# UNIVERSIDAD AUTONOMA DE NUEVO LEON

FACULTAD DE CIENCIAS FISICO-MATEMATICAS

# "F O C U S"

## T E S I S

QUE EN OPCION AL TITULO DE
LICENCIADO EN CIENCIAS COMPUTACIONALES

PRESENTA

ELIZABETH PEREZ LOZANO

SAN NICOLAS DE LOS GARZA, N. L. MARZO DE 1987.

DEDICATORIAS

Gracias a Tí Señor
por haberme dado
vida y salud, para
que yo obtuviera una
de mis metas mas
importantes.

A mis Padres, ya
que gracias a su
gran apoyo y sacrificios
he podido alcanzar una
meta muy importante.

Gracias a mis Amigos
por su gran ayuda que
me brindaron en todos
los aspectos.

Y a Tí Carlos, gracias
por estar conmigo y darme
ese apoyo tan especial y
y alentador que lo había
necesitado por siempre.

Sección I.- Preparación de Reportes


| | |
|---|---|
| Introducción | 1-01 |
| Conceptos Generales | 1-02 |
| Preparando Reportes | 1-02 |
| Sistema Operativo | 1-03 |
| Descripción de Archivos | 1-03 |
| Comandos | 1-05 |
| Formas Requeridas | 1-07 |
| El Comando TABLE | 1-07 |
| Series de Pedidos para el mismo Archivo | 1-07 |
| Testamentos que se Requieren | 1-08 |
| Definiendo Cálculos | 1-11 |
| Sumarizando Elementos Requeridos (Resumen) | 1-13 |
| Recomendaciones | 1-13 |
| Convencionalismos de un Estatuto | 1-14 |
| Espacios entre Palabras | 1-14 |
| Nombres de Campos de Datos | 1-14 |
| Desplegados de Nombres de Campos | 1-15 |
| Corrección de Errores | 1-15 |
| El Estatuto HELP | 1-16 |
| Indicación del Estatuto | 1-16 |
| Terminando ó Continuando con el Comando TABLE | 1-16 |
| Convensiones de Impresión | 1-17 |
| OFFLINE, ONLINE | 1-18 |
| Copias Múltiples, el Estatuto RETYPE | 1-19 |
| Reportes Anchos en Pantalla.- Paneles | 1-19 |
| Elementos del Lenguaje de Petición | 1-20 |
| Verbos | 1-21 |
| Orden de Aparición | 1-22 |
| PRINT, LIST | 1-22 |
| SUM | 1-23 |
| Campos de Datos Numéricos | 1-24 |
| Objetos Verbales Múltiples | 1-24 |
| Sinónimos de SUM, WRITE, ADD | 1-25 |
| Conteo del Campo | 1-25 |
| Sumando y Contando | 1-26 |
| Prefijos CNT. y SUM. | 1-27 |
| Contado las Entradas | 1-27 |
| Contando en un Archivo Múltiple | 1-28 |
| Utilizando el COUNT en un Reporte Matriz | 1-29 |
| Totales de Columnas y Renglones | 1-30 |
| Una Manera Simple de Referirse a Todos Los Campos | 1-30 |
| Enunciados Múltiples, Juegos de Verbos | 1-31 |

Operaciones Realizadas Directamente con Objetos Verb. 1-34
Datos Faltantes.- Prefijo (ALL) 1-36

-Selección de Registros-

Criterios de Selección 1-38
Probando Valores Acumulados 1-39
Selección TOTAL con ACROSS 1-39
Relación de la Selección de TOTALES con Archivo HOLD 1-41
Leyendo los Archivos Prueba de un Archivo 1-41
Condiciones de Control.- Frase ON 1-44
Cálculos 1-45
El Comando DEFINE 1-45
Definir Nombres de Campo 1-49
Archivos Múltiples con Valores DEFINE 1-49
Expresiones de Cálculo 1-50
Funciones Especiales 1-51
Funciones de Fecha 1-52
Refiriendose a Operaciones Directas en Cálculos 1-53
Column-Totals 1-53
Row-Totals 1-54

-Archivos Externos-

Archivos Externos 1-56
Reteniendo un Reporte.- El Archivo HOLD 1-56
Sintáxis del HOLD 1-57
Produciendo un Archivo Externo.- La opción SAVE 1-58
Renombrando el Archivo Externo.- La opción SAVE AS 1-59
El Comando FILEDEF 1-60
Almacenando despues de Imprimir 1-61
El Comando JOIN 1-61
Control de Formato 1-63
Encabezados de Reporte 1-63

Pie de Reporte 1-65
Líneas de Encabezados 1-65
Líneas de Sub-encabezados 1-66
Líneas de Contenido 1-66
Titulos.- La Frase AS 1-67
Líneas de Subtotales 1-68
Líneas Utilizando El Comando RECAP 1-68
Líneas de SUBFOOT 1-68
Ancho de Columnas 1-68
Espacio entre Columnas 1-69

IN y ACROSS 1-69
IN con OVER y FOLD-LINE 1-70
Operaciones con Datos en Encabezados 1-70
Centrando Encabezados y Pie de Página 1-71
Tópicos Especiales 1-72
Formato de Datos 1-72
Cambiando las Posturas de Omisión 1-74
El Comando SET 1-75
Uso del Comando SET 1-76

Descripción de los Archivos FOCUS 2-01

Introducción 2-01
Convenciones Generales 2-01
Relación de Segmentos 2-01
CROSS-REFERENCE 2-02
Ilustración de Datos 2-03
Creando Descripción de Archivos FOCUS 2-04
Verificando una Descripción de Archivo 2-05
Cambiando un Archivo Maestro 2-06
Almacenando Archivos FOCUS y Descripción de Archivos 2-06
Archivo Fuente 2-07
Estructura de los Archivos 2-08
Características de los Archivos 2-09
Características de los Segmentos 2-09
Características de los Campos 2-09
Estructura de un Archivo con Múltiples Trayectorias 2-10
Segmento Unico 2-11
Preparando un Segmento por Partes: FIELDTYPE=I 2-13
CROSS-REFERENCE. Dinámico-Usando el Comando JOIN 2-14
Funciones de Comando USE 2-17
Cambiando el Default FILETYPE ó FILEMODE 2-17
Identificando un Archivo con un Nombre Diferente 2-18
Identificando un Archivo de Datos 2-18
Protegiendo un Archivo de Cambios 2-18
Nueva Base de Datos 2-19
Administración de la Base de Datos 2-19
Estableciendo Identidad de Usuarios 2-20
Accesos de Niveles Necesarios, ACCESS= 2-20

Mantenimiento de Archivos FOCUS

Conceptos Generales 3-01
Transacciones de Entradas de una Pantalla completa CRT: 3-04

| | |
|---|---|
| Proceso de Mantenimiento de Archivo Básico | 3-04 |
| Secuencia de Subcomandos | 3-05 |
| Describiendo Transacciones.- Fuentes de Datos | 3-06 |
| El Subcomando FIXFORM | 3-06 |
| Espacion en Blanco Insertados | 3-06 |
| Valores de Datos Binarios | 3-08 |
| Punto Decimal Implícito | 3-09 |
| Describiendo Grupos Repetitivos | 3-09 |
| Describiendo Valores de Datos que Pueden no Estar Presentes (Datos Condicionales) | 3-10 |
| Actualizando una Lista Variable de Campos | 3-11 |
| Describiendo Transacciones (Formato Libre ) | 3-11 |
| Espacios en Blanco Indicadores | 3-13 |
| Datos Conteniendo Caracteres Especiales | 3-13 |
| Subcomando FREEFORM | 3-14 |
| Subcomando PROMPT | 3.16 |
| Indicaciones Para Valores de Transacciones | 3-16 |
| Corrigiendo una Respuesta Anterior | 3-17 |
| Escribiendo hacia Adelante | 3-17 |
| Indicando para Grupos de Campos | 3-18 |
| Respuesta de Datos Faltantes | 3-20 |
| Repetición de Ultima Petición | 3-20 |
| Cancelando Respuestas Anteriores | 3-20 |
| Calculando Valores Nuevos | 3-22 |
| Cálculo y Validez después de MATCH o NEXT | 3-23 |
| Concordando Transacciones Registrar de Base de Datos | 3-25 |
| Incluyendo Los Registros en Niveles de Detalles de un Archivo | 3-26 |
| Cortando Campos sin Clave | 3-26 |
| El Subcomando NEXT | 3-27 |
| Acciones cuando no concuerdan | 3-31 |
| Acciones después de NEXT o NONEXT | 3-33 |
| Manteniendo Segmentos Unicos | 3-34 |
| Poniendo al Corriente con Transacciones de Formato Libre | 3-36 |
| Actualizando Registros en Indice | 3-36 |
| Condiciones de Omisión | 3-38 |
| Control de Registros Duplicados ON MATCH INCLUDE | 3-39 |
| Errores de Apunte y Mensajes de Error | 3-39 |
| El Subcomando TYPE | 3-41 |
| Subcomandos de Control de Transacciones | 3-43 |
| Fuentes de Transacciones Mixtas | 3-45 |
| El Subcomando END | 3-45 |
| START y STOP | 3-46 |
| CASE lógico | 3-47 |
| Los Subcomandos CASE y ENDCASE | 3-47 |
| Reglas del CASE | 3-48 |

Mantenimiento del Registro con casos Múltiples 3-49
Estableciendo una Posición en el Archivo 3-49
Concordando Segmentos Múltiples en un Comando Sencillo 3-50
Activando Campos 3-51
Procesamiento Parcial de Transacciones FIXFORM 3-53
Mezclando Tipos de Transacciones 3-54
Reconstruyendo un Archivo FOCUS.-
Utilizando el Comando REBUILD 3-57

Editor Interactivo SCAN

Uso del SCAN 4-01
Controlando la Pantalla 4-02
Desplegando en Forma de Filas 4-03
Reteniendo Comandos 4-05
Concepto de Posición Actual 4-05
Posicionamiento de Registros Específicos 4-05
Cambiando Valores en Campo 4-06
Borrando Registros 4-06
Agregando Registros 4-06
Borrando la Pantalla 4-07
El Subcomando DISPLAY 4-07
Terminando la Sección SCAN 4-08

Dialoguer Manager

Introducción 5-01
Nombre de los Archivo FOCUS 5-01
Ejecutando Archivos FOCUS 5-02
Procedimientos Omitidos 5-02
Procedimientos con Valores Omitidos 5-03
Nombre de Variables Posicionales 5-04
Reemplazando Valores para Variables Sustituibles 5-04
Reemplazando Valores en la Linea EXEC 5-04
Valores Sustituibles 5-05
Sintáxis para Valores Sustituibles 5-06
Contestación Implicada 5-07
Preguntas Directas 5-07
Uso de una Linea de Respuesta en el Modo Directo 5-08
Variables del Sistema 5-09
Variables Globales 5-10

Estatutos de Control del Dialoguer Manager 5-11
Conceptos de Ejecución 5-13
Expresiones Compuestas 5-14
Probando la Longuitud, Tipo y Existencia
de una Variable 5-14
Comando TYPE 5-16
Comando SET 5-16
Terminación de un Procedimiento 5-17
Comando READ 5-18
Otras Facilidades del Dialoguer Manager 5-19
Suprimiendo la Ejecución 5-19
Ocultando Procedimiento FOCEXEC 5-20
Borrando la Pantalla para la Selección de un Menu 5-20
Entrada de Datos en la Forma de Pantalla LLena 5-20

FIDEL

Lenguaje Interactivo para Entrada de Datos FOCUS 6-01
Describiendo la Pantalla CRT: 6-02
Sustitución de Campo de Datos 6-04
Limpiando la Pantalla 6-05
Pantallas Continuas 6-06
Usando Pantallas a lo Largo 6-07
Opciones de Procesamiento 6-07
Opción MATCH/NOMATCH 6-09
Múltiples CRTFORMS 6-10
Combinación de Pantallas (LINE) 6-10

INTRODUCCION:

El sistema "FOCUS" es un sistema comprensivo para el control de información. Este sistema cuenta con las facilidades para describir archivos simples y archivos completamente interconectados; para insertar, cambiar y borrar registros archivados y para elaborar reportes, utilizando la informacion archivada.

El propósito de "FOCUS" es el de controlar una aplicacion entera y asi reducir la necesidad de sustituir la programación de la computadora. El sistema esta estructurado para programadores y para personas que no lo son. Las reglas del "FOCUS" son reglas lógicas y se ha hecho el intento de evitar nociones criticas inesperadas por computadora. Esto le permite a los usuarios que hagan referencia de nuevas combinaciones a sus aplicaciones especiales.

Este manual esta dividido en varias secciones e incluye el método para describir archivos FOCUS, como mantenimiento de información y cómo elaborar reportes utilizando la información contenida en los archivos. Porque el lenguaje para solicitar reportes también se puede usar para archivos existentes que no son FOCUS, la primera sección de éste manual está dedicado a cómo elaborar reportes, no se necesita mucho conocimiento sobre como estructurar un archivo para poder usar el lenguaje solicitando reportes.

SECCION (1).- PREPARACION DE REPORTES

Instrucciones en inglés son usadas en la preparación de reportes, éstas proveen las direcciones por la cual los registros han sido traídos, que cálculos han sido ejecutados, como las lineas de los reportes resultantes por lo cual han sido sorteados.

SECCION (2).- DESCRIPCION DE ARCHIVOS EXTERNOS

Estos son archivos existentes, que son la combinación de uno o varios archivos FOCUS. Este tipo de archivos no requiere de mantenimiento FOCUS.

SECCION (3).- DESCRIPCION DE ARCHIVOS FOCUS

Es cuando la información es almacenada en archivos focus. Toda esta parte puede ser interrelacionada de una manera comprensiva, este mantenimiento de información es ejecutado facilmente en un medio ambiente controlado. Proveen las facilidades de ofrecer los archivos, designar un rango de opciones y para jerarquias.

SECCION (4).- MANTENIMIENTO DE BASE DE DATOS FOCUS

Un lenguaje simple es usado para el control de todos los procesos, como la suma de nuevos registros a una base de datos focus, borrar registros y cambiar valores de registros existentes.

SECCION (5).- SCAN-FACILIDAD DE EDICION INTERACTIVA

La facilidad del comando SCAN es mostrarnos información de un archivo determinado, la cual esta aplicacion es solo de uso para el analista, ya que podemos cambiar y borrar información existente, con facilidad pero con mucho riesgo para el usuario en caso de su uso.

