina y realiza actividades de cooperación técnica en gestión de información y conocimiento científico, con el propósito de fortalecer y ampliar el flujo de información científica en salud en Brasil y en los demás países de América Latina y el Caribe, como condición esencial para el desarrollo de la salud, incluyendo planificación, gestión, promoción, investigación, educación y atención.

El convenio que fundamenta BIREME es renovado a cada cinco años por los miembros del Comité Asesor Nacional de la institución (OPAS, Ministerio de la Salud de Brasil, Ministerio de Educación y Cultura de Brasil, Secretaría de Salud del Estado de São Paulo y Universidad Federal de São Paulo – Unifesp). Esta última ofrece la infraestructura física necesaria al establecimiento de la institución.

En 2004 la institución asumió la responsabilidad de convertirse en una entidad que se basa en el conocimiento.

La Biblioteca Virtual en Salud (BVS)

Con el surgimiento y consolidación de la Internet como medio predominante de información y comunicación, el modelo de cooperación técnica de BIREME evolucionó desde 1998 hacia la construcción y desarrollo de la Biblioteca Virtual en Salud (BVS) como espacio común de convergencia del trabajo cooperativo de productores, intermediarios y usuarios de información. La BVS promueve el desarrollo de una red de fuentes de información científica y técnica con acceso

universal en la Internet. Por primera vez se abre la posibilidad real de acceso equitativo a la información en salud.

BIREME tiene a la Biblioteca Virtual como modelo para la gestión de información y conocimiento, lo que implica la cooperación y convergencia de instituciones, sistemas, redes e iniciativas de productores, intermediarios y usuarios en la operación de redes de fuentes de información locales, nacionales, regionales e internacionales, privilegiando así el acceso abierto y universal.

Actualmente, todos los países de América Latina y el Caribe (Región) participan directa o indirectamente en los productos y servicios cooperativos promovidos por la BVS, lo que involucra a más de mil instituciones en más de 30 países.

La BVS es simulada en un espacio virtual de la Internet formada por la colección o red de fuentes de información en salud de la Región. Usuarios de distintos niveles y localización pueden interactuar y navegar en el espacio de una o varias fuentes de información, independientemente de su localización física. Las fuentes de información son generadas, actualizadas, almacenadas y operadas en la Internet por productores, integradores e intermediarios, de modo descentralizado, obedeciendo a metodologías comunes para su integración a la BVS.

La BVS organiza la información en una estructura que integra e interconecta bases de datos referenciales, directorios de especialistas, eventos e instituciones, catálogo de recursos de información disponibles en la Internet, colecciones de textos completos con destaque para la colección SciELO (Scientific Electronic Online) de revistas científicas, servicios de disseminación selectiva de información, fuentes de información de apoyo a la educación y la toma de decisión, noticias, listas de discusión y apoyo a comunidades virtuales. Por lo tanto, el espacio de la BVS constituye una red dinámica de fuentes de información descentralizada a partir de la cual se puede recuperar y extraer información y conocimiento para subsidiar los procesos de decisión en el área de la salud.

La Biblioteca Virtual en Salud es visualizada como la base distribuida del conocimiento científico y técnico en salud registrado, organizado y almacenado en formato electrónico en los países de la Región, accesible de forma universal en la Internet de modo compatible con las bases internacionales.

Referencia del IsisScript

<IsisScript>

1. Puede Contener : <function> <section> <trace>
2. Atributos : name
3. Sintaxis : <IsisScript> ... </IsisScript>

El elemento **<IsisScript>** es usado para indicar un grupo de instrucciones IsisScript.

El atributo *name* es usado para nombrar y documentar el grupo.

Ejemplo

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

IsisScript.name

1. Puede ser usado en: <call> <function> <IsisScript> <section>
2. Sintaxis: name=...

El atributo **name** es opcional, cuando es usado sirve sólo para hacer la documentación del IsisScript.

Ejemplo

```
<IsisScript name>HelloWorld>
```

```

<section>
<display>Hello World!</display>
</section>
</IsisScript>

```

<call>

1. Puede Contener : <pft>
2. Puede ser usado en: <do> <function> <hl> <loop> <section> <update>
3. Atributos : name
4. Sintaxis : <call> ... </call>

El elemento **<call>** indica la llamada de una función. La función a ser llamada es especificada por el atributo *name*.

El argumento del elemento **<call>** es pasado como parámetro para la función.

Ejemplo

```

<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. Puede ser usado en: <call> <function> <IsisScript> <parm> <section>
2. Sintaxis: name=...

Nombre de la función a ser llamada.**Ejemplo**

```

<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. Puede Contener: <pft>
2. Puede ser usado en: <do> <function> <hl> <loop> <section> <update>
3. Sintaxis: <cgitable> ... </cgitable>

Para cada línea del argumento especificado, adiciona al campo especificado el contenido del CGI ("value") correspondiente al "name" especificado. Cada línea del argumento equivale a la definición <field action="cgi" tag="...">name</field>

Ejemplo

```

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

```

<define>

1. Puede Contener: <pft>
2. Puede ser usado en: <do> <function> <hl> <loop> <section> <update>
3. Sintaxis: <define> ... </define>

Define campos que serán usados dentro del elemento <loop>. Cada línea equivale a la definición <field action="define" tag="...">Isis_...</field>

Ejemplo

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

<display>

1. Puede Contener: <htmlpft> <pft>
2. Puede ser usado en: <do> <function> <hl> <loop> <section> <update>
3. Sintaxis: <display> ... </display>

Envía un texto a la salida corriente. El IsisScript empieza teniendo como salida corriente el "standard output" del sistema operativo.

El redireccionamiento de la salida es informado por medio del elemento <file>.