SECCION (6).- FOCEXEC, FACILIDAD DIALOGUER MANAGER

Puede ser catalogado como un procedimiento para ejecución repetitiva o requiere de una pregunta o respuesta, o sea la máquina dialoga con el usuario, puede suplir el texto de la pregunta, checando las respuestas. Un procedimiento es invocado simplemente tecleando una letra o un número.

SECCION (7).- FIDEL-FOCUS LENGUAJE DE ENTRADA DE DATOS

Sistema de entrada de datos interactivos como parte de la facilidad del mantenimieto de datos.

SECCION (8).- CONCEPTOS GENERALES

A) PREPARACION DE REPORTES:

Los reportes son preparados en focus, tecleando estatutos imperativos en inglés, consistiendo de uno o más instrucciones. Todos estos estatutos contienen toda la información que el ususario tienda a proveer con una orden, da registros, sorteo de líneas, ejecuta cualquier cálculo, acumula totales, etc.

Los estatutos requeridos no solamente controlan la selección de registro, también la apariencia de reportes de impresión. Esto generalmente simplifica el proceso de preparación de un reporte.

Algunas veces, es necesario que estén varios estatutos en orden para preparar el reporte final.

SISTEMA OPERATIVO:

FOCUS es usado por varios sistemas operativos y monitores de comunicacin. Este no esta virtualmente diferenciado entre su uso entre varios sistemas, pero cada medio ambiente requiere de diferentes protocolos para nombrar y catalogar archivos FOCUS.

FOCUS es usado más comunmente con los sistemas operativos y monitores de comunicación siguientes:

| SISTEMA OPERATIVOS | MONITORES/COMUNICACION |
|---|---|
| VM/370 | CMS |
| MVS | TSO |
| MVS | IMS/DC |
| MVS | CICS |
| PC-DOS | (INTEGRATED) |

C) DESCRIPCION DE ARCHIVOS:

Un orden de escribir una instrucción o varias intrucciones, con las cuales estos estatutos son usados para la descripción de los archivos los estatutos indican su disponibilidad. Estas descripciones son almacenadas en un directorio llamado "MASTER FILE".

1.- ARCHIVO MAESTRO (MASTER FILE)

Cada archivo de datos tiene asociado un archivo maestro en el cual están definidos los campos de los que está formado el archivo. (indica su longuitud, e inclusive su formato alfanumérico, numérico, entero, etc...). Este archivo es definido con una extensión ".MAS"

Ejémplo:

FILENAME.MAS

Donde:

MAS----------> MASTER
FILENAME----------> LONGUITUD DE 8 CARACTERES.
(NUMERICOS O ALFANUMERICOS)

## 2.- ARCHIVOS DE DATOS FOCUS

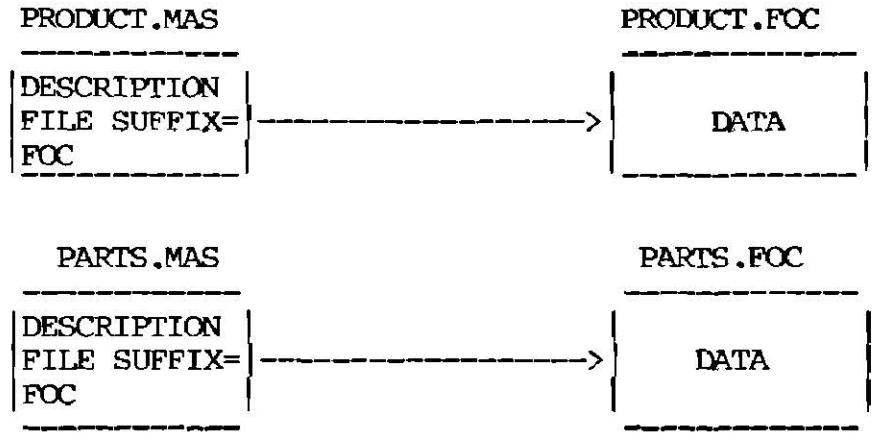
Los registros de datos estan dados en un archivo FOCUS (.FOC), son generalmente almacenados en un archivo fisico, el cual está referenciado con un nombre de archivo.

Por Ejémplo:

UN ARCHIVO FOCUS NOMBRADO PRODUCT===

ES: PRODUCT.FOC

EJEMPLO DE ARCHIVO FOCUS:




Los archivos FOCUS pueden ser solamente creados atravéz de el uso del sistema manejador de datos FOCUS y comandos asociados.

## 3.- ARCHIVOS EXTERNOS DE DATOS:

Un archivo externo (no focus) puede ser descrito de una manera similar a uno de los usos de archivos FOCUS. La descripción de los datos es puesta en un archivo maestro. Los datos pueden residir en una cinta o disco. La correspondencia entre el nombre seleccionado por el archivo y el nombre la cual los datos estan almacenados en cinta o disco son proveidos por un comando FOCUS, "FILEDEF".

Ejémplo:

SALES.DAT, ENTONCES NO ES NECESARIO "FILEDEF"

Si la convención simple de usar el tipo de archivo es DAT, se observa un paso molesto, este puede ser evitado. FOCUS conoce sobre cual nombre y tipo de referencia el archivo es almacenado en disco.

Cuando FOCUS crea un archivo temporal es dado un tipo de archivo para (.FTM).

Por ejémplo:

```
HOLD.FTM FOCUS AUTOMATICAMENTE BUSCA UN
ARCHIVO CON ESTE TIPO, TAN BIEN COMO (.DAT)
```

Archivos externos, los cuales tienen un formato fijo de registros tienen que dar un archivo FOCUS con SUFFIX igual a FIX.

Ejémplo:

```
DESCRIPCION DE ARCHIVOS EXTERNOS Y ARCHIVOS DE DATOS

  RIVER.MAS                          RIVER.DAT
 ------------                       ------------
| DESCRIPCION |                    |            |
| FILE SUFFIX=|------------------->|    DATA    |
| FIX         |                    |            |
 ------------                       ------------
```

Si los archivos tienen esquema con registros de formato libre los campos de datos son separados por comas.

D) COMANDOS:

FOCUS contiene muchos medio ambientes para entrada de nuevos datos, ejecucióon de registros de mantenimiento, etc. El medio ambiente para preparar reportes es generado por el comando "TABLE". El medio ambiente para ejecutar cálculos entre los datos que forman parte del archivo, es por medio de la utilización del comando "DEFINE".

Los comandos específicos que son usados por la preparacióon de reportes son:

| COMANDOS: | ACCION: |
|---|---|
| TABLE | Este comando nos ayuda a obtener resultados finales de un archivo de datos, que son los reportes. Aquí podemos hacer combinaciones entre los campos del archivo de datos. |

| | |
|---|---|
| DEFINE | Nos ayuda a hacer operaciones matemáticas y lógicas utilizando o haciendo referencia a un archivo de datos. |
| RETYPE | Se utiliza cuando terminamos de imprimir un reporte, tecleando <RETYPE> automaticamente se repite la impresión. |
| ON LINE | Nos indica que no va permitir imprimir reporte determinado. |
| OFF LINE | Nos indica que nos permitira imprimir un reporte. |
| SAVE | Almacenada una lista puede ser FOCEXEC ó MASTER. |
| EX | Ejecución de un sistema. |
| FIN | Se ocupa para salir completamente de FOCUS. |
| HELP | Muestra información en la pantalla acerca de varios tópicos de FOCUS. |

El estatuto TABLE

Se indica el estatuto TABLE para solicitar un reporte tabular. El estatuto puede empezar en cualquier parte de la línea.

El nombre del archivo de datos de donde se deben de retraer los registros se pide despues. Se puede emitir un regreso de carril (return) y se puede poner el nombre del archivo en la segunda línea ó el estatuto y el nombre del archivo se puede poner en la misma línea.

Ejémplo:

```
> TABLE FILE SAMPLE  (DONDE "SAMPLE" ES EL NOMBRE
                       DEL ARCHIVO.)
```

O

```
> TABLE
> FILE
> SAMPLE
```

Para definir el contenido y el formato del reporte continue el estatuto TABLE y escriba una instrucción de petición o más, luego escriba la instrucción END, en una línea, solo esto es para finalizar este proceso. La estructura de la petición del reporte es la siguiente:

```
> TABLE FILE filename
            .
            .
            .
>       petición
            .
            .
            .
> END
```

Se puede utilizar el comando TABLE cuantas veces se requiera en un programa, siempre y cuando se finalize con la intrucción END. No se permiten TABLE'S anidados.

Para salir de FOCUS, es necesario indicar el comando FINISH ó su forma corta FIN en una sola línea.

Series de pedidos para el mismo archivo

Cuando varios pedidos se refieren al mismo archivo, el nombre de este archivo se puede establecer a asi no sera necesario repetir tantas veces el nombre del archivo, la sintaxis es:

SET FILE = FILENAME

EJÈMPLO: PEDIDOS PARA EL MISMO ARCHIVO

```
>> SET FILE = FIRST
>> TABLE
     :
     :
   REQUEST
     :
     :
>> END
>> TABLE
     :
     :
   REQUEST
     :
     :
>> END
>> FINISH
```

Los siguientes ejèmplos hacen una construcción desde el más simple pedido válido. Cada ejémplo agrega un nuevo elemento, cada uno de estos elementos se describen a detalle en las secciones de este manual. Estos elementos son: (1) verbos, objetos verbales, (2) clasificaciones, (3) selección, (4) cálulos, (5) coltrol de direcciones.

La estructura del pedido es lógica.

Un entendimiento general de cómo funciona, el lenguaje facilita su aplicación en una situación real y sería mas fácil para recordar.

Los ejèmplos usan un archivo simple llamado PROD contiene 7 campos de datos. Los nombres y relaciones entre éstos campos son la siguiente.

| CAMPOS | IDENTIFICACION |
|---|---|
| PROD-TYPE | Identidad del producto |
| AREA | Area geográfica |
| CUSTOMER | Nombre del cliente |
| MONTH | Mes desde el 1 hasta el 12 |
| UNITS | Cantidad de unidades embarcadas |
| AMOUNT | Valor (en dolares) de embarque |
| FACTOR | Factor de ajustación |
| BUDGET | Presupuesto de valores en dolares |

EJEMPLO: VERBO SINGULAR Y OBJETO VERBAL

```
TABLE FILE PROD
SUM UNITS
END
```

EL REPORTE APARECE COMO:

```
UNITS
-----
132560
```

EJEMPLO: OBJETO VERBAL SINGULAR "SORTEADO" POR LOS RENGLONES DE LA PAGINA.

```
TABLE FILE PROD
SUM UNITS
BY MONTH  -   FRASE DE SORTEAR POR LAS HOJAS
```

El reporte aparece como:

| MONTH | UNITS |
|---|---|
| 1 | 12345 |
| 2 | 11672 |
| 3 | 9764 |

EJEMPLO: SORTEAR LAS COLUMNAS DE LA PAGINA HORIZONTALMENTE.

```
>> TABLE FILE PROD
>> SUM UNITS
>> BY MONTH
>> ACROSS AREA  (FRASE DE SORTEAR LAS COLUMNAS DE LA PAGINA
                 HORIZONTALMENTE)
>> END
```

EL REPORTE APARECE :

| | AREA | | | |
|---|---|---|---|---|
| MONTH | EAST | NORTH | SOUTH | WEST |
| 1 | 6025 | 2015 | 3100 | 1205 |
| 2 | 5162 | 1408 | 2050 | 3052 |
| : | : | : | : | : |

ETC....

EJEMPLO:

CONDICION DE SELECCION REGISTROS

```
>> TABLE FILE PROD
>> SUM UNITS
>> BY MONTH
>> IF PROD-TYPE IS AXLES OR BEARING (FRASE DE SELECCION)
>> END
```

El reporte aparece como en el ejémplo pero solo se usan los datos para PROD_TYPE pedidos.

EJEMPLO:

DOS CAMPOS DE SORTEO MAS SUBTOTAL POR MES

```
>> TABLE FILE PROD
>> SUM UNITS
>> BY MONTH
>> BY AREA
>> ON MONTH SUB-TOTAL
>> END
```

EL REPORTE APARECE COMO:

| MONTH | AREA | UNITS |
|---|---|---|
| 1 | EAST | 6025 |
| | NORTH | 2015 |
| | SOUTH | 3200 |
| | WEST | 3052 |
| TOTAL MONTH 1 | | 14292 |
| 2 | EAST | 5162 |
| : | : | : |
| : | : | : |

ETC....

EJEMPLO:

CAMPO CALCULADO ATRAVES DE OTROS CAMPOS

```
>> TABLE FILE PROD
>> SUM UNITS AND AMOUNT AND COMPUTE
>> COMPOS-PRICE = AMOUNT/UNITS;
>> BY MONTH  BY AREA
>> ON MONTH SKIP-LINE
>> END
```

EL REPORTE APARECE COMO:

```
MONTH  AREA  UNITS  AMOUNT  COMPOS-PRICE

  1    EAST  6625   12410      2.46
       NOTH  2015    4163      2.08
       SOUTH 3200    6170      1.95
       WEST  3052    7240      2.43

  2    EAST   :       :         :
        :
        :

ETC.....
```

DEFINICION DE CALCULOS

Campos de información nuevos o temporales también se pueden definir, a parte de los campos de información reales en un archivo. Estos pueden ser combinaciones matemáticas o lógicas de campos de información reales o de otros campos temporales.

Los cálculos pueden definirse y ser usados de muchas maneras en una instrucción de petición. Independientemente del tipo de cálculo usado por el campo individual del registro de datos o de los datos acumulados de un reporte como datos de entrada, la sintáxis especificada es la misma. Esto se especifica en la sección "Cálculos de campos de datos temporales".

Los cálculos que se describen de antemano para su uso futuro de un requerimiento se utitliza el estatuto "DEFINE". Esto y el estatuto "TABLE" para especificar el criterio y disposición de reintegración del reporte, pueden ser mezclados libremente durante una sesión final.

EJEMPLO:

```
>> DEFINE FILE SAMPLE
>> NEWVALUE= expression;
      :
      :
>> END

>> TABLE FILE SAMPLE
        :
        :
     request
        :
        :
>> END
```

Se recomienda a los usuarios que primero dominen las ideas básicas del lenguaje de petición del reporte después procedan a los cálculos.

EJEMPLO:

```
Definiendo calculos en campos individuales.
>> DEFINE FILE PROD
>> NEWAMOUNT = IF FACTOR GT 1 THEN
>> AMOUNT*1.10 ELSE AMOUNT*FACTOR;
>> END

>> TABLE FILE PROD
>> "COMPARISON OF ACTUAL AND REVISED AMOUNT"
>> SUM AMOUNT AND NEWAMOUNT AND COMPUTE
>> DIFERENCE = AMOUNT-NEWAMOUNT ;
>> BY MONTH
>> END
```

EL REPORTE APARECE COMO :

COMPARISON OF ACTUAL AND REVISED AMOUNT

| MONTH | AMOUNT | NEWAMOUNT | DIFFERENCE |
|---|---|---|---|
| 1 | 31642 | 33479 | - 1837 |
| 2 | 38741 | 39417 | - 676 |
| 3 | 45234 | 49648 | - 4414 |

RESUMEN :

Hay cinco tipos de elementos en un estatuto de petición.

- Verbos y objetos verbales.
- Cálculos sobre los objetos verbales.
- Separación por géneros en hileras o através de columnas.
- Condiciones de selección .
- Direcciones de control.

Un estatuto tiene que contener una frase verbal compuesta por un verbo o un objeto verbal. Todos los demás elementos son opcionales.

El orden de presentación de los elementos es arbitrario, ellos pueden ser escritos en al misma línea o en diferentes líneas, las siguientes instrucciones son equivalentes.

```
>> SUM UNITS
>> BY MONTH

>> SUM UNITS BY MONTH

>> BY MONTH SUM UNITS

>> SUM UNITS BY
>> MONTH
```

El reporte pedido trata con nombres de campos. No tiene importancia si el campo está en el archivo, si proviene de otro archivo (por medio de segmento virtual), si es un valor computado descrito en un diccionario permanente o si es un campo definido temporalmente.

RECOMENDACIONES :

* Organizar un estatuto de petición con los cinco tipos de elementos y escribirlos en el orden dado en el resumen. Esto es, primero la frase verbal, cálculos, sortear las frases, condiciones de selección y por último las direcciones de control. Colocar los diferentes tipos de elementos en líneas separadas.

* No junte muchas palabras en una línea, se pueden usar todas las líneas que sean necesarias, no hay ningún límite de líneas en un mandato. Un mandato termina cuando se escribe la palabra "END" en una línea.

* Use un "ALIAS" corto para referirse a un campo en lugar del nombre completo del campo. No cambie los "ALIAS" usando espacios en blanco entremezclados en el nombre o con caractères especiales. Esto lo hace más fácil de escribir y evita errores de ortografía y de escritura.

CONVENCIONALISMO DE UN ESTATUTO

Espacios entre palabras.

Se pueden escribir una o más palabras en cada línea de las intrucciones dadas. Cada palabra es separada de la siguiente por uno o más espacios. Algunas veces una palabra contiene un espacio en blanco, en este caso el grupo de palabras que debe de ser interpretado como un solo elemento se debe describir entre comillas sencillas. Por ejemplo, en la frase,

IF PRODUCT IS 'STEEL FLANGE'

Hay un espacio en blanco después de la palabra STEEL, por lo tanto dos palabras son unidas entre las comillas.

Nombres de Campos de Datos.

Un archivo FOCUS esta compuesto de elmentos de datos básicos,cada uno con un nombre de campo ya proporcionado o un ALIAS más corto. Los nombres de estos campos de datos son usados de maneras muy variadas en los estatutos. Ya que se mencionan con cierta frecuencia, se usa una regla especial para verificar la validez del nombre del campo. Esta regla le permite al campo de datos el ser referido como la secuencia más corta de caracteres el cual le permite emparejarse de manera única a un nombre válido o a un ALIAS que ha sido descrito en un diccionario central. Esta puede ahorrar errores y mucha escritura.

EJEMPLO:

Referencia a Nombres de Campo

El ARCHIVO MAESTRO contiene:

| FIELD | ALIAS |
|---|---|
| AMOUNT | AMT |
| AREA | ARE |
| QUANTITY | SIZE |

| TYPED FIELD | RESULTS |
|---|---|
| AM | Referencia al campo AMOUNT |
| AR | Referencia al campo AREA |
| Q | Referencia al campo QUANTITY |
| A | Error, no es única |
| SIZE | Referencia al campo QUANTITY |

Desplegados de Nombres de Campos

En cualquier punto de proceso del estatuto, si se introduce una línea ?F sólo, aparecera la lista de los nombres de campo. Esto es de gran utilidad si se ha olvidado escribir alguno o si se necesita hacer referencia a uno sin tener la necesidad de salirse del proceso.

Corrección de Errores

El FOCUS está conciente de un estatuto dado, y cuando un error de escritura, o un error de ortografía o cualquier otra razón ocasiona que una interpretación de una intrucción se interrumpa, la palabara no reconocida aparecera y se dara la oportunidad para reinscribir esa palabra. Si no se desea continuar con la instrucción, se puede escribir la palabra QUIT como respuesta y se cancelara la instrucción.

EJEMPLO:

Corrección de error en un reporte dado.

```
>> TABLE FILE PROD
>> SUM AMOUNT AND UNITTS BY MONTH

(FOC002) FIELD NAME IS NOT IN THE DICTIONARY: UNITTS
         REPLY: UNITS
>> IF PROD_TYPE IS BEARINGS
>> END
```

El número de errores a la izquierda del mensaje diagnosticado intenta clasificar el error. Una lista de estos errores se encontrará en la sección "Mensajes de Error del Estatuto del Reporte".

Cuando el estatuto no se pide o introduce en el "momento", entonces el error diagnosticado es el mismo pero no se da la oportunidad de corregir de inmediato. En cambio, el estatuto sigue leyendose hasta que se encuentre la línea antes almacenada END aparezca. El proceso del estatuto será cancelado cuando exista un error en el estatuto antes almacenado.

Cuando se esté desarrollando un reporte dado es conveniente grabar todas las líneas escritas para después revisarlas o para editar el resultado para uso futuro.

Si, en respuesta al "REPLY" se introduce un signo de "?" (en lugar de una palabra o frase), aparecerá sobre la línea el texto completo del mensaje equivocado.

Si se responde ?F aparecerá la lista de los nombres de campo.

EL ESTATUTO HELP

Están disponibles una serie de archivos conteniendo textos sobre varios tópicos. Si se introduce el estatuto HELP, aparecerá un menú completo de tópicos de texto archivados. Si se introduce el estatuto HELP seguido de un tema, aparecerá el archivo de ese tema. Por ejémplo, HELP TABLE, resume la información sobre el tema TABLE. Los usuarios pueden aumentar sus propios archivos sólo necesitan el archivo .ERR.

INDICACION DEL ESTATUTO

Cuando FOCUS está listo para recibir un estatuto aparecerá un signo ">" en la teminal del operador. En este momento puede escribirse el estatuto y comenzará de inmediato el procesamiento. En cualquier momento, si se introduce el retorno del carril (RETURN), el FOCUS repetira el procesamiento, Por Ejemplo: TABLE, DEFINE, ETC....

TERMINANDO O CONTINUANDO CON EL ESTATUTO "TABLE"

Se puede terminar una instrucción de un reporte escribiendo una de tres palabras solas sobre una línea.

Estas son:

| | |
|---|---|
| END | El final normal de un procedimient, despué de imprimir la palabra END, comenzará la preparación y respuesta del reporte. Después de imprimirse el reporte, se estará en el nivel de estatuto del FOCUS en donde se puede imprimir un estatuto nuevo. |

QUIT — Esto concluirá el procedimiento del reporte actual y regresará al operador de inmediato al nivel de comando del FOCUS.

RUN — Este es un final normal del procedimiento actual, pero después de que se haya impreso el reporte se quedará en TABLE y seguirá activo el mismo archivo de datos. Por lo tanto el estatuto TABLE y el nombre del archivo no necesariamente tienen que ser re-editado. Esto es muy conveniente cuando se esté en un diálogo interactivo usando la base de datos y una variedad de reportes pedidos.

EJEMPLO:

TERMINANDO UN REPORTE CON "RUN"

```
>> TABLE FILE PROD
>>         :
>>         :
T>      REPORTE
>>         :
>>         :
>> RUN
T>      REPORTE
>>         :
>>         :
>> END
>>
```

Nótese que una indicación de T> se imprime cuando en FOCUS está listo para recibir otra intrucción. La letra "T" con el signo ">" sirve para recordar que el estatuto TABLE está todavía activo. De igual forma con esto se recuerda de esto cuando se oprime <return>

CONVENSIONES DE IMPRESION

Los reportes que aparecen en la pantalla son divididos en páginas de 24 líneas por 160 columnas, ya que la mayoría de las pantallas sólo pueden mostrar 80 columnas en un mismo tiempo, un juego de teclas permiten mover la impresión de izquierda a derecha y de arriba hacia abajo.

Cuando un reporte se excede de las 80 columnas, un mensaje informará del ancho total, de otra manera será menor de 80 columnas. Se moverá de página a página oprimiendo las teclas F7 y F8 de izquierda a derecha oprimiendo F9 y F10. La tecla ENTER moverá rápidamente hacia adelante y el operador deberá oprimir esta tecla en la última página para señalar que se ha terminado de ver el reporte. ver el PC/FOCUS "GUIA DE OPERACIONES" para cualquier información adicional sobre el control de la pantalla.

OFFLINE, ONLINE

A menos de que se mande otra cosa, el reporte aparecerá escrito sobre la pantalla. Esta designación puede cambiarse en donde puede dirigirse el reporte a un impresor interconectado. Antes de pedirle el reporte hay que meter el estatuto OFFLINE de FOCUS. De esta manera, todos los reportes serán dirigidos al dispositivo que ha sido identificado con OFFLINE. La omisión asociada será la impresora. La terminal de comunicación puede convertirse de nuevo en la impresión destinada, registrando el estatuto opuesto ONLINE.

EJEMPLO:

Reporte impreso en la pantalla e impresora

```
>> TABLE FILE PROD
>>       :
>>       :
>>    REPORTE
>> END
>> OFFLINE          Aquí el reporte está dirigido a una
>> RETYPE           impresora determinada.
>> ONLINE           El reporte aparecerá en la pantalla.
```

Si se dispone de una impresora más amplia, se tendrá que hacer uso del comando MS-DOS LPT1:130 antes de registrar el PC/FOCUS y así cambiar la impresora a una de 130 caractéres de ancho.

Generalmente todos los reportes de OFFLINE van al mismo archivo de salida, éste archivo es asignado automáticamente cuando se registra el FOCUS y en casi todos los casos ésta será la impresora. Una vez registrada en FOCUS se puede pedir el estatuto OFFLINE CLOSE lo cual cierra cualquier archivo actual que esta esperando a ser impreso.

COPIAS MULTIPLES- EL ESTATUTO "RETYPE"

Inmediatamente después de haber impreso un reporte, se puede obtener una copia registrando el estatuto RETYPE.

EJEMPLO:

COPIAS MULTIPLES

```
>> TABLE FILE PROD
>>      :
>>      :
>>    REPORTE
>>      :
>> END
>> RETYPE
>> RETYPE
```

REPORTES AMPLIOS SOBRE LA PANTALLA.- PANELES

La mayoría de las panatallas tienen 80 caractéres, los reportes que tengan muchas columnas o se excedan de 80, pueden ser escritas en páneles. El PC/FOCUS usa un total de 160 carácteres como normal. Por ejemplo, los primeros 160 caracteres de la página 1, etiquetada página 1.1 se escibirán por los próximos 160 carácteres en la próxima página, la cual será etiquetada como página 1.2, la página 2 será etiquetada como página 2.1 etc...El estatuto que indica lo anterior es el siguiente:

```
>> SET PANEL=N
```

Donde N es un número entero que está entre 12 y 0

ELEMENTOS DEL LENGUAJE DE PETICION

* VERBOS

* CLASIFICACION

* SELECCION

VERBOS

Un verbo es una palabra de acción. Estas acciones se llevan a cabo en los campos, los cuales han sido asignados como los objetos de los verbos. La lista de verbos es:

| VERBO | SIGNIFICADO |
|---|---|
| LIST | Escribe una lista de registros en una línea y los enumera. |
| PRINT | Escribe una lista de registros en una línea pero no los enumera. |
| COUNT | Cuenta el número de frecuencias. |
| SUM | Escribe el registro sobre una línea el cual puede contener otros campos computacionales de registros adicionales, sobreescribiendo campos alfanuméricos. |
| ADD | Lo mismo que SUM. |
| WRITE | Lo mismo que SUM. |

La sintáxis para una frase verbal sencilla es:

VERB Nombre de Campo AND Nombre de Campo AND Nombre de Campo.

La conjunción AND es un conector opcional entre los objetos verbales que mejoran la legibilidad. Se pueden nombrar hasta 64 campos de esta manera. La palabra THE puede usarse opcionalmente adelante de un nombre de campo y ayudaría a mejorar la legibilidad de la frase. Ya sea que se use el nombre del campo completo o su ALIAS, cualquiera se puede utilizar para identificar un campo de datos o su corta truncación única.

EJEMPLO:

Frases con Verbos Simples

```
>> SUM AMOUNT
>> SUM AMOUNT AND UNITS AND FACTOR
>> LIST THE PROD_TYPE AND AREA
>> PRINT PROD_TYPE AND MONTH AND AREA AND UNITS
>> SUM AMOUNT BUDGET UNITS
>> COUNT PROD_TYPE
```

ORDEN DE APARICION

El nombre con que se proveen los nombres de campo en las frases verbales es el orden en el cual se imprimen las columnas del reporte.

EJEMPLO:

Orden de Campos en una Frase Verbal

```
>> TABLE FILE PROD
>> LIST PROD_TYPE AND UNITS AND AMOUNT
>> IF MONTH IS 1 IF AREA IS EAST OR WEST
>> END
```

El reporte aparece como:

```
LIST    PROD_TYPE   UNITS     AMOUNT
----    ---------   -----     ------

 1      AXLES       150       3062.45
 2      BEARINGS    324       4189.74
 :
 :
ETC....
```

Si la orden se cambió de tal manera que la frase verbal se:

```
LIST UNITS AND PROD-TYPE AND AMOUNT
```

entonces las columnas del reporte aparecerá así:

```
LIST      UNITS       PROD-TYPE          AMOUNT
-----     -----       ---------          ------
 1        150         EJES               3062.45
 2        354         SOPORTES           4189.74
 .
 .
etc...
```

PRINT Y LIST

Los verbos PRINT y LIST le dan un valor sencillo a cada uno de los campos de datos mencionados para aparecer en una línea. En un sentido, los registros de datos se enlistan. El verbo LIST le agrega un número serial a cada línea del reporte. En ocasiones, ésto de gran utilidad ya que indica claramente que cada línea representa un registro del archivo de datos y no que son una acumulción de registros.

En donde haya un o más pausas divisionales en un reporte con una LIST, las líneas serán re-enumeradas cada vez que el campo de selección mayor cambie de valor.