Ejemplo

```
<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. Puede Contener: <call> <cgitable> <define> <display> <do> <export> <extract> <field> <file> <flow> <hl> <label> <list> <loop> <parm> <proc> <return> <trace> <update>
2. Puede ser usado en: <do> <function> <hl> <loop> <section> <update>
3. Atributos: task
4. Sintaxis: <do> ... </do>

El elemento **<do>** informa el inicio de una tarea IsisScript. Las tareas posibles son: intervalo de registros, intervalo de claves, investigación, repetición, lista, importación de registros, ordenación de bases de datos, carga de archivo invertido y generación completa de archivo invertido. La tarea de lista puede ser de frecuencia, ordenación o simple.

El atributo *task* es usado para informar el tipo de tarea a ser ejecutada.

Los parámetros necesarios para la ejecución de la tarea son informados por medio del elemento *<parm>*.

El elemento *<loop>* informa el bloque de instrucciones a ser ejecutadas para cada ítem de la tarea.

Ejemplo

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

Do.task

1. Opciones: fullinversion import invertedload keyrange list mastersort mfnrange search update
2. Puede ser usado en: <do>
3. Sintaxis: task=...

Informa la tarea generadora de registros que serán utilizados por el elemento *<loop>*.

La ausencia del atributo *task* indica una repetición independiente de tarea.

Ejemplo

```
...
<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=fullinversion

Hace la generación completa del archivo invertido.

La base de datos es informada por el elemento *<parm name=db>*.

La FST (Field Select Table) es informada por el elemento *<parm name=fst>*.

Ejemplo

```

...
<do task=fullinversion>
<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 inversion:
',v2001/</pft></display>
<loop></loop>
<display><pft>'Finished. ' /</pft></display>
<display><pft>'Lock status =
'v1102/</pft></display>
</do>
...

```

Do.task=import

Accede a datos de un archivo texto y carga en la forma de registros de base de datos.

El nombre del archivo es informado por medio del elemento *<parm name=file>*.

El elemento *<parm name=type>* puede ser usado para informar el tipo del archivo texto; archivo texto *ISO 2709* es asumido si el tipo no es informado.

Ejemplo

```

...
<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

Hace la carga del archivo invertido por medio de una base de datos que contiene las claves ordenadas.

La base de datos a ser cargada es informada por el elemento `<parm name=db>`.

La base de datos que contiene las claves ordenadas es informada por el elemento `<parm name=keyddb>`.

Ejemplo

```

...
<do task=invertedload>
<parm name=db>TEST</parm>
<parm name=keyddb>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

Recorre una lista de claves del archivo invertido de una base de datos.

La base de datos es informada por el elemento `<parm name=db>`.

La clave inicial es informada por el elemento `<parm name=from>`.

La clave final es informada por el elemento `<parm name=estoy>`.

La cantidad de claves puede ser limitada por el elemento `<parm name=count>`.

El elemento `<parm name=reverse>` puede ser usado para recorrer las claves en orden inversa.

El elemento `<parm name=posting>` puede ser usado para recorrer también la lista de posting de cada clave.

El elemento `<parm name=posttag>` puede ser usado para recorrer la lista de postings de cada clave solamente para el tag especificado.

Ejemplo

```

...
<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

Recorre la lista de ítems previamente cargados por medio del elemento `<list action=load type=...>`.

La lista puede ser ordenada por medio del elemento `<parm name=sort>`.

El ítem inicial es informado por el elemento `<parm name=from>`.

El ítem final es informado por el elemento `<parm name=estoy>`.

La cantidad de registros puede ser limitada por el elemento `<parm name=count>`.

El elemento `<parm name=reverse>` puede ser usado para recorrer la lista en orden inversa.

Ejemplo

```

...
<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 los registros de una base de datos.

La base de datos es informada por el elemento *<parm name=db>*.

La clave para ordenación es informada por el elemento *<parm name=key>*.

El tamaño de la clave es informado por el elemento *<parm name=keylength>*.

Ejemplo

```

...
<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

Recorre una lista de registros de una base de datos.

La base de datos es informada por el elemento *<parm name=db>*.

El MFN inicial es informado por el elemento *<parm name=from>*.

El MFN final es informado por el elemento *<parm name=estoy>*.

La cantidad de MFNs puede ser limitada por el elemento *<parm name=count>*.

El elemento *<parm name=reverse>* puede ser usado para recorrer los MFNs en orden inversa.

Ejemplo

```
...
<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

Investiga una base de datos y recorre la lista de registros encontrados.

La base de datos es informada por el elemento *<parm name=db>*.

La expresión de investigación es informada por el elemento *<parm name=expresionismo>*.

El registro inicial es informado por el elemento *<parm name=from>*.

El registro final es informado por el elemento *<parm name=estoy>*.

La cantidad de registros puede ser limitada por el elemento *<parm name=count>*.

El elemento *<parm name=reverse>* puede ser usado para recorrer los registros en orden inversa.

Ejemplo

```
...
<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

Actualiza o añade un registro en la base de datos.

La base de datos es informada por medio del elemento `<parm name=db>`.

El elemento `<parm name=mfn>` es usado para informar el MFN a ser actualizado o si hay que añadir un nuevo registro.

El elemento `<parm name=lockid>` es usado para identificar al "dueño" del registro.

El elemento `<parm name=espiré>` puede ser usado para informar que el "dueño" del registro tuvo su tiempo, para permanecer con el registro bloqueado, expirado.

Ejemplo

```

...
<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. Puede Contener: `<pft>`
2. Puede ser usado en: `<do>` `<function>` `<hl>` `<loop>` `<section>` `<update>`
3. Sintaxis: `<export>` ... `</export>`

Añade el registro corriente a un archivo de texto.

El nombre del archivo debe haber sido previamente informado por medio del elemento `<parm file=...>`.

El tipo del archivo estándar es el ISO2709; para generar otro tipo de archivo exportación infórmelo previamente con el elemento `<parm type=...>`. Los tipos de archivo de exportación son: *ISO2709*, *HLine* y *VLine*.

Ejemplo

...

```

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

```

<extract>

1. Puede Contener: <pft>
2. Puede ser usado en: <do> <function> <hl> <loop> <section> <update>
3. Sintaxis: <extract> ... </extract>

Añade las claves extraídas del registro corriente en la base de datos de claves.

La FST (Field Select Table) debe ser previamente informada por medio del elemento <parm name=fst>.

La base de datos que contendrá las claves debe ser previamente informada por medio del elemento <parm name=keyfdb>.

El campo que contendrá las claves debe ser previamente informado por medio del elemento <field action=define tag=...>Isis_Key</field>.

El campo que contendrá los datos sobre el posting debe ser previamente informado por medio del elemento <field action=define tag=...>Isis_Posting</field>.

Ejemplo

```

...
<do task=mfnrange>
  <parm name=fst>1 0 v1</parm>
  <parm name=keyfdb>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. Puede Contener: <pft>

2. Puede ser usado en: <do> <function> <hl> <loop> <section> <update>
3. Atributos: action from previous split tag type
4. Sintaxis: <field> ... </field>

El elemento **<field>** es usado para adicionar, modificar, borrar, importar, exportar y definir campos.

El atributo *action* informa el uso del elemento.

El atributo *tag* informa el campo a ser afectado.

El atributo *split* puede ser usado para informar que cada línea será una nueva incidencia del campo.

El atributo *previous* informa si el contenido anterior será eliminado o no.

El atributo *type* informa el tipo de campo a ser accedido.

El atributo *from* informa el número del campo a ser accedido.

Ejemplo

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

field.action

1. Opciones: add cgi define delete export hl import occ replace statusdb statusfile
2. Puede ser usado en: <field> <file> <flow> <function> <htmlpft> <list> <return>
3. Sintaxis: action=...

Indica la acción que el elemento *<field>* va a ejecutar.

Ejemplo

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

field.action=add

Añade una nueva incidencia al campo especificado por el atributo **tag**.

El argumento del elemento contiene el dato a ser adicionado.

Ejemplo

```
...
<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 el contenido del campo especificado por el atributo **tag** con el valor correspondiente del CGI.

El argumento del elemento *<field>* indica cual es el ítem del CGI cuyo valor será adicionado.

Ejemplo

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

field.action=define

Define el número del campo especificado por el atributo **tag** como un número de actualización automática del IsisScript.

El argumento del elemento *<field>* indica qué tipo de información será almacenada en el campo.

Los siguientes argumentos pueden ser definidos:

Isis_Current - Índice de la ejecución corriente del *<loop>*

Isis_Total - Total de veces posible para el *<loop>*

Isis_From - Parámetro *from* del *<loop>*

Isis_To - Parámetro *to* del *<loop>*

Isis_Lock - Campo de almacenamiento del control de bloqueo de registros

Isis_Status - Almacenamiento del estado de la ejecución de la tarea

Isis_Item - Ítem de la lista

Isis_Value - Cantidad de ítems calculados por el conteo de frecuencia

Isis_Key - Clave corriente

Isis_Posting - Datos sobre el posting corriente

Isis_Postings - Total de postings de la clave corriente

Isis_Items - Total de ítems en la lista

Isis_ErrorInfo - Apuntador de error en la investigación

Isis_Keys - Claves para destacar texto

Isis_MFN - Número del campo de almacenamiento del MFN para exportación o importación de registros

Isis_RecordStatus - Número del campo que contendrá el estado del registro

Ejemplo

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

field.action=delete

Elimina una o todas las incidencias del campo especificado por el atributo **tag**.

El argumento del elemento *<field>* indica la incidencia a ser eliminada, use ALL en el argumento para indicar todas las incidencias.

Ejemplo

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

field.action=export

Modifica el contenido de uno o más campos en el objetivo anterior del IsisScript.

El campo es especificado por el atributo **tag**.

Use *tag=list* para indicar que el argumento contiene la lista de campos a ser exportada.

Ejemplo

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

field.action=hl

Sustituye el contenido del campo especificado por el atributo **tag** con un nuevo valor utilizando el esquema de destacar texto.

Ejemplo

```
...
<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 el contenido de uno o más campos en el objetivo corriente del IsisScript copiando los campos del objetivo anterior.

El campo es especificado por el atributo **tag**.

Use *tag=list* para indicar que el argumento contiene la lista de campos a ser importada.

Ejemplo

```

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

```

field.action=occ

Modifica el contenido del campo especificado por el atributo *tag* con el contenido del campo especificado por el atributo *from*, solamente con la incidencia especificada por el argumento del elemento.

Ejemplo

```

...
<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

Sustituye el contenido del campo especificado por el atributo **tag**. El argumento contiene el nuevo contenido.

Ejemplo

```

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

```

field.action=statusdb

Sustituye el contenido del campo especificado por el atributo **tag** con el estado de la base de datos informada por el argumento del elemento *<field>*.

Si existe el master o el archivo invertido o ambos, el campo será creado conteniendo el subcampo **s** (^s). El subcampo **s** contendrá el carácter *m*, si el master existe, y el carácter *i*, si el invertido existe.

Si el master existe, el campo contendrá también el subcampo **n** (^n) con el total de registros en la base de datos más uno, el subcampo **d** (^d) con el

número de bloqueo de entrada de datos (Data Entry Lock), y el subcampo **e** (^e) con el número de bloqueo exclusivo (Exclusive Write Lock).