EJEMPLO:

El verbo LIST

```
>> TABLE FILE PROD
>> LIST PROD-TYPE AND AMOUNT BY AREA BY MONTH
>> IF MONTH FROM 1 TO 3
>> END
```

El reporte aparece como:

| AREA | MONTH | LIST | PROD-TYPE | AMOUNT |
|---|---|---|---|---|
| EAST | 1 | 1 | AXLES | 1241.23 |
| | | 2 | BEARINGS | 1318.05 |
| | 2 | 3 | AXLES | 1614.42 |
| WEST | 1 | 1 | AXLES | 1443.00 |
| | | 2 | BUSHINGS | 245.37 |

Si la palabra PRINT ha sido sustituida por la palabra LIST, la columna etiquetada con LIST será omitida del reporte impreso.

EJEMPLO:

El verbo PRINT

```
>> TABLE FILE PROD
>> PRINT PROD-TYPE AND AMOUNT
>> BY AREA BY MONTH IF MONTH FROM 1 TO 3
>> END
```

El reporte aparecerá igual que el anterior pero ahora sin la columna LIST donde enumera el número de registros sorteados por algun campo.

SUM

Cuando se pide el total de los valores de los campos de datos entonces, se usa el verbo SUM. Por ejemplo, SUM UNITS BY MONTH, significa que todos los valores de las UNITS de campo deberán de agregarse a sumarse juntos para cada mes.

NUMERO DE REGISTROS = 26000  LINEAS= 12

Notese que los valores de los 26000 registros se han acumulado en 12 líneas divididas en el reporte escrito.

EJEMPLO:

El verbo SUM

```
>> TABLE FILE PROD
>> SUM UNITS
>> BY MONTH
>> END
```

El reporte aparecerá como:

```
MONTH          UNITS
-----          -----
  1            18000
  2            14625
  3            10843
  .
  .
  .
ETC......
```

CAMPOS DE DATOS NUMERICOS

Solo los campos de datos numéricos pueden sumarse juntos si un campo no numérico es el objeto del verbo, como SUM CUSTOMER, entonces esto no se rechazar como un error, sino el valor del último registro pedido sera impreso. En algunas situaciones, esto es de gran utilidad, particularmente cuando hay muchos campos de objetos verbales.

OBJETOS VERBALES MULTIPLES

Muchos campos de datos pueden ser nombrados simultaneamente como objetos del verbo. En consecuencia

SUM UNITS AND AMOUNT AND NEW-QTY BY AREA

Se refiere a tres campos de datos en la porción de la instrucción. Aparecerá una columna para cada una de los reportes escritos en el orden en que sean mencionados los campos y los valores en cada columna son los totales para cada campo. De esta menera el reporte aparecerá así:

| AREA | UNITS | AMOUNT | NEW-QTY |
|---|---|---|---|
| EAST | 10000 | 26000.00 | 28000.00 |
| NORTH | 8000 | 19000.00 | 26000.00 |

.
.
.
ETC.....

Se puden nombrar como objetos del verbo hasta 64 campos de datos en una intrucción.

SINONIMOS DE SUM, WRITE Y ADD

El verbo WRITE tiene un efecto idéntico al verbo SUM (ó ADD). Su uso es muy conveniente para transmitir el intento de una orden en casos en donde los valores ocurran una sola vez y por lo tanto no es acumulado, a diferencia de otros valores que la misma orden si se acumulan. Esto dependerá de las condiciones de sorteo. Pueden señalar de manera única un registro para cada linea escrita. Por ejemplo: WRITE PRODUCT BY AREA BY CUSTOMER BY MONTH.

Esta sintáxis es mejor que la de SUM PRODUCT ya que los datos para PRODUCT son valores no-numéricos.

COUNT

Se puede obtener el conteo del número de frecuencia de algunos campos de datos o de algunas condiciones usando el verbo COUNT.

CONTEO DEL CAMPO

El campo de datos en el cual se contará su número de frecuencias, es mencionado como el objeto del verbo COUNT. Esto requiere con fracuencia algun conocimiento de la manera de como ha sido solo estucturado el archivo. Por ejemplo, si el archivo PROD ha sido estructurado:

```
AREA------>CLIENTE----> PRODUCTO
                        MES
                        UNIDAD
                        CANTIDAD
                        FACTOR
```

Entonces un conteo del número de clientes de cada area se pide así:

```
>> COUNT CUSTOMER  BY AREA
```

El reporte aparecerá como:

| AREA | CONTEO/CLIENTE |
|---|---|
| EAST | 248 |
| NORTH | 172 |
| SOUTH | 89 |
| WEST | 102 |

Nótese que el título o encabezado sobre la columna contada contiene la palabra COUNT bajo el nombre del campo que está siendo contado.

EJEMPLO:

Contar la cantidad de numeros de productos ordenados por area y cliente entonces el reporte puede ser:

```
>> COUNT PRODUCT
>> BY AREA BY CUSTOMER
```

SUMANDO Y CONTANDO

Una suma de los valores de los campos y el número de registros usados para la obtención de la suma puede obtenerse pidiendo que se ejecuten juntos SUM y COUNT.

EJEMPLO:

Sumando y Contando

```
>> TABLE FILE PROD
>> SUM AMOUNT AND COUNT
>> BY AREA
>> IF PRODUCT IS BEARINGS AND GEARS
>> IF UNITS FROM 10 TO 500
>> END
```

El reporte aparece como:

| AREA | AMOUNT | COUNT |
|---|---|---|
| EAST | 19000.00 | 48 |
| NORTH | 16500.00 | 39 |
| SOUTH | 4500.00 | 20 |
| WEST | 10800.00 | 28 |

Nótese el uso de las palabras AND COUNT en el mismo enunciado.

LOS PREFIJOS CNT. Y SUM.

La identidad de cuales campos contar y cales acumular pueden esclareserse cuando son mezclados usando los prefijos "CNT" para contar y acumular otros con "SUM".

WRITE SUM.SALES AND CNT.ORDERS AND SUM.AMOUNT AND CNT.PRODUCT

Los encabezados sobre las columnas del reporte resultante aprecerá así:

SALES ORDERS/COUNT AMOUNT PRODUCT/COUNT

CONTANDO LAS ENTRADAS

Cuando se necesita sólo el número de frecuencias de cada campo de control seleccionado, se usa la frase COUNT ENTRIES. Cada vez que se extracte un registro de campos de datos, la línea donde aparecerá sera aumentada.

COUNT ENTRIES BY PRODUCT

Indicará la cantidad de veces en que se mencinará cada producto en el archivo. Esto es más legible que el enunciado COUNT PROD BY PROD

Nótese la palabra ENTRIES. Preserva la sintáxis del objeto verbal después de un verbo.

Esto es igual que el estatuto

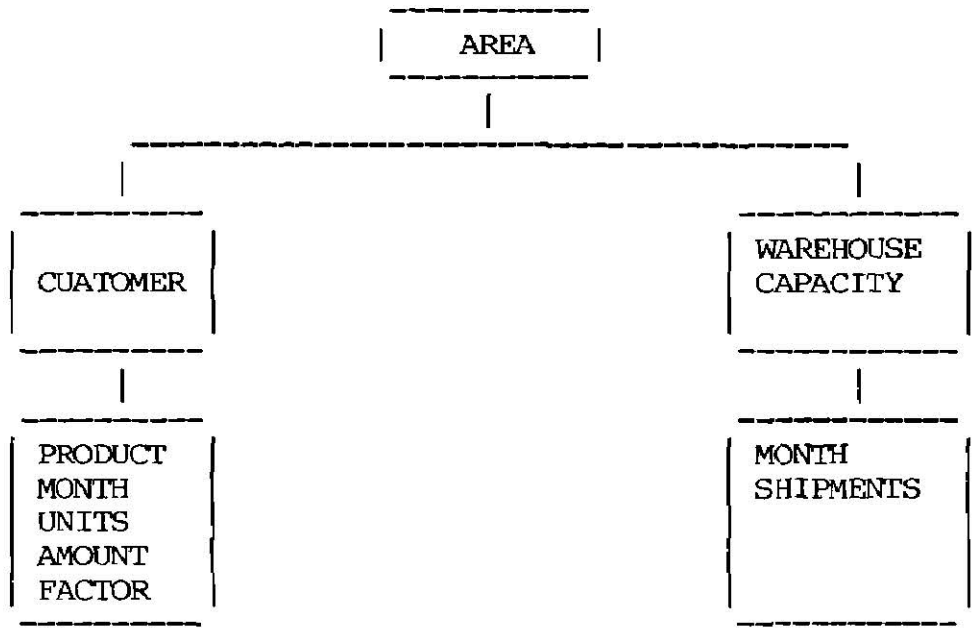
COUNT PROD BY PROD

El reporte aparecerá como:

| PRODUCT | PRODUCT/COUNT |
|---|---|
| AXLES | 105 |
| BEARINGS | 78 |
| . | |
| . | |
| . | |
| ETC..... | |

CONTANDO EN UN ARCHIVO MULTIPLE

Considerese la siguiente estructura del archivo en donde hay un número de variable de bodegas en cada area, en adición a los clientes en el área. Esquemáticamente el archivo aparecerá:




El enunciado es:

COUNT CUSTOMER AND WAREHOUSE BY AREA

aparecerá asi:

| AREA | CUSTOMER/COUNT | WAREHOUSE/COUNT |
|---|---|---|
| EAST | 248 | 6 |
| NORTH | 172 | 3 |
| SOUTH | 89 | |
| WEST | 102 | 3 |

Nótese que no hay entradas para bodegas en el área sur. Esto refleja que no se encontraron registros en el archivo de bodegas para esta área. Esto es precisamente lo que el enunciado de mandato solicita cuando los datos son archivados de esta manera.

Un tipo de solicitud mas involucrada combinando conteo, selección y juegos múltiples se ilustra en seguida.

EJEMPLO:

Contando en dos grupos con condiciones de sorteo.

```
>> TABLE FILE PROD
>> SUM AMOUNT AND COUNT
>> BY AREA
>> SUM SHIPMENTS AND COUNT
>> BY AREA  BY WAREHOUSE
>> END
```

El reporte aparece como:

| AREA | AMOUNT | COUNT | WAREHOUSE | SHIPMENTS | COUNT |
|---|---|---|---|---|---|
| EAST | 347943.00 | 1228 | NEWARK | 6492 | 10 |
| | | | PHILA | 3761 | 8 |

UTILIZANDO EL "COUNT" EN UN REPORTE MATRIZ

Contando en un reporte matriz se puede obtener combinando el conteo con la intrucción de seleccionar un campo a travéz de las columnas de la página. En este caso las columnas son mostradas a lo largo de algunas variables de interés en adición a los campos de selección controlando las hileras del reporte. Nótese el uso de la frase ACROSS MONTH en el siguiente ejemplo, y del efecto de la intrucción "COLUMN-TOTAL".

EJEMPLO:

```
>> TABLE FILE PROD
>> "DISTRIBUTION OF PRODUCTS SOLD EACH MONTH"
>> "IN EACH REGION"
>> COUNT PRODUCT AND COLUMN-TOTAL BY REGION
>> ACROSS MONTH
>> END
```

El reporte aparece como:

```
DISTRIBUTION OF PRODUCTS SOLD EACH MONTH
IN EACH REGION

          MONTH
          1   2   3   4   5   6   7   8   9   10  11  12

REGION
---------------------------------------------------------

EAST      20  10  15  16  19  30  40  50  51  40  31  20
NORTH     14  20  21  26  28  29  42  30  21  16   4   5
SOUTH     18  10  14  19  19  18  17  16   1   4   9   7
WEST      15   8   7   4   9  10  11  10   3   4   2   4

TOTAL     68  48  57  65  75  87 110 106  76  64  46  36
```

TOTALES DE COLUMNAS Y RENGLONES

Los totales de hileras se pueden obtener con reportes de tipoe de matriz agregando la palabra AND ROW-TOTAL junto con la de AND COLUMN-TOTAL

EJEMPLO:

```
>> TABLE FILE PROD
>> COUNT ENTRIES AND ROW-TOTAL AND COLUMN-TOTAL
>> BY PRODUCT ACROSS MONTH IF MONTH NOT-FROM 4 TO 9
>> END
```

UNA MANERA SIMPLE DE REFERIRSE A TODOS LOS CAMPOS

Un rasgo muy usual le permite a todos los campos de datos en un segmento el ser referidos sin tener que nombrarlos explícitamente en un procedimiento. Sólo se necesita saber un solo nombre de campo y el prefijo SEG. junto a él para activar a éstas especificaciones. Por ejemplo, si el segmento se describe asi:

```
SEGMENT = DATREC
FIELD   = MONTH
FIELD   = UNITS
FIELD   = AMOUNT
FIELD   = FACTOR
```

El procedimiento es:

```
PRINT SEG.MONTH
```

Es equivalente a:

```
PRINT MONTH AND UNITS AND AMOUNT AND FACTOR
```

Esta característica puede ser mas que una ayuda conveniente. Por ejemplo, un procedimiento general podría mostrar el contenido completo de un segmento aún y cuando se suprima cambios posteriores al archivo.

La sintáxis tiene dos formas:

```
SEG.FIELDNAME
```

```
SEGMENT FIELDNAME
```

ENUNCIADOS MULTIPLES, JUEGOS DE VERBOS

Un enunciado generalmente consiste de una frase verbal sencilla y de sus condiciones de selección asociados. Un enunciado FOCUS puede consistir de un grupo de frases verbales separadas cada una con sus propias condiciones de selección. Para mostrar toda la información, tiene que existir una relación significativa entre los grupos de condiciones de selección separadas. La regla utilizada es que cada grupo tiene que contener cuando menos los mismos campos de control de selección que los de su predecesor y puede adjuntar muchos otros.

EJEMPLO:

Grupo de verbos múltiples

```
>> TABLE FILE PROD
>> SUM AMOUNT AS 'TOTAL,AMOUNT' BY MONTH
>> SUM AMOUNT AND UNITS
>> BY MONTH BY CUSTOMER
>> PRINT ESTIMATE BY MONTH BY CUSTOMER
>> BY PRODUCT
>> IF AREA IS EAST
>> END
```

El reporte aparece como:

| MONTH | TOTAL AMOUNT | CUSTOMER | AMOUNT | UNITS | PRODUCT | ESTIMATE |
|---|---|---|---|---|---|---|
| 1 | 26000 | AJAX CO. | 13000 | 800 | AXLES | 6500 |
| | | BUSH MFG. | 10000 | 600 | GEARS | 4000 |
| | | | | | SCREENS | 6000 |
| | | CAP BROS. | 3000 | 250 | AXLES | 1800 |

El uso de los múltiples grupos de objetos es de gran utilidad en casos en donde las estructuras de los archivos estén compuestos de diversos tipos de segmentos de datos de números de ocurrencias desiguales. Contando los diferentes números de ocurrencias pueden ser ambigüas, a menos de que se proveen de un enunciado muy claro de qué es lo que se quiera contar.