Ejemplo

```
...
<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

Sustituye el contenido del campo especificado por el atributo **tag** con el estado del archivo informado por el argumento del elemento *<field>*.

Después de la ejecución, el campo contendrá el subcampo **s** (^s) con el carácter “e” si el archivo existe, en caso contrario el campo quedará ausente.

Ejemplo

```
...
<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. Puede ser usado en: *<field>*
2. Sintaxis: *from=...*

Especifica el número del campo que será accedido por el atributo *action=occ*.

Ejemplo

```
...
<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. Opciones: add delete
2. Puede ser usado en: <field>
3. Sintaxis: previous=...

Especifica si el campo siendo importado o exportado tendrá su contenido anterior eliminado o si nuevas incidencias serán adicionadas.

La ausencia de este atributo significa que el contenido anterior será eliminado.

Ejemplo

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

field.previous=add

Adiciona nuevas incidencias.

Ejemplo

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

field.previous=delete

Informa que es para eliminar las incidencias anteriores.

Ejemplo

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

field.split

1. Opciones: flddir occ
2. Puede ser usado en: <field>
3. Sintaxis: split=...

Indica la forma en que el dato será almacenado.

Ejemplo

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

field.split=flddir

El texto a ser almacenado en el campo especificado por el atributo **tag** es el directorio de campos del registro con los respectivos contenidos.

Cada línea contiene el número del campo (5 dígitos), un espacio en blanco y el contenido del campo.

Ejemplo

```
...
<do task="mfncrange">
<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 línea del argumento del elemento *<field>* será almacenada en una nueva incidencia del campo especificado por el atributo *tag*.

Ejemplo

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

field.tag

1. Opciones: list
2. Puede ser usado en: <field> <parm>
3. Sintaxis: tag=...

El atributo **tag** es usado para especificar el número del campo.

Use **tag=list** para informar que la lista de campos será pasada por el argumento del elemento *<field>*.

Ejemplo

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

field.tag=list

Informa que la lista de campos será pasada por el argumento del elemento *<field>*.

Use la coma "," para separar la lista de campos. Use la barra "/" para indicar un intervalo de campos. Use el abre corchetes "[" para informar que el campo será almacenado con el número de campo especificado después del carácter dos puntos ":" y antes del cierra corchetes "]".

Ejemplo

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

field.type

1. Opciones: flag
2. Puede ser usado en: *<field>* *<file>* *<htmlpft>* *<list>* *<pft>*
3. Sintaxis: type=...

Informa el tipo del dato que está siendo accedido.

Ejemplo

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

field.type=flag

Informa que el campo que está siendo accedido del CGI es del tipo presente/ausente.

Si está presente y vacío, el contenido del campo será almacenado con *On*, sino el campo será almacenado con el valor pasado.

Este atributo solamente tiene efecto con el atributo *action=cgi*.

Ejemplo

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

<file>

1. Puede Contener: <pft>
2. Puede ser usado en: <do> <function> <hl> <loop> <section> <update>
3. Atributos: action type
4. Sintaxis: <file> ... </file>

El elemento **<file>** puede ser usado para crear, desbloquear y cerrar bases de datos, crear archivos temporales, borrar archivos y cambiar la salida estándar del IsisScript.

El atributo *action* informa la acción.

El atributo *type* informa el tipo de archivo que sufrirá la acción.

Ejemplo

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

file.action

1. Opciones: append close create delete unlock
2. Puede ser usado en: <field> <file> <flow> <function> <htmlpft> <list> <return>
3. Sintaxis: action=...
Informa la acción a ser ejecutada por el elemento *<file>*.

Ejemplo

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

file.action=append

Abre un archivo de salida para adicionar los textos al final.

Ejemplo

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

file.action=close

Cierra el archivo de salida cuando el atributo *type=output* o una base de datos cuando el atributo *type=database*.

Ejemplo

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

file.action=create

Combinado con el atributo *type=output* la acción consiste en crear un nuevo archivo de salida. Combinado con el atributo *type=database* la acción consiste en inicializar una base de datos.

Ejemplo

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

file.action=delete

Borra un archivo.

Ejemplo

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

file.action=unlock

Desbloquea una base de datos.

Los valores del bloqueo de entrada de datos (Data Entry Lock) y el bloqueo exclusivo (Exclusive Write Lock) son cerados.



Los registros permanecen inalterados.

Ejemplo

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

...

file.type

1. Opciones: database file inverted master output tempfile
2. Puede ser usado en: <field> <file> <htmlpft> <list> <pft>
3. Sintaxis: type=...

Informa el tipo de archivo meta de la acción.

Ejemplo

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

file.type=database

Informa que la acción actúa sobre una base de datos.

Ejemplo

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

file.type=file

La acción incide sobre un archivo.

Válido solamente para la acción *action=delete*.

Ejemplo

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

file.type=inverted

Informa que la acción actúa sobre el archivo invertido de una base de datos.

Válido solamente para la acción *action=create*.

Ejemplo

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

file.type=master

Informa que la acción actúa sobre el archivo master de una base de datos.
Válido solamente para la acción *action=create*.

Ejemplo

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

file.type=output

Informa que la acción actúa sobre el archivo de salida.

Ejemplo

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

file.type=tempfile

Informa que la acción actúa sobre un archivo temporal.
Válido solamente para la acción *action=create*.

El argumento informa el número del campo en que será almacenado el nombre del archivo temporal único retornado por el sistema operativo.

Ejemplo

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

<flow>

1. Puede Contener: <pft>
2. Puede ser usado en: <do> <function> <hl> <loop> <section> <update>
3. Atributos: action
4. Sintaxis: <flow> ... </flow>

El elemento **<flow>** es usado para desviar la secuencia de ejecución de las instrucciones del IsisScript.

El atributo *action* informa la acción.

Ejemplo

```

...
<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. Opciones: exit jump skip
2. Puede ser usado en: <field> <field> <flow> <function> <htmlpft> <list> <return>
3. Sintaxis: action=...
Informa la acción que el elemento <flow> deberá seguir.

Ejemplo

```

...
<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 la ejecución del IsisScript corriente.

El argumento del elemento <flow> informa el código a ser retornado al sistema operativo.

Ejemplo

```

...
<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

Desvía la ejecución del IsisScript al elemento *<label>* correspondiente.
El argumento de la instrucción *<flow>* informa el destino.

Ejemplo

```
...
<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

Desvía la ejecución del IsisScript al inicio del *<loop>* del objetivo corriente o abandona el objetivo corriente y vuelve al objetivo anterior.
El argumento del elemento *<flow>* debe ser *Next* o *Quit* respectivamente.

Ejemplo

```
...
<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. Puede Contener: *<call>* *<cgitable>* *<define>* *<display>* *<do>* *<export>* *<extract>* *<field>* *<file>* *<flow>* *<hl>* *<label>* *<list>* *<parm>* *<proc>* *<return>* *<trace>*
2. Puede ser usado en: *<IsisScript>*

3. **Atributos:** *action from name split tag*

4. **Sintaxis:** `<function> ... </function>`

El elemento **<function>** inicia un bloque de declaración de una función.

Use los atributos *action*, *tag* y *split* para recibir parámetros según lo descrito para el elemento **<field>**.

Ejemplo

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

function.action

Vea: `<field action=...>`

Paso de parámetros para la función.

Tiene la misma funcionalidad del atributo *action* del elemento **<field>**.

Ejemplo

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

function.from

Vea: `<field from=...>`

Paso de parámetros para la función.

Tiene la misma funcionalidad del atributo *from* del elemento **<field>**.

Ejemplo

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

function.name

1. Puede ser usado en: `<call>` `<function>` `<IsisScript>` `<section>`
2. Sintaxis: `name=...`

El atributo **name** identifica la función que está siendo declarada.

Este nombre será utilizado por el elemento `<call>` para llamar la función.

Ejemplo

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

function.split

Vea: `<field split=...>`

Paso de parámetros para la función.

Tiene la misma funcionalidad del atributo *split* del elemento `<field>`.

Ejemplo

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

function.tag

Vea: `<field tag=...>`

Paso de parámetros para la función.

Tiene la misma funcionalidad del atributo *tag* del elemento `<field>`.

Ejemplo

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

<hl>

1. Puede Contener: <call> <cgitable> <define> <display> <do> <export> <extract> <field> <file> <flow> <label> <list> <parm> <proc> <trace>
2. Puede ser usado en: <do> <function> <loop> <section> <update>
3. Sintaxis: <loop> ... </loop>

El elemento <hl> inicia un bloque de instrucciones para destacar texto.

Ejemplo

```
...
<hl>
<parm name="prefix"><b></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. Puede Contener: <pft>
2. Puede ser usado en: <display>
3. Atributos: action type
4. Sintaxis: <htmlpft> ... </htmlpft>

Interpreta y formatea un archivo HTML que contenga instrucciones del lenguaje de formato.

Ejemplo

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

htmlpft.action

1. Opciones: convert
2. Puede ser usado en: <field> <file> <flow> <function> <htmlpft> <list> <return>
3. Sintaxis: action=...

Especifica una acción diferente de la acción estándar del elemento <htmlpft>.

Ejemplo

```

...
<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

Convierte un HTML con especificaciones de formato para un formato, pero no interpreta el formato convertido, muestra la especificación generada.

Ejemplo

```

...
<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

Vea: <pft type=...>

Tipo de acción a ser tomada para ejecución del formato.

Ejemplo

```

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

```

<label>

1. Puede ser usado en: <do> <function> <hl> <loop> <section> <update>
2. Sintaxis: <label> ... </label>

El elemento **<label>** indica un punto para cual el IsisScript podrá desviar la secuencia de ejecución de las instrucciones por medio del elemento *<flow action=jump>*.

Ejemplo

```
...
<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. Puede Contener: <pft>
2. Puede ser usado en: <do> <function> <hl> <loop> <section> <update>
3. Atributos: action type
4. Sintaxis: <list> ... </list>

El elemento **<list>** altera la lista interna del IsisScript. Las opciones son: adicionar ítems a la lista o eliminar todos los ítems de la lista.

El atributo *action* indica la opción.

El atributo *type* indica el tipo de lista.

Ejemplo

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

list.action

1. Opciones: delete load
2. Puede ser usado en: <field> <file> <flow> <function> <htmlpft> <list> <return>
3. Sintaxis: action=...

Informa la acción a ser ejecutada en la lista del IsisScript.

Ejemplo

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

list.action=delete

Elimina todos los ítems de la lista.

Ejemplo

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

list.action=load

Adiciona nuevos ítems a la lista.

Cada línea informada por el argumento será un nuevo ítem en la lista.

Ejemplo

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

list.type

1. Opciones: freq list sort
2. Puede ser usado en: <field> <file> <htmlpft> <list> <pft>
Informa el tipo de almacenamiento en la lista.

Ejemplo

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

list.type=freq

Lista de conteo de frecuencia de ítems.

Un ítem repetido no es insertado en la lista, es adicionado en el total de incidencias de aquel ítem.

Ejemplo

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

list.type=list

La lista de ítems sin ordenación.

Ejemplo

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

list.type=sort

La lista es ordenada por el contenido del ítem.

Ejemplo

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

<loop>

1. Puede Contener: <call> <cgitable> <define> <display> <do> <export> <extract> <field> <file> <flow> <hl> <label> <list> <parm> <proc> <return> <trace>
2. Puede ser usado en: <do>
3. Sintaxis: <loop> ... </loop>

El elemento **<loop>** indica un grupo de instrucciones que serán repetidas para todos los datos encontrados de acuerdo con el tipo de tarea especificada en el elemento **<do>** correspondiente.

Ejemplo