Nota: El encabezado sobre el primer campo de datos AMOUNT se ha cambiado en el ejemplo para que sea diferente de la segunda ocurrencia en el segundo procedimiento.

Los múltiples grupos de verbos y de condiciones de selección en el mismo enunciado permiten:

1.- Un reporte que sumarize varios niveles.
2.- Un valor ha ser calculado, el cual es el porcentaje de un subtotal.
3.- Una procedimiento para acumular valores con facilidad o para contar valores en diferentes niveles de un archivo.

Pueden estar presentes seis verbos y sus condiciones de selección asociados. En este caso el primer grupo de objetos verbales se comportan como totales generales.

Las reglas para las condiciones de selección cuando están presentes mas de un grupo de condiciones de selección son:

1.- Cada grupo de condiciones de selección tienen que usar los campos previamente nombrados como primer grupo de campos de selección.

2.- Cada grupo de condiciones de selección puede agregarle al grupo previo, nuevos campos de selección.

3.- Solamente el último grupo de selección puede usar las intrucciones PRINT ó LIST.

4.- La condición IF se aplica a los registros seleccionados para el reporte completo.

EJEMPLO:

Mostrando un porcentaje de un sub-total.

```
>> TABLE FILE PROD
>> SUM UNITS  AS  'TOTAL, SOLD, IN AREA'
>> BY AREA
>> PRINT UNITS AND COMPUTE
>> PA=100*C2/C1; AS 'PERCENT, AREA'
>> BY AREA BY PROD-TYPE
>> ON AREA SUB-TOTAL
>> END
```

El reporte aparece como:

| AREA | TOTAL SOLD IN AREA | PROD-TYPE | UNITS | PERCENT AREA |
|---|---|---|---|---|
| EAST | 3600 | AXLES | 1000 | 27.78 |
| | | BEARINGS | 1200 | 33.33 |
| * TOTAL | 3600 | | 2200 | 61.11 |

WEST
.
.
.
etc.....

Nota:

1.- La notación de columna, C1, C2, etc... se usa para referirse a los objetos verbales, porque los dos tienen el mismo nombre en este ejemplo.

2.- Un porcentaje del total de la columna puede obtenerse directamente prefijando el campo del objeto verbal con la palabra PCT.

3.- La frase "AS" se usa para proveer un encabezado a la columna el cual es diferente del nombre del campo.

EJEMPLO:

CONT y SUM en procedimiento múltiple.

```
>> TABLE FILE PROD
>> COUNT PROD-TYPE BY AREA
>> SUM AMOUNT
>> BY AREA BY MONTH FROM 1 TO 3
>> ON ARE SKIP-LINE
>> END
```

El reporte aparece como:

| AREA | PROD-TYPE/COUNT | MONTH | AMOUNT |
|---|---|---|---|
| EAST | 20 | 1 | 1500 |
| | | 2 | 1200 |
| NORTH | 18 | 1 | 1200 |

Operaciones realizadas directamente sobre los objetos verbales

Cuando se pide un registro de un archivo, se puede usar uno o más de sus campos de datos para buscar la línea adecuada del reporte para colocar el registro. Si muchos registros ocupan la misma línea de salida debido a la naturaleza de la condición de selección, entonces la falla será la de agregar los campos de datos computacionales juntos, por ejemplo si el enunciado es la de WRITE UNITS BY MONTH, aparecerá 12 líneas escritas, una por mes, se pueden acumular muchos registros sobre cada línea.

En lugar de acumular los valores el mayor de todos los valores pueden haber sido usados, ó el número, etc... Hay 12 operaciones directas de este tipo que pueden ser especificados. La sintáxis es la de agregarle un prefijo al nombre del campo de datos para que sea procesado. Por ejemplo, WRITE MAX.UNITS BY MONTH (Escribir el número máximo de unidades por mes).

Los prefijos de operación directos son:

| PREFIJOS | SIGNIFICADO |
|---|---|
| MAX | Selecciona el valor máximo. |
| MIN | Selecciona el valor mínimo. |
| AVE | Computa el valor promedio (x/n). |
| ASQ | Computa la suma promedio de cuadrados. |
| FST | Selecciona el primer valor pedido. |
| LST | Selecciona el último valor pedido. |
| PCT | Computa el porcentaje de la columna total. |
| RPCT | Porcentaje de la hilera. |
| TOT | Computa la columna total. |
| SUM | Suma de las columnas. |

Las operaciones directas solo pueden ejecutarse en los campos mencionados en la frase verbal. Si la selección se realiza en el valor promedio, por ejemplo, entonces los valores promedios se tienen que obtener primero y el reporte sostenido en un archivo de HOLD. La selección final se realiza en el archivo HOLD.

EJEMPLO:

Sorteo por porcentajes en forma descendente

```
>> TABLE FILE PROD
>> SUM PCT.AMOUNT AND HOLD
>> BY PROD_TYPE HOLD
>> END
>> TABLE FILE HOLD
>> PRINT PROD_TYPE
>> BY HIGHEST AMOUNT
>> END
```

El reporte aparece como

| AMOUNT | PROD_TYPE |
|---|---|
| 16.40 | GEARS |
| 12.05 | BEARINGS |
| 9.21 | AXLES |
| 7.75 | SLEEVES |
| ETC..... | |

Nótese que la palabra HIGHEST al frente del nombre del campo, cambia la secuencia de selección de HIGH-TO-LOW para que el valor más alto se imprima primero. Ej. HIGHEST AMOUNT

DATOS FALTANTES - PREFIJO (ALL)

Un enunciado de petición nombra los datos a revelar. La implicación es que todos los campos de datos nombrados tienen que por lo menos estar presentes. Por ejémplo imprimir Estado y Ciudad, implica que ambos tienen que estar presentes para que el proceso de selección realize. Exluye casos en el que el Estado este presente pero no la ciudad. ETC..

Para incluir, por ejemplo todos los Estados en el reporte, aún para aquellos en donde no se registro ningun dato Ciudad, se agrega el prefijo ALL, al nombre del campo. Entonces la frase:

```
    PRINT ALL.STATE AND CITY
o la frase
    PRINT CITY BY ALL.STATE
```

Incluirá los registros cortos en el reporte. En el reporte escrito aparecerá un punto '.' en lugar del dato que no está disponible. Nótese que todos los datos faltantes nulifican las condiciones que puedan ser aplicadas a él.

SELECCION DE REGISTROS

* CONDICIONES DE SELECCION
* PRUEBAS EN VALORES TOTALES
* LEYENDO LAS PRUEBAS DE UN ARCHIVO

CRITERIO DE SELECCION

Seleccionando los registros

La ausencia de una condición de selección en el enunciado de petición, ocasiona que cada registro en el archivo de datos sea considerado aceptable y seleccionado. Se puede usar cualquier número de frase de selección y puede referirse a cualquier campo de datos en el archivo. (ya sean campos reales o verdaderos, calculados, etc..) la sintáxis es:

IF nombre de campo RELATION literal [or literal or literal...]
IF etc...

La frecuencia particular de un campo se considera aceptable si su valor obedece a la RELATION (relaciones) a cualquiera de las literales (valores ) indicados. Las relaciones son:

| RELACIONES | SIGNIFICADO |
|---|---|
| IS, EQ | Igualdad entre el valor del campo o literal |
| IS-NOT, NE | Desigualdad entre los valores del campo y literal. |
| IS-FROM,FROM,GE | Valor del campo = ó > que la literal. |
| TO, LE | Valor del campo = ó < que la literal. |
| IS-MORE-THAN | Valor del campo mayor que la literal. |
| IS-LESS-THAN,LT | Valor del campo menor que el literal. |
| FROM TO | Valor del campo en rango. |
| NOT-FROM TO | Valor del campo no en rango. |
| CONTAINS | Caracteres en valor del campo contiene caracteres en la literal. |
| INCLUDES | En una cadena de campo de valores todas las literales estan presentes. |
| EXCLUDES | En una cadena de campos de valores todas las literales no estan presentes. |
| OMITS | Los caracteres en un valor del campo no contiene caracteres en la literal. Contrario de CONTAINS. |

Probando Valores Acumulados

IF TOTAL

Las condiciones de selección son usadas en un enunciado de petición para limitar la aceptación de registros individuales a aquellos que pasan las pruebas. Estos son los registros que se sortean, acumulan, y procesan en el formato de salida del reporte terminado. Un requerimiento de tipo común de selección secundaria es el de eliminar 'líneas' del reporte terminado, cuyo total para un campo específico no pasa la condición de prueba. Este tipo de selección es válida solo si hay un proceso de acumulación en los campos en el reporte. Esto es, si el verbo es SUM, WRITE o COUNT; y no si el verbo es LIST o PRINT. Tanto las selecciones registros individuales como las selecciones de valores totales, pueden usarse en las mismas demandas.

Una selección en el total de salida de un campo de datos se realiza colocando la palabra TOTAL antes del nombre del campo en una frase IF. La sintáxis para una selección en un total es:

IF TOTAL nombre del campo RELACION LITERAL o LITERAL....etc..

El nombre del campo en el cual se realiza la prueba tiene que ser uno de los campos del objeto verbal ya que estos son los campos que han sido acumulados. Una operación directa como AVE (PROMEDIO) o MAX (MAXIMO) puede ejecutarse en el campo y aplicarse la prueba TOTAL al valor resultante.

EJEMPLO: Acumulando totales

```
>> TABLE FILE PROD
>> SUM UNITS AND AMOUNT
>> BY PRODUCT BY MONTH
>> IF AREA IS EAST OR  WEST
>> IF TOTAL UNITS UNITS EXCEEDS 1000
>> END
```

Selección TOTAL con ACROSS

Cuando se produce un reporte matriz usando la palabra ACROSS para desplegar el campo del objeto verbal através de la página, entonces cada celda o intersección de hileras (fila) con columna se prueba en contra de la selección. Si la celda falla, entonces su valor se retira y es tratada como si ningun dato haya sido sacado de la celda. Nótese sin embargo, los encabezados de las columnas que han sido pedidas como resultados del ACROSS no cambian aún y cuando todo los valores bajo esa columna no pasan la selección TOTAL.

EJEMPLO: Utilizando totales con ACROSS

```
>> TABLE FILE PROD
>> COUNT PRODUCTS
>> ACROSS MONTH
>> BY CUSTOMER
>> IF TOTAL COUNT IS-LESS-THAN 7
>> IF AREA IS WEST
>> END
```

El reporte aparece como:

| | MONTH | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| CUSTOMER | | | | | | | | | | | | |
| AJAX CO | 4 | 3 | . | 1 | 2 | . | 5 | 4 | 2 | 1 | . | 2 |
| BENTLY INC | 2 | 6 | 3 | 1 | 4 | . | 3 | 2 | 4 | 3 | . | 3 |
| COKE BROS | 1 | 4 | 5 | 2 | 3 | . | 5 | 2 | 1 | 4 | 3 | 2 |

Relación de la selección de TOTALES con archivo HOLD

Un reporte no tiene que ser impreso. Puede sostenerse, haciendo una petición subsecuente en contra de esta archivo 'HOLD' (ver la sección archivos HOLD). Ya que la petición subsecuente usa los valores ya acumulados en el archivo HOLD, automáticamente ejecuta una selección en los totales. Por lo tanto, el mismo efecto puede obtenerse por las dos operaciones de:

1.- Preparar un reporte y sostenerlo en un archivo HOLD.
2.- Seleccionar los contenidos del archivo HOLD y desplegar los valores del archivo.

El archivo HOLD tiene otros usos, pero si uno de sus usos ha sido seleccionar un final total, entonces es más fácil y rápida usar la selección TOTAL en la petición original.

Si se prepara un reporte matriz en la peticion original, entonces el hecho de que las columnas no cambien como resultados de las condiciones de prueba totales produce un efecto diferente entre la selección TOTAL y el acercamiento del archivo HOLD. Esta diferencia tiene un uso cuando se desea desplegar un set de columnas arregladas en un reporte matriz, independientemente de los registros pedidos.

Leyendo los Valores de Prueba de un Archivo

En lugar de escribir todas las condiciones de pruebas literales en el enunciado de petición, se puede usar una referencia del archivo que contiene las literales. Los datos de este archiuvo serán sustraídos y sustituidos como las literales en la frase de selección.

1.- Permite que los grupos usados con mas frecuencia sean almacenados y mantenidos aparte de los enunciados de petición los cuales pueden ser usados.

2.- Un archivo puede ser contruído como resultado de un enunciado de petición (archivo HOLD ó archivo SAVE) y es usado entonces para proporcionar selecciones para enunciados de petición subsecuentes. Esto es particularmente de gran utilidad en casos en donde las condiciones originales de selección no pueden ser usados inmediatamente para localizar los registros deseados. Por ejemplo. todos los registros en donde un valor excede el promedio de todos los valores.

La sintáxis para referirse a un archivo de prueba de literales es:

IF fieldname relation (filename) or (filename)...etc..

En lugar de una literal, se encierra en paréntesis un nombre de archivo.

Un grupo de nombres de archivos puede ser usados, separados por la palabra OR. También, se puede intermezclar los literales reales con los nombres de archivos. Ej.

IF fieldname relation (filename) or literale... etc..

El nombre de archivo tiene que:

1.- Estar definido en un FILEDEF ó

2.- Tener el tipo de archivo .DAT

Esto le permite al archivo externo ser localizado.
Los valores de los datos deben aparecer uno cada línea y se usara el primer valor encontrado. Puede haber otras informaciones en el archivo externo en cada línea.

El primer valor de datos debe comenzar en la columna uno en el archivo externo. Los valores de datos deben tener el formato de caracteres y los dígitos numéricos serán convertidos a números computacionales internos cuando sea necesario.

Sin embargo, si el archivo externo tiene el nombre de HOLD , entonces se asume que los valores de los datos estan en un formato interno. Un archivo HOLD es un archivo temporal FOCUS el cual puede se creado durante el transcurso de la sesión.

EJEMPLO: Leyendo Pruebas Literales de un Archivo

```
>> TABLE FILE PROD
>> SUM UNITS AND MONTH
>> BY CUSTOMER BY MONTH
>> IF PRODE-CODE IS (EXPER)
>> IF AREA IS EAST
>> END
```

EJEMPLO: Realizando Pruebas Dinámicas

```
>> TABLE FILE PROD
>> SUM AVE.UNITS BY MONTH
>> PRINT UNITS AND HOLD AND COMPUTE
>> OVER=IF C1 GT C2 THEN 1 ELSE 0
>> BY MONTH BY CUST-CODE
>> END
>> TABLE FILE HOLD
>> PRINT CUST-CODE AND HOLD
>> IF OVER IS 1
>> END
>> TABLE FILE PROD
>> LIST PROD_TYPE AND CUSTOMER AND UNITS AND AMOUNT
>> IF CUST-CODE IS (HOLD)
>> END
```

CONDICIONES DE CONTROL

Frase "ON"

Se proporciona una gran variedad de acciones los cuales se relacionan con lo que le suceden al reporte escrito cuando un sorteo cambia los valores del campo. Por ejemplo, se quiere desplegar un SUB-TOTAL?.

La sintáxis para especificar las condiciones de control usa una frase que comienza con la palabra "ON". Este es seguido por el nombre del campo. Cuando este campo cambia su valor en el reporte escrito, entonces justo antes de que se imprima el siguiente valor, se efectúa la acción mencionada.

| | | |
|---|---|---|
| | SUB-TOTAL [MULTILINE] | Acumula y desplega todos los subtotales. |
| | SUBTOTAL | Acumula y desplega el subtotal |
| | PAGE-BREAK | Brinca de hoja. |
| ON FIELDNAME | SKIP-LINE | Salta de linea. |
| | FOLD-LINE | Se pliega una larga linea. |
| | SUMMARIZE | Suma todos los campos en función de algun SORT especificado. |
| | RECOMPUTE | Suma solo un campo de algun SORT indicado. |
| | SUP-PRINT,NOPRINT | Se suprime la impresion de algun campo. |
| | COMPUTE | Realiza cálculos entre campos. |
| | UNDER-LINE | Dibuja una linea atraves de la página. |
| | SUBFOOT | Inserta un texto libre despues de los valores. |
| | SUBHEAD | Inserta un texto libre antes de los valores. |

CALCULOS

Los cálculos son cualquier manipulación de valores de datos para producir valores nuevos. Esto puede involucrar procesos aritméticos, funciones matemáticas, funiones lógicas o la edición de caracteres.

El lenguaje de petición de los reportes del FOCUS realiza cálculos en tres formas. Los datos de entrada para estos cálculos peden ser:

1.- Los campos individuales de cada registro de datos.

Los nuevos datos de campos temporales se define como cualquier combinación de campos de datos reales o de otros temporales. Estos campos se puede entonces usar una petición de reportes exactamente como campo de datos reales. Pueden ser seleccionados, sorteados, acumulados, etc. El comando DEFINE se usa para crear estos campos de datos temporales.

2.- Los resultados de acumulados de las columnas en una petición de reportes.

Se crean nuevas columnas en un reporte basados en las columnas que ya existen. Ya que las columnas existenetes pueden ser acumulaciones que resultan de una frase de petición como SUM AMOUNT AND UNITS, las columnas nuevas usan estos campos acumulados. La palabra clave COMPUTE en la porción de la frase verbal del enunciado de petición es usada para especificar las operaciones a realizar en las columnas de salida del reporte.

3.- Los resultados acumulados del sub-total

Nuevas hileras son creadas basadas en los valores de cual sería el sub-total de las columnas en el momento en que se realizen los cálculos. La palabra clave RECAP es una condición de control usada para especificar cuando el nuevo renglon ha de ser computado. Por lo general solo se usan valores numéricos como datos de entrada para este tipo de cálculos.

Calculando Campos de Datos Temporales

EL COMANDO DEFINE

Los campos de datos temporales pueden ser definidos como combinaciones matemáticas o lógicas de campo de datos reales o de otros temporales. Una vez definidos, pueden ser más tarde en peticiones de reportes de la misma manera como se usan los campos de datos reales, Ej. para control de sorteos, como objetos verbales, como campos de selección , etc. Todas las opciones de generación de reportes en el enunciado de petición como AVE., SUBTOTAL.,PAGE-BRAKE, ETC..son aplicables.

Algunos usos de los campos temporales son:

1.- Calcular valores numéricos nuevos que no estan en el registros de datos.

2.- Calcular valores numéricos condicionales basados en la lógica condicional IF-THEN-ELSE.

3.- Calcular nuevas filas de caracteres alfanuméricos provenientes de otras filas.

4.- Manipular fechas para computar durabilidades.

5.- Clasificar valores de datos en rangos o grupos arbitrarios.

6.- Cambiar el formato impreso de un campo temporal sin cambiarlo permanentemente al diccionario maestro.

7.- Causar que se invoquen funciones del usuario escritas con los cálculos realizados y usados en el reporte.

Sintáxis del comando DEFINE

Los campos de datos temporales son definidos para un archivo dado bajo el comando DEFINE. La sintáxis es:

```
>> DEFINE FILE Nombre de campo
>> Nombre/Formato= Expresión;
>> Nombre/Formato= Expresión;
>>        :          :
>>        :          :

>> END
```

Por Ejemplo:

```
>> DEFINE FILE PROD
>> PRICE = AMOUNT/UNITS;
>> ADJUSTED/D8.2 = IF FACTOR EQ 0 THEN PRICE
                                  ELSE FACTOR*.5*PRICE;
>> END
```

El número de campos computados definidos actualmente para un archivo dado está limitado por el requerimiento de que el número total de campos computados no excedan 256, y que el largo combinados de todos los campos de la base de datos y de los campos definidos sea menor que 12,288 caracteres.

Se requieren tres elementos para definir un campo computado:

NAME — Un nombre de 12 caracteres para el cual puede ser referido en un enunciado de petición o por otros campos computados.

FORMAT — El formato de despliegue. La misma selección disponible para campos reales.

EXPRESSION — El cálculo o cálculo condicional que se va a efectuar puede ser hasta 40 lineas de largo para un campo determinado. Debe concluirse con un punto y coma.

Definir Nombres de Campos

El nombre usado para un campo DEFINE puede ser de hasta 12 caracteres de largo. El nombre puede ser intencionalmente el mismo que el nombre de un campo actual o de un campo computado previamente. En tales situaciones una petición de reporte se usará los valores asociados con la última definición. Esto se hace para facilitar los siguientes tipos de preguntas:

En una petición catalogada larga hay varias referencias al campo actual llamado PRICE. Deseamos simular un incremento del 10% del precio y ejecutar la petición catalogada sin alteraciones.

```
>> DEFINE FILE XYZ
>> PRICE = 1.1*PRICE;
>> END
>> EXEC petición catalogada
```

Archivos Múltiples con Valores DEFINE

La sintáxis para definir los campos en varios archivos en un mismo tiempo es:

```
>> DEFINE FILE archivo1
>> Nombre/Formato = Expresión;
>> Nombre/Formato = Expresión;
>>      :
>> END
>> DEFINE FILE archivo2
>> Nombre/Formato = Expresión;
>> Nombre/Formato = Expresión;
>>      :
>> END
```

Las definiciones computadas para un archivo permanecen intactas para la duración de la sesión o hasta que otro comando DEFINE sea editado para ese archivo. Cuando esto ocurre, solo se borraran todas las definiciones para ese archivo y los nuevos son usados a menos de que la palabra ADD siga al nombre de archivo. Las definiciones pueden borrarse sin proporcionar nuevas, escribiendo:

```
>> DEFINE FILE Nombre de Archivo CLEAR
>> END
```

Todas las expresiones computadas definidas bajo el comando DEFINE serán borradas al final del curso cuando la palabra FINISH sea escrita.

Si las expresiones deben almacenarse porque se usan con frecuencia, entonces se pueden catalogar expresiones permanenetes por medio del DIALOGUER MANAGER y despues solicitado para ser usado escribiendo el nombre del procedimiento EXEC.

EXPRESIONES DE CALCULO

Las expresines FOCUS para calcular campos de datos pueden usarse

1.- Operaciones Aritméticas
2.- Operaciones Lógicas
3.- Funciones Especiales
    a.- Funciones Aritméticas
    b.- Funciones de Fecha
    c.- Funciones de Edición
    d.- Sucesión de Datos
    e.- Decodificación

Operaciones Aritméticas

| | |
|---|---|
| + | Más |
| - | Menos |
| * | Multiplicación |
| / | División |
| ** | Exponente |

Operaciones Lógicas

| | |
|---|---|
| EQ | Igual |
| NE | No Igual |
| LE | Menor Qué ó Igual |
| LT | Menor Qué |
| GE | Mayor Qué ó Igual |
| GT | Mayor Qué |
| AND | Conexión Lógica AND |
| OR | Conexión Lógica OR |
| NOT | Conexión Lógica NOT |
| CONTAINS | Caracter en el segundo operando contenidos en el primero. |
| OMITS | Caracter en el segundo operando omitidos en el primero. |

Estos operandos son usados con mas frecuencia en cálculos condicionales del tipo IF-THEN-ELSE.

La sintáxis general es:

IF expresión THEN resultado 1 ELSE resultado2

En donde el resultado 1 o 2 pueden ser otro IF-THEN-ELSE

Nota:

1.- Las literales alfanuméricas deben estar encerradas entre comillas sencillas. ej. IF AREA EQ 'NORTH'

2.- Cada expresión puede ser de hasta 40 lineas pero finalizar con un punto y coma.

3.- Se pueden usar los paréntesis para ser mas claro el orden para ver si se puede dar como resultado una redundancia.

4.- Los campos de datos temporales pueden tener cualquier formato disponible a campos de datos reales. Por lo tanto las variables alfanuméricas pueden ser definidas.

FUNCIONES ESPECIALES

Las expresiones definidas tienen a su disposición una variedad de funciones especiales. Estas realizan operaciones a la orden que por lo general son de gran uso. Los usuarios pueden proporcionar sus propias funciones especiales.

EJEMPLOS:

Funciones Aritméticas

```
ABS                 Valor Absoluto
---
PRICE/D9.2 = ABS(AMOUNT-OLDAMOUNT)/100;

INT                 Parte Entera
---
YEAR/I16 = INT(DATE/10000);

MAX                 Valor Máximo
---
LARGE/D8 = IF FACTOR GT 10 THEN MAX(10,AMOUNT)
                           ELSE MAX(0,AMOUNT,VALUE/10);
```

```
MIN                 Valor Mínimo
---
LOW/16 = MIN(0,AMOUNT,NEWAMOUNT,OTHER);

LOG                 Logaritmo, Base e
---
VAL = 100*AMOUNT*LOG(PRICE);

SQRT                Raiz Cuadrada
----
VALUE = 100*AMOUNT/SQRT(TOTAL);
```

Nótese que los argumentos de las funciones pueden ser espresiones aritméticas. Ej. SQRT(VALUE*FACTOR/UNITS);

FUNCIONES DE FECHAS

YMD — Duración entre dos fechas almacenadas en la forma año-mes-día. Por ejemplo, 760422. El resultado es el número de días entre dos fechas incluyendo años bisiestos.La sustracción es la segunda fecha menos la primera.

```
DAYS/I6 = YMD(BEGINDATE,ENDDATE);

si BEGINDATE = 760112
     ENDDATE = 760203  entonces,
        DAYS = 22
```

MDY — Duración entre dos fechas almacenadas en la forma de mes-día-año. Por ejemplo 031176. El resultado es el número de días entre el primero y segundo argumentos proporcionados.

```
DAYS/I4 = MDY(TERMDATE,031176);
```

DMY — Duración entre dos fechas almacenadas en una anotación DDMMYY, Ej, 031175. El resultado es el número de días entre el primero y segundo argumento proporcionado.

```
DAYS = MDY(FINDATE,STDATE);
```

Refiriendose a Operaciones Directas en Cálculos

La sintáxis para ejecutar operaciones en campos nombrados como objetos verbales, requieren que el prefijo sean separados por un punto(.) o un asterico (*) del nombre del campo de datos. Por ejemplo, MAX.COST. Se recomienda prefijar los campos con un punto.

EJEMPLO: Refiriendose a Campos Prefijados en una Tabla

```
>> TABLE FILE CARS
>> WRITE MAX.RCOST AND MIN.RCOST AND COMPUTE
>> SPREAD=100*(MAX.RCOST-MIN.RCOST)/MAX.RCOST;
>> BY MPG IN-GROUPS-OF 10
>> END
```

La misma sintáxis es aplicada cuando se hace referencia a campos con encabezados o en pie de columnas, Por ejemlo, "HIGHEST MILEAGE WAS <MAX.MPG>".

La lista de Operaciones Directas son:

| | |
|---|---|
| AVE.field | Promedio. |
| ASQ.field | Promedio de la suma de cuadrados (raíces). |
| MIN.field | Valor Mínimo. |
| MAX.field | Valor máximo. |
| LST.field | Ultimo Valor. |
| FST.field | Primer Valor. |
| PCT.field | Porcentaje de Total de Columna. |
| TOT.field | Total Final. |
| ALL.field | Todos los casos con o sin descendientes. |
| SUM.field | Suma de Valores. |
| CNT.field | Conteo de Valores. |

COLUMN-TOTALS

Los totales en una columna son una característica tan común en un reporte que se puede hacer una petición directa para ellos. La petición se hace en la porcion verbal del enunciado de petición, agregando las palabras AND COLUMN-TOTAL. Los totales de las columnas bajo cada columna numérica será desplegada al final del reporte. Una sintáxis de alternativa es el de agregar la frase ON TABLE COLUMN-TOTAL al enunciado de petición.

EJEMPLO: Total de Columnas

```
>> TABLE FILE PROD
>> "OVERALL RESULTS FOR ALL PRODUCTS"
>> SUM UNITS AND AMOUNT
>> BY AREA
>> ON TABLE COLUM-TOTAL
>> END
```

OVERALL RESULTS FOR ALL PRODUCTS

| AREA | UNITS | AMOUNT |
|---|---|---|
| EAST | 1000 | 4,506.40 |
| NORTH | 1200 | 2,104.60 |
| SOUTH | 1800 | 349.50 |
| WEST | 1100 | 1,248.25 |
| TOTAL | 5100 | 8,208.75 |

En donde quiera que sea pedido un sub-total, la columa de los totales se producen automáticamente.

ROW-TOTALS

El total de todas las columnas numéricas desplegadas en una línea es una característica muy necesitada. Por lo tanto, se puede hacer una referencia directa al ROW-TOTAL. Esto se puede hacer agregando la palabra AND ROW-TOTAL a la porción del verbo del enunciado de petición .