```
...
<do task=search>
```

```

<parm name=db> <pft>v2001</pft></parm>
<parm name=expression><pft>v2005</pft></parm>
<loop>
<display><pft>mfnt</pft></display>
</loop>
</do>
...

```

<parm>

1. Puede Contener: <pft>
2. Puede ser usado en: <do> <function> <hl> <loop> <section> <update>
3. Atributos: name tag type
4. Sintaxis: <parm> ... </parm>

El elemento **<parm>** informa un parámetro al bloque de instrucciones al cual pertenece.

Ejemplo

```

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

```

parm.name

1. Opciones: 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. Puede ser usado en: <call> <function> <IsisScript> <parm> <section>
3. Sintaxis: name=...
Nombre del parámetro.

Ejemplo

```

<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

Cambia la tabla de caracteres alfabéticos de IsisScript durante la sección corriente.

La tabla de caracteres alfabéticos informa, para la actualización del archivo invertido y para la extracción de claves, cuáles caracteres serán considerados como alfabéticos.

Los caracteres que no están en la tabla serán considerados como delimitadores.

En una sección sin la opción *<parm name=actab>* el IsisScript asume la tabla ANSI.

Ejemplo

```

<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=bufferize

Permite alterar el tamaño del "buffer" interno (en bytes) del WXIS que es usado para almacenar el resultado de la formatación.

Ejemplo

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

parm.name=cipar

Activa una tabla de indicación de nombres lógicos con nombres físicos de archivos para la sección corriente.

Cada línea contiene una indicación, a la izquierda del carácter = (igual) está el nombre lógico y a la derecha el nombre físico del archivo.

El carácter * (asterisco) indica que la indicación vale para cualquier archivo de base de datos.

Ejemplo

```
...
<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 la cantidad de veces que el grupo de instrucciones del elemento *<loop>* será ejecutado.

Ejemplo

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

```

parm.name=db

Informa la base de datos a ser accedida en las siguientes tareas:

```

task=mnrange
task=keyrange
task=search
task=update
task=fullinversion
task=mastersort
task=invertedload

```

Ejemplo

```

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

```

parm.name=decod

Informa la base de datos de parámetros de expansión de campos decodificados.

Ejemplo

```

...
<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>mf/</pft></display>
</loop>
</do>
...

```

parm.name=delimiter

Informa el separador de campos para importación de registros en la opción *RLine*. En la ausencia de este parámetro se asume el carácter | (pipe).

Ejemplo

```
...
<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 el tiempo máximo que el registro permanecerá bloqueado. Después de ese período, el registro puede ser bloqueado por otro usuario (otra identificación).

Ejemplo

```
...
<do task=update>
<field action=cgi tag=2001>db</field>
<field action=cgi tag=2002>mfnc</field>
<parm name=db><pft>v2001</pft></parm>
<parm name=mfnc><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 la expresión de investigación.

Ejemplo

```

...
<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 el nombre del archivo que será importado o exportado.

Ejemplo

```

...
<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 el valor a ser adicionado en la adición de nuevos ítems en la lista de frecuencia.

Ejemplo

```

...
<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 cuál es el primer ítem a ser accedido por el grupo del elemento *<loop>*.

Ejemplo

```

...
<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 la FST (Field Select Table) que será usada para actualización del archivo invertido o para extracción de claves.

Ejemplo

```

...
<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 la base de datos de parámetros de conversión de contenido.

Ejemplo

```

...
<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 la lista de índices para la investigación en la base de datos.

Ejemplo

```

...
<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

Clave para ordenación de la base de datos.

Ejemplo

```

...
<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 el campo que es la clave de ordenación de la base de datos.

Ejemplo

```

...
<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

Tamaño de la clave de ordenación de la base de datos.

Ejemplo

```

...
<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 la lista de claves para el esquema de realce de texto.

Ejemplo

```

...
<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=keybdb

Base de datos que contendrá las claves que serán extraídas, si es usada como parámetro para el elemento *<extract>*.

Base de datos con las claves ordenadas para carga del archivo invertido si es usada como parámetro del elemento *<do task=invertedload>*.

Ejemplo

```
...
<do task=invertedload>
<parm name=db><pft>v2001</pft></parm>
<parm name=keybdb><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 bloqueo de registro.

Ejemplo

```
...
<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 claves (por registro) en la extracción por FST (Field Select Table).

En la ausencia de este parámetro el IsisScript asume 1024.

Ejemplo

```

...
<do task=fullinversion>
<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 inversion:
',v2001/</pft></display>
<loop></loop>
<display><pft>'Finished.'/</pft></display>
<display><pft>'Lock status =
'v1102/</pft></display>
</do>
...

```

parm.name=mfn

Número del registro a ser actualizado, o *New* para indicar que será un nuevo registro de la base de datos, o *GetNew* para indicar que será un nuevo registro de la base de datos conteniendo los campos importados del registro del objetivo anterior del IsisScript.

Ejemplo

```

...
<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

Cantidad de postings para cada clave.

Utilice ALL para indicar "todos los postings".

Ejemplo

```

...
<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 el número del campo del posting a ser accedido en el intervalo de claves.