EJEMPLO: Total de Renglones

```
>> TABLE FILE PROD
>> SUM UNITS AND ROW-TOTAL
>> ACROSS MONTH FROM 1 TO 3
>> BY AREA
>> END
```

El reporte aparece como:

| | MONTH | | | |
|---|---|---|---|---|
| | 1 | 2 | 3 | TOTAL |
| AREA | | | | |
| EAST | 500 | 800 | 200 | 1500 |
| NORTH | 600 | 400 | 940 | 1940 |
| SOUTH | 50 | 100 | 30 | 180 |
| WEST | 120 | 210 | 400 | 730 |

# ARCHIVOS EXTERNOS

* HOLD como un archivo FOCUS

* SAVE como un archivo externo

ARCHIVOS EXTERNOS

*Almacenando y Reteniendo Información*

Un enunciado de petición no tiene que originar que un reporte sea impreso de inmediato. En lugar, se pueden especificar tres otras acciones. Estas son:

| | |
|---|---|
| HOLD [AS Nombre] | Un reporte es preparado pero no impreso. El reporte crea un archivo temporal nombrado HOLD del cual se pueden dirigir otras peticiones, permitiendo que cuestiones del tipo "post-procesados" se pueden formular fácilmente. |
| SAVE [AS Nombre] | Las líneas de los datos en el reporte son escritas en un archivo externo. Se preparará un archivo externo. |

Reteniendo un Reporte- El Archivo HOLD

El FOCUS proporciona la habilidad de reintegrar y procesar información y después retener los resultados para ser procesados posteriormente. Ya que los resultados de la primera petición están siendo retenidos, se habló ahora directamente del archivo HOLD. Los campos de este archivo HOLD tienen los mismos nombres como si aparecieran en los encabezados de la columna si el reporte hubiera sido impreso. Las peticiones subsecuentes se refieren entonces a un archivo HOLD y usados como nombres de campo. El archivo HOLD es un archivo completo externo válido con una descripción temporal en el diccionario del archivo. Nótese que el ALIAS no se corre del archivo original al archivo HOLD. Se proporciona un nuevo grupo de ALIAS automáticamente. Estas son E01,E02, etc..

EJEMPLO: Archivo HOLD

Problema: Un archivo contiene productos y envíos por un area mensual. Encontrar todos los productos (PROD-TYPE) con los envíos totales (UNITS) arriba de 10,000 para el año.

```
>> TABLE FILE PROD
>> SUM UNITS
>> BY PROD-TYPE
>> ON TABLE HOLD
>> END
>> TABLE FILE HOLD
>> PRINT PROD-TYPE AND UNITS
>> IF UNITS FROM 10000
>> END
```

Sintáxis del HOLD:

La petición para un HOLD puede ser colocada en la porción del verbo del enunciado. Ej. SUM UNITS AND HOLD, pero se prefiere la frase ON TABLE HOLD. Además se puede imprimir un reporte y después retenerlo, si ningun otro comando ha sido editado después de haber sido impreso el reporte. En este caso la palabra HOLD es escrita como un comando.

EJEMPLO: Comando HOLD

```
>> TABLE FILE PROD
>> SUM UNITS
>> BY MONTH BY AREA
>> IF PROD-TYPE IS 'X$$$$$$$$$'
>> END
>> HOLD
```

El archivo HOLD se mantiene en uso por la duración del curso y se le puede referir como si fuera un archivo verdadero tanto como se necesite. Es escrito dentro de un disco como un archivo temporal y borrado al final del curso. Un segundo archivo creado durante el curso reemplaza al primero. Por lo tanto, solo un archivo llamado HOLD puede activarse en un momento determinado. Sin embargo un archivo HOLD puede dársele un nombre único al momento de ser creado, y en cualquier número de estos pueden ser usados.

La definición del arhivo HOLD es:

```
FILEDEF HOLD DISK C:HOLD.FTM
```

La letra del archivo cambia automáticamente para colocar el archivo HOLD en el disco pueda escribirse conectado con la cantidad más grande de espacio usable.

Reteniendo el archivo HOLD

En el momento en que se crea un archivo HOLD se le puede asignar un nombre. El archivo creado puede ser usado inmediatamente bajo ese nombre, de la misma manera como se usa el nombre de omisión HOLD. El nombre es proporcionado en el enunciado de petición en la forma:

```
>> ON TABLE HOLD AS Nombre
```

La estructura de un archivo HOLD re-nombrado puede preguntando escribiendo: ? HOLD Nombre

Al final de la sesión un archivo HOLD re-nombrado permanece en el disco bajo el campo de:

Nombre.MAS (Descripción)

Nombre.FTM (Datos)

Puede ser leído en otra sesión FOCUS como un archivo externo regular usando un FILEDEF:

```
FILEDEF Nombre DISK C:Nombre.FTM
```

Cualquier número de archivo del tipo HOLD pueden ser creados en una sesión FOCUS.

EJEMPLO: Re-nombrando Archivos HOLD

```
>> TABLE FILE CARS
>> SUM AVE.MPG
>> BY CAR
>> ON TABLE HOLD AS MILFIL
>> END
>> TABLE FILE MILFIL
>> PRINT CAR BY HIGHEST MPG
>> END
```

Produciendo un Archivo Externo- La opción SAVE

En lugar de imprimir un reporte formateado como resultado de un enunciado de petición, las líneas reintegradas y procesadas pueden ser escritas en un archivo externo y almacenados. Solo se escriben líneas de datos; no se escriben encabezados o sub-totales. Los campos se escriben uno seguido de otro sin el espacio usual de los intervalos, el archivo externo es un archivo de sequencia. Puede ser usado por otros programas, o puede ser re-integrado dentro de otros archivos FOCUS.

Para producir un archivo externo se agregan las palabras AND SAVE a la porción del verbo del enunciado de petición o la frase ON TABLE SAVE.

EJEMPLO: Produciendo un archivo externo llamado SAVE

```
>> TABLE FILE PROD
>> PRINT PROD-TYPE AND AREA
>> BY MONTH IN UNITS IS 0
>> ON TABLE SAVE
>> END
```

```
Numero de Registros en la Tabla = 1590  Lineas = 1590

EBCDIC  RECRD  NAMED   SAVE

FIELD       FORMAT   LENGTH

MONTH       I2       2
PROD-TYPE   A10      10
AREA        A8       8

TOTAL                20
```

Nótese el desplegado después de los registros reintegrados. Describe la disposición del archivo externo.

Archivo Externo - Nombre Omitido

El *nombre omitido del archivo externo* es SAVE . Un FILEDEF es editado internamente si el usuario no ha editado el suyo propio, el cual es:

FILEDEF SAVE DISK C:SAVE.FTM

El archivo omitido SAVE será escrito dentro del disco conectado que tiene el mayor espacio disponible. El parámetro del nombre de archivo en el FILEDEF toma ese valor.

*Renombrando el Archivo Externo - La opción* "SAVE AS"

Si dos peticiones, una seguida de la otra, son elaboradas y ambas producen archivos externos, entonces el segundo grupo de registros de reintegración , son escritos sobre el primer grupo. Por lo tanto es necesario tener dos archivos externos, o cualquier número requerido. Esto se puede lograr proporcionando otro nombre al archivo SAVE, cuando uno es pedido. La sintáxis es el de agregar el nombre que puede ser de hasta 8 caracteres de largo inmediatamente después de la palabra SAVE, en la forma:

ON TABLE SAVE AS *Nombre*

EJEMPLO: Re-nombrando un archivo SAVE

```
>> TABLE FILE PROD
>> SUM AMOUNT AND COUNT
>> BY PROD-TYPE BY MONTH
>> IF UNITS EXCEEDS 1000
>> ON TABLE SAVE AS MYSAV
>> END
```

El Comando FILEDEF

El comando PC/FOCUS FILEDEF permite asociar un archivo de un disco físico que tiene una letra, nombre y sufijo de disco con un nombre lógico de 8 caracteres que el FOCUS puede ser. Esto permite cambiar el archivo físico sin cambiar cualquiera de los procedimientos FOCUS los cuales pueden referirse al mismo nombre lógico.

La sintáxis es:

```
                      DIAL
                      DISK
    FILEDEF Nombre    PRINTER  C: Nombre.EXT   Opciones
                      TERM
```

En donde las opciones son:

| | |
|---|---|
| LRECL nnn | Para fijar un tamaño de registro establecido en lugar de un tamaño variable para cada registro. |
| APPEND | Para permitir que los registros nuevos se agreguen al final del archivo de datos cuando el archivo ya existe. |

En MS-DOS

Si el usuario ha editado un FILEDEF entonces los registros extraídos serán escritos en la designación pedida.

Si no se ha editado un FILEDEF posterior, entonces se edita una omisión internamente usando el nombre proporcionado en el enunciado pedido. Por ejemplo:

```
FILEDEF MYSAV DISK C:MYSAV.FTM
```

Los usuarios pueden crear una sucesión de diferentes archivos externos en el disco durante un curso sin tener que editar otro FILEDEF anteriores.

El disco seleccionado para escribir en él es el disco conectado con el espacio más grande sin usar. Esto se hace automáticamente.

Almacenando despues de Imprimir

Despues de haberse producido un comando TABLE puede aún ser convertido en un archivo externo. En el nivel de comando, antes de que cualquier comando intervenga, se imprime la palabra SAVE. La acción es similar al de RETYPE, pero en lugar de otra copia impresa, las líneas del reporte se escriben en un archivo externo.

EJEMPLO: SAVE como un comando

```
>> TABLE FILE PROD
>> SUM UNITS
>> BY AREA
>> END
>> SAVE
```

El Comando JOIN

El FOCUS proporciona el comando JOIN, en donde se combinan datos de varios archivos FOCUS o de archivos secuenciales. La sintáxis es:

```
JOIN campo1 IN archivo1 TO [ALL] campo2 IN archivo2 AS nombre de union
```

en donde

| | |
|---|---|
| campo1 | Es cualquier campo en el archivo llamado archivo1 |
| campo2 | Es cualquier campo en el archivo llamado archvio2 |

que tiene un índice o un método equivalente de acceso directo.

Después de haber ejecutado un comando JOIN, El FOCUS le permite a los comandos TABLE y DEFINE que sean aplicados como si archivo1 fuera un archivo nuevo desarrollado de ambos archivos originales.

En el siguiente ejemplo el archivo PROD es unido a un archivo que contiene informaciones sobre comisiones de ventas por territorios en venta, para producir un reporte compuesto.

EJEMPLO: El Comando JOIN

```
>> JOIN AREA IN PROD TO ALL TERRITORY IN SALESCOM AS NEW
>> TABLE FILE PROD
>> SUM UNITS BY AREA
>> SUM POINTS AND COMPUTE RATIO = POINTS/UNITS;
>> BY AREA BY SALESREP
>> ON AREA SUMMARIZE
>> END
```

El reporte aparece como:

| AREA | UNITS | SALESREP | POINTS | RATIO |
|---|---|---|---|---|
| EAST | 1642 | BROWNE | 862 | .52 |
| | | MEHTA | 1984 | 1.21 |
| | | RUBIN | 1482 | .90 |
| | | VRONSKY | 1640 | 1.00 |
| * TOTAL EAST | 1642 | | 5.968 | 3.63 |
| NORHT | 2296 | GRASSO | 2122 | .92 |

:
:
ETC..

Se puede efectuar hasta 16 JOINS al mismo tiempo, un archivo una vez unido, puede unirse o ser unido de nuevo. Y el archivo unido puede en si, es una parte de una unión.

CONTROL DE FORMATO

* ENCABEZADOS
* PIE DE PAGINA
* TITULOS DE COLUMNAS
* ESPACIO DE COLUMNAS

El FOCUS formulará completamente un reporte basados en la información sobre campos de datos el cual se encuentra en el diccionario central. Casi todas las opciones usadas pueden ser anuladas. Para visualizar el proceso de despliegue del reporte y las varias opciones que el usuario elija, se considera el siguiente tabulador. Nótese que hay nuevos tipos de líneas que pueden aparecer en un reporte.

1.- Encabezados de Reporte
2.- Encabezados de Página
3.- Títulos de Columna
4.- Líneas de Subencabezados
5.- Líneas de Contenido
6.- Líneas de Subtotal
7.- Subpie de Página
8.- Pie de Página
9.- Pie de Reporte

Encabezados de Reporte

Un encabezado de reporte puede imprimirse al frente de la primera página del reporte o en la parte superior de la primera página arriba de cualquier línea como el encabezado de página. La sintáxis es:

```
ON TABLE PAGE-BRAKE AND SUBHEAD
"     TEXTO      "
"     TEXTO      "
      :
      ETC...

ON TABLE SUBHEAD
"     TEXTO      "
```

El uso de frase PAGE-BREAK es opcional y pondrá en posición el encabezado del reporte en una hoja separada, el cual es seguido por la primera página del reporte. La facilidad de poner en posición el texto y para insertar datos en el texto, puede estar en disposición para usarse.

EJEMPLO: Encabezado del Reporte

```
>> TABLE FILE PROD
>> SUM UNITS BY MONTH BY PROD-TYPE ACROSS AREA
>> ON TABLE PAGE-BREAK AND SUBHEAD
>> " SUMMARY OF UNITS SALES IN EACH MONTH"
>> "                FOR                  "
>> "             XYZ COMPANY             "
>> END
```

Pie de Reporte

Un pie de reporte puede escribirse al final, ya sea en la última página o en una pa®gina aparte. La sintáxis es:

```
>> ON TABLE PAGE-BREAK AND SUBFOOT
>> "            TEXTO          "
>> "            TEXTO          "
>>               :
>>               :
>>             ETC...
>> 0
>> ON TABLE SUBFOOT
>> "              TEXTO          "
>> "              TEXTO          "
```

El uso del PAGE-BREAK hace que el pie del reporte aparezca en una hoja aparte después de la última hoja del reporte. Todas las posiciones del texto de apoyo y la inserción de datos en el texto están a disposición para su uso.

EJEMPLO: Pie de Reporte

```
>> TABLE FILE PROD
>> SUM UNITS BY MONTH BY PROD-TYPE
>> ON TABLE SUBFOOT
>> " TOTAL UNITS SOLD WERE <ST.UNITS"
>> END
```

Líneas de Encabezados

El encabezado del reporte es proporcionado enteramente por el usuario. Puede ocupar de 1 a 57 líneas de la página. Se pueden designar valores de datos reintegrados para colocarse en el momento del curso dentro del encabezado. Si no se proporciona un encabezado, entonces se imprime una línea en blanco por omisión arriba de los títulos de la columna.

Se proporciona un encabezado en el enunciado de petición encerrando el texto deseado entre comillas dobles.

EJEMPLO:

```
>> TABLE FILE PROD
>> " MUESTRA DE COMO COLOCAR UN ENCABEZADO "
>> " ARRIBA DE CADA PAGINA EN EL REPORTE DE SALIDA"
>> PRINT UNITS BY MONTH
>> END
```

Líneas de Subencabezado

Antes de imprimir una línea de contenido, se puede imprimir un encabezado arriba de la línea. Este encabezado es controlado por los campos de sorteo y contiene un texto libre, más datos intercalados.

EJEMPLO: Subencabezados

```
>> TABLE FILE PROD
>> SUM UNITS BY AREA NOPRINT BY PRODUCT
>> ON AREA SUBHEAD
>> END
```

Líneas de Pies - Utilizando comando FOOTING

De la misma manera que cada página puede tener un encabezado arriba de las líneas de contenido, puede tener un pie de página abajo de las líneas de contenido. El texto del Pie de Página es proporcionado entre signos de comillas y es impreso exactamente como es recibido precedido por la palabra FOOTING en una línea.

EJEMPLO: Notas al Pie de cada Página

```
>> TABLE FILE PROD
>> PRINT UNITS BY AREA
>> FOOTING
>> "THIS IS FOOTNOTE IS ADDED TO EACH"
>> "PRINTED PAGE"
>> END
```

La misma sintáxis que se usa para encabezados se usa aquí. Esto es comillas encierran las líneas de texto y son apareadas si son necsarias, para formar líneas de 128 caracteres; se pueden usar 57 líneas.

Los encabezados pueden centrarse y colocarse al final de la página la sintáxis completa es:

```
FOOTING [CENTER] [BOTTOM]
```

Líneas de Contenido

Las líneas de contenido son el resultado de la reintegraciones controladas por los enunciados de petición. Varias opciones de formato afectan al desplegado de las líneas del contenido.

| | |
|---|---|
| ON Nombre del Campo SKIP-LINE | Insertará un espacio después de los bloques de las líneas del contenido para mejorar la visibilidad o doblegará el espacio del reporte completo. |
| ON Nombre del Campo FOLD-LINE | Convertirá una línea en dos líneas y reducirá su espacio a lo ancho de la página. |
| ON Nombre del Campo PAGE-BREAK | Cuando el campo nombrado cambia de valor, las líneas del contenido comienzan a escribirse en una página nueva. |
| OVER | Colocará la línea del contenido verticalmente en la página, un campo sobre otro en lugar de horizontalmente un campo seguido de otro. |
| BY Nombre de Campo SUP-PRINT | Sorteará las líneas del contenido por un campo normal, pero suprimirá imprimirlo. |

La posición de omisión de las líneas de contenido pueden ser anuladas por medio de la palabra clave "IN" despues del campo. Ej. SUM UNITS IN 50.

Títulos - La Frase AS

Los títulos arriba de cada columna asumirá por omisión el nombre del campo de datos como fue almacenado en el diccionario control. Se puede sustituir hasta 5 líneas de texto para este nombre de campo en el enunciado de petición. El texto es proporcionado en una frase que comienza con la palabra AS inmediatamente despues del nombre del campo. Ej.

```
>> WRITE CANTIDAD AS "TOTAL,SHIPMENTS,IN,MONTH"
>> BY AREA AS "REGION,OF,COUNTRY"
```

Nótese que las comas en el texto separan las líneas. Se pueden usar hasta 5 líneas de texto en un título de columna.

EJEMPLO: Títulos de Columnas

```
>> TABLE FILE PROD
>> PRINT PROD-TYPE AS 'CATEGORY,OF,PRODUCT'
>> AND AREA AS 'REGION'
>> BY MONTH AS 'M,O,N,T,H'
```

El titulo del reporte aparece como:

```
M
O
N        CATEGORY
T        OF
H        PRODUCT       REGION
-        --------      ------
```

Líneas de Sub-Totales

La frase "ON Nombre de Campo SUB-TOTAL" originará que el subtotal de las columnas de datos numéricos se despliegue cuando cambie de valor el campo nombrado. Una línea en blanco separará el primer sub-total de la última línea del contenido. Otro espacio en blanco separará el último sub-total del principio de las líneas de contenido nuevos. Se generará también totales completos con la frase SUB-TOTAL.

Líneas utilizando el comando RECAP

En lugar de, o con las líneas de sub-totales, se pueden desplegar tras recapitulaciones de las líneas de contenido. Estos son cálculos ejecutados con los valores de los subtotales de los valores de los datos hasta el punto de recapitulación. Se inicia con la frase de petición ON Nombre del Campo RECAP y despues seguido con los cálculos realizados cuando el campo normal cambia de valor.

Líneas de Subfoot

Después de que cambie de valor el campo de control y antes de imprimirse el siguiente valor, se pueden imprimir una serie de líneas de texto libre con datos intercalados.

Ancho de Columnas

El ancho de caracteres que abarca una columna es el largo del tamaño del formato de datos, o del tamaño del titulo. Los tamaños de las omisiones normales son especificadas por campo en el diccionario FOCUS. Entonces, si el formato de los campos de datos llamados UNITS es I7 entonces, siete caracteres serán colocados para el ancho de la columna. Si el formato es I3 entonces 5 caracteres serian el ancho, ya que el titulo es más largo que los datos. Si se proporciona un texto multi-linea en el enunciado de petición entonces lo mas ancho de estas líneas es comparada al tamaño del formato de datos. Cuando un campo de datos computacionales es una coma impresa en el despliegue, entonces el número de comas posibles es agregada al tamaño del formato. Por ejemplo una coma editada como D11.2 ocupará 13 caracteres ya que dos comas son posibles. Ej.

XX,XXX,XXX.XX

Nótese que el punto decimal cuenta como un caracter en el tamaño del formato.

El uso de una linea de titulo mas larga que el tamaño del formato de los datos es una manera conveniente de espacear el reporte a lo ancho de las columnas de la página. Por Ejemplo:

>> WRITE UNITS BY MONTH AS 'MONTH' Neutralizara el reporte completo hacia la derecha.

Espacio entre Columnas

El FOCUS coloca dos espacios entre cada columna en el reporte escrito. La suma de los anchos de las columnas más los espacios deben estar dentro del ancho de la página escrita. La omisión del ancho de la página es de 160 caracteres.

El número de omisiones de espacios antes de una columna, puede ser controlado cuando sea necesario.

IN y ACROSS

La frase IN puede usarse con ACROSS para especificar la columna iniciadora del grupo ACROSS completo y el espacio entre cada columna dentro del ACROSS.

EJEMPLO:

```
>> SUM UNITS IN +1 ACROSS MONTH IN 30
>> BY PRODUCT
```

Esto forzara un espacio entre las columnas de datos en la matriz, y la matriz completa comenzará en la posición 30.

IN y El Lenguaje Modelo Financiero

EJEMPLO:

```
>> SUM UNITS AS 'QUANTITY' IN 1
>> AND AMOUNT AS 'DOLLARS' IN 50
>> FOR AREA IN 20
>> EAST LABEL E OVER
>> WEST LABEL W OVER
>> RECAP COMP = 100*E/W; AS 'COMPARISON'
```

Esto colocará los valores de los datos de UNITS y AMOUNT en cualquier lado del modelo pequeño para AREA. El reporte será espaciado así:

```
1           20                50

QUANTITY    AREA              DOLLARS

3000        ESTE               9,845
12000       WEST              34,315

25          COMPARISON            29
```

Si las posiciones de las columnas seleccionadas están sobre puestas entonces la última columna a imprimirse aparecerá. No es un error sobreponer columnas, pero cuando la longuitud de los datos está en duda se debe usar un espacio relativo para evitar sobrepornerlos.

El valor relativo de IN +0 es aceptable y remueve todo el espacio al frente de un campo.

IN con OVER y FOLD-LINE

Cuando un campo está colocado sobre otro, entonces las posiciones son aplicadas a la línea en donde ocurre la referencia de campos. Por Ejemplo;

```
>> SUM AMOUNT BY AREA
>> BY PRODUCT IN 10
>> ON PRODUCT FOLD-LINE
```

Los valores de los datos PRODUCT comienzan en la columna 10 en la segunda línea.

Operaciones con Datos en Encabezados

El nombre de cualquier campo de datos mencionados pueden ponersele un prefijo por medio de una operación directa de la misma manera en cualquier objeto verbal lo puede también.

| Prefijo | Significado | Ejemplo |
|---|---|---|
| MAX | Valor Máximo | "Embarque mas Grande=<MAX.UNITS" |
| MIN | Valor Mínimo | "Embarque mas Pequeño=<MIN.UNITS" |

Y ASI PARA TODOS LOS DEMAS PREFIJOS:

| | |
|---|---|
| AVE | Valor Promedio |
| ASQ | Prom. de Suma de Cuadrados |
| PCT | Porcentaje del Total de Columna |
| FST | Primera Reintegración |
| LST | Ultima Reintegración |
| TOT | Total de Columna |
| CT | Total de Columna en Curso |
| ST | Sub-total Actual |

EJEMPLO: Total de Columna en Encabezados

```
>> TABLE FILE PROD
>> " TOTAL SHIPMENTS WERE EQUAL TO <TOT.UNITS"
>> PRINT UNITS AS 'UNITS,SHIPED'
>> BY PROD-TYPE IF MONTH IS 2
>> END
```

El reporte aparecerá como:

```
TOTAL SHIPMENTS WERE EQUAL TO 12562

PROD-TYPE          UNITS
                   SHIPED
---------          -------

AXLES              352
BEARINGS           241
   .                .
   .                .

ETC..              ETC..
```

Centrando Encabezados y Pies de Página

Si la palabra CENTER sigue a la palabra HEADING o FOOTING entonces cada línea de encabezado o pie de página será centrada sobre el cuerpo de datos en el reporte.

EJEMPLO:

```
>> TABLE FILE PROD
>> HEADING CENTER
>> "DISTRIBUTION REPORT"
>> "FOR"
>> " <AREA"
>> SUM UNITS AND AMOUNT AND PCT.AMOUNT
>> BY AREA NOPRINT PAGE-BREAK SUB-TOTAL
>> BY PRODUCT
>> END
```

El pie de la página seguirá a la última línea impresa de la porción tabular de la página. Puede colocarse en la parte interior de la página sin importar el número de líneas en la porción tabular agregando la palabra BOTTOM

```
FOOTING  BOTTOM
        0
FOOTING CENTER BOTTOM
```

Para reportes impresos en la terminal, la opción PAUSE=ON es de gran utilidad en este caso.

Suprimiendo Números de Página

El conteo automático de páginas puede ser suprimido con el comando FOCUS SET PAGE=OFF.

Para suprimir todos los indicadores de página, como es conveniente algunas veces en un CRTFORM se usa el comando FOCUS: SET PAGE=NOPAGE.

TOPICOS ESPECIALES

Formato de Datos

Uno de los atributos describiendo cada campo de datos en un archivo FOCUS es el USAGE FORMAT (Formato a Usarse) de los datos. Este atributo permite el tipo de datos, su longuitud y despliega las opciones de impresión para cada campo.

Las combinaciones de Tipos y Longuitudes son:

| TIPO | LONGUITUD | SIGNIFICADO |
|---|---|---|
| I | 1 A 9 | Número Entero sin Posición Decimal |
| F | 1 A 9 | Número Decimal Precisión Sencilla Exacta |
| D | 1 A 15 | Número Decimal Presición Doble Exacta |
| P | 1 A 15 | Número Decimal Empacado |
| A | 1 A 256 | Caracteres Alfanuméricos |

Las opciones de impresión afectan al reporte escrito solamente. Ellos no estan activos para archivos externos.

| OPCION | SIGNIFICADO | MUESTRA | EFECTO |
|---|---|---|---|
| S | Supresión de Ceros | I4S | Si el valor de los datos es cero se imprime un espacio en su lugar. |
| C | Impresión de Coma | P8.2C | Cada 3 digitos significativos, se inserta una coma. |
| B | Corchetes Negativos | F6.0B | Números negativos se encierran entre corchetes. |
| R | Crédito CR Negativo | D9.2R | Números negativos tiene la letra CR después de ellos. |
| M | Signo de Dollar Flotante | P9.2M | Un signo de dólar '$' es colocado antes del primer digito significativo. |
| N | Signo de Dollar No Flotante | D12.2N | Un signo de dolar '$' es colocada en una posición arreglada, hacia la izquierda del campo. |
| MDY | Mes-Día-Año | I6MDY | Una forma de fecha MMDDYY será impresa como MM/DD/YY. |

| | | | |
|---|---|---|---|
| T | Traducir MES con: | | |
| | MTDY<br>YMTY<br>DMTY<br>MT | I6MTDY<br>P6YMTY<br>A6DMTY<br>I2MT | Se imprimirá un nombre de MES de 3 caracteres en lugar del número. |
| L | Ceros Principales | P6L | Se agregan ceros al principio. |
| E | Anotación Científica | P12.5E | Solo dígitos no-cero serán desplegados. |

Todas las opciones pueden ser especificadas en cualquier orden. Ambas opciones M y N implican automáticamente la opción C, el formato tipo D, implica también automáticamente la opción C. La opción T de traducción de mes puede especificarse en cualquier lugar.

Cambiando las Posturas de Omisión

El comando SET

Varios parámetros a la cual se dieron valores de omisión, pueden cambiar sus valores durante una sesión FOCUS.. Los parámetros que pueden ser cambiados son:

| PARAMERTROS | VALOR | DESCRIPCION |
|---|---|---|
| LINES | 57 | El número de líneas en una página de reportes que seran impresos antes de comenzar una página nueva. |
| PAUSE | OFF | Los reportes impresos en vivo en una terminal de comunicación no hace pausa normalmente antes de comenzar a imprimir. Re-colocar el PAUSE a ON permitirá un ajuste normal del papel. |
| PAPER | 66 | El número máximo de líneas en el papel tiene que ser mas largo que el número de líneas realmente impresas para proporcionar un margen. |

| | | |
|---|---|---|
| PAGE-NUM | ON | La palabra PAGE y el número de página son impresos arriba cada página de un reporte. Colocar PAGE-NUM=OFF suprimirá la enumeración de páginas=NO PAGE eleminará los quiebres de página. |
| DANEL | 0 | Las páginas del reporte son impresas en paneles separads el cuál el ancho es el valor de esta variable. Un valor de cero (la omisión) significa no panelado. |
| MSO | ON | Cuando se coloca en OFF muchos mensajes