Ejemplo

```
...
```

```

<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

Prefijo a ser insertado en el esquema de realce de texto, o prefijo a ser usado por el elemento *<htmlpft>*.

Ejemplo

```

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

```

parm.name=reset

Permite que la actualización del archivo invertido no altere la información de registro del archivo maestro con inversión pendiente. Aplicable para bases de datos con múltiples archivos invertidos.

Ejemplo

```

...
<do task="fullinversion">
<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 inversion:
CDS' /</pft></display>
<loop></loop>
<display><pft>'Finished.' /</pft></display>
<display><pft>'Lock status =
'v1102 /</pft></display>
</do>
...

```

parm.name=reverse

Informa que los registros resultados de la tarea especificada en el elemento <do> serán accedidos en orden inversa.

Ejemplo

```

...
<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 el formato de ordenación de la lista interna del IsisScript.

Ejemplo

```

...
<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_Items</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>
ay>
</loop>
</do>

```

...

parm.name=stw

Informa el archivo con la tabla de palabras que no deben ser incluidas en el archivo invertido (Stop Word Table) y que será usada para actualización del archivo invertido o en la extracción de claves.

Ejemplo

```
...
<do task=fullinversion>
  <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 inversion:
  ',v2001</pft></display>
  <loop></loop>
  <display><pft>'Finished. ' </pft></display>
  <display><pft>'Lock status =
  'v1102</pft></display>
</do>
...
```

parm.name=suffix

Sufijo a ser insertado en el esquema de realce de texto, o sufijo a ser usado por el elemento `<htmlpft>`.

Ejemplo

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

parm.name=task

Indica el tipo de tarea que será utilizada por el elemento `<loop>`. Tiene efecto si no es especificado el atributo `task` del elemento `<do>`.

Ejemplo

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

```

<loop>
...
</loop>
</do>
...

```

parm.name=to

Informa cuál es el último ítem a ser accedido por el grupo del elemento *<loop>*.

Ejemplo

```

...
<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 el tipo de archivo para exportación o importación.

Los tipos posibles son: *ISO2709*, *HLine*, *RLine* y *VLine*.

ISO2709 es una norma ISO (International Standards Organization) pero limita el número de identificación de los campos en 3 dígitos.

HLine es el más eficiente, utiliza el comando H del elemento *<proc>*.

RLine es usado solamente para importación, donde cada línea de un archivo secuencial corresponde a un registro.

VLine es el recomendado para permitir modificación vía editor de texto.

Ejemplo

```

...
<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

Cambia la tabla de conversión de carácter para mayúscula del IsisScript durante la sección corriente.

Esta tabla informa, para la actualización del archivo invertido, para la extracción de claves y para la opción mode del lenguaje de formato, todos los caracteres correspondientes de minúscula o mayúscula con acentuación para la correspondiente mayúscula sin acento.

En una sección sin el elemento `<parm name=uctab>` el IsisScript asume la tabla ANSI.

Ejemplo

```
<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. Puede ser usado en: `<field> <parm>`
2. Sintaxis: `tag=...`

El atributo **tag** es usado para especificar el número del campo.

Ejemplo

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

parm.type

1. Opciones: check
2. Puede ser usado en: <do>
3. Sintaxis: type=check
Especifica el tipo del parámetro.

Ejemplo

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

parm.type=check

Permite la verificación de la sintaxis de una FST (Field Select Table). actualiza el campo especificado por el atributo tag con el código de error (5 dígitos), un espacio y el punto en que fue detectado el error de sintaxis, o 00000 si no hay error de sintaxis.

Ejemplo

```
...
<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. Puede Contener: <pft>
2. Puede ser usado en: <call> <cgitable> <define> <display> <export> <extract> <field> <file> <flow> <htmlpft> <label> <list> <parm> <pft> <proc> <return> <trace> <write>
3. Atributos: type
4. Sintaxis: <pft> ... </pft>
Formatea el registro corriente.

Ejemplo

```
...
<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. Opciones: check reload
2. Puede ser usado en: <field> <file> <htmlpft> <list> <pft>
3. Sintaxis: type=...
Tipo de acción a ser tomada para ejecución del formato.

Ejemplo

```

...
<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 la verificación de la sintaxis de un formato. Devuelve el código de error (5 dígitos), un espacio y el punto en que fue detectado el error de sintaxis o en caso de que no haya error de sintaxis.

Ejemplo

```

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

```

pft.type=reload

Use esta opción para informar al IsisScript que el formato tendrá que ser recompilado cada vez que esta instrucción sea ejecutada.

Ejemplo

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

<proc>

1. Puede Contener: <pft>
2. Puede ser usado en: <do> <function> <hl> <loop> <section> <update>
3. Sintaxis: <proc> ... </proc>
Modifica el contenido del registro corriente.

Ejemplo

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

<return>

1. Puede Contener: <pft>
2. Puede ser usado en: <function>
3. Atributos: action split tag
4. Sintaxis: <return> ... </return>
Sale de la función corriente.

Ejemplo

```
...
<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

Vea: <field action=...>

Devolución de parámetros para quien llamó la función. Tiene la misma funcionalidad del atributo *action* del elemento <field>.

Ejemplo

```

...
<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

Vea: <field split=...>

Devolución de parámetros para quien llamó la función. Tiene la misma funcionalidad del atributo **split** de la instrucción <field>.

Ejemplo

```

...
<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

Vea: <field tag=...>

Devolución de parámetros para quien llamó la función. Tiene la misma funcionalidad del atributo *tag* del elemento <field>.

Ejemplo

```
...
<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

Vea: <field tag=...>

Devolución de parámetros para quien llamó la función. Tiene la misma funcionalidad del atributo *tag* de la instrucción <field>.

Ejemplo

```
...
<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. Puede Contener: <call> <cgitable> <define> <display> <do> <export> <extract> <field> <file> <flow> <hl> <label> <list> <parm> <proc> <trace>
2. Puede ser usado en: <IsisScript>
3. Atributos: name

4. Sintaxis: <section> ... </section>

El elemento **<section>** es usado para iniciar una secuencia de instrucciones que acceden campos comunes y utilizan tablas comunes.

El atributo *name* puede ser utilizado para identificación.

Ejemplo

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

section.name

1. Puede ser usado en: <call> <function> <IsisScript> <section>

2. Sintaxis: name=...

El atributo **name** es opcional, cuando es usado sirve para identificar la sección.

Ejemplo

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

<trace>

1. Puede Contener: <pft>

2. Puede ser usado en: <do> <function> <hl> <IsisScript> <loop> <section> <update>

3. Sintaxis: <trace> ... </trace>

Activa o desactiva la muestra de la instrucción que está siendo ejecutada. Los modos posibles son normal, línea a línea o tabla, respectivamente: *On*, *BR* y *Table*.

Ejemplo

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

<update>

1. Puede Contener: <call> <cgitable> <define> <display> <do> <export> <extract> <field> <file> <flow> <hl> <label> <list> <parm> <proc> <return> <trace> <write>
2. Puede ser usado en: <do>
3. Sintaxis: <update> ... </update>
Inicia un bloque de instrucciones para modificación o adición de un registro.

Ejemplo

```

...
<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. Puede Contener: <pft>
2. Puede ser usado en: <update>
3. Sintaxis: <write> ... </write>

Elemento que graba la modificación del registro.

Si el elemento <parm name=mfn> indica *New* o *GetNew*, entonces adiciona un nuevo registro, sino actualiza el mfn pasado como argumento.

Si el argumento del elemento <write> es *Unlock*, el registro será desbloqueado después de ser grabado; si es *Lock*, el registro será grabado y bloqueado; si es *NoUnlock*, el registro permanecerá bloqueado y la información de bloqueo permanecerá igual; si es *Delete*, el registro será borrado.

Ejemplo

```

...
<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>  
...
```

Citas 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.

Glosario

- **Archivo.** En computación, un conjunto de datos que se puede grabar en algún dispositivo de almacenamiento. Los archivos de datos son creados por aplicaciones, como por ejemplo un procesador de textos.
- **Archivo invertido.** Conjunto de seis archivos físicos, cinco de los cuales contienen los términos de búsqueda del diccionario (organizados como un árbol B*) y el sexto contiene la lista de apuntadores asociadas a cada término. A fin de optimizar el almacenamiento en disco, se mantienen dos árboles B* por separado: uno para los términos de hasta 10 caracteres (almacenados en los archivos *.N01* y *.L01*) y otro para los términos de más de 10 caracteres (almacenados en los archivos *.N02* y *.L02*). El archivo *.CNT* contiene campos de control para ambos árboles B*). En cada archivo del árbol B* el archivo *.N0x* contiene los nodos del árbol y el archivo *.L0x* contiene las hojas. Los registros de las hojas apuntan al lugar donde se encuentran los apuntadores que contienen la información para localizar los registros (postings) en la base de datos. Este archivo se identifica con la extensión *.IFP*.

- **Backup.** Procedimiento en el que uno o más archivos y/o directorios son duplicados para otro dispositivo de almacenamiento (cinta o disco), para producir una copia de seguridad, que puede restaurarse en el caso de que algún dato sea borrado accidentalmente o si ocurrió daño físico de los datos originales.
- **Base de datos.** Colección de datos estructurados para que sea posible acceder a ellos y manipularlos fácilmente. Está formada por unidades denominadas registros, cuyos diversos atributos son representados por campos y subcampos. Por ejemplo, en un archivo "catastro de clientes", cada cliente representa un registro, que posee varios campos, como "NOMBRE", "CÓDIGO DEL CLIENTE", "TELÉFONO" etc.
- **Bases de datos bibliográfica.** Versión electrónica de un catálogo o índice bibliográfico.
- **Campo.** Elemento de un registro que permite almacenar información específica. Ver *Base de datos*.
- **CDS/ISIS - MicroISIS.** Software desarrollado y mantenido por la UNESCO para el tratamiento de datos bibliográficos.
- **CGI.** Es un padrón para conectar aplicaciones externas con los servidores de información, como el HTTP o servidores del Web.
- **Clave.** Expresión que identifica una o más informaciones de determinada clase o tipo y que puede ser usada en la búsqueda.
- **Formato de presentación.** Conjunto de comandos que determinan como debe ser la salida de datos de una base de datos ISIS.
- **Formato electrónico.** Cualquier forma de almacenamiento, recuperación y presentación de información pasible de transmisión online o grabación en medios magnéticos u ópticos.

- **Formato ISO (de intercambio de datos).** Patrón establecido por la ISO para intercambio de datos entre instituciones, redes y usuarios. Se refiere la norma ISO 2709.
- **Formato LILACS.** Formato de descripción bibliográfica establecido por BIREME, basado en la UNISIST Reference Manual for Machine-readable Bibliographic Descriptions.
- **Indización.** Procedimiento de identificar y describir el contenido de un documento con términos que representan los temas correspondientes a ese documento, con el objetivo de recuperarlo posteriormente.
- **Posting.** Consiste de la dirección de una clave extraída del archivo maestro.
- **Registro.** Conjunto estructurado de datos que permite almacenar determinado asunto. Ver *Base de datos*.
- **Subcampo.** Elemento que contiene la menor parte de información de un campo, cuyo sentido puede no ser claro si no fuera analizado en conjunto con los otros elementos relacionados.
- **URL.** Patrón definido para direccionamiento de contenidos de datos vía protocolo TCP/IP. Los navegadores de internet utilizan la URL para acceder a páginas en la web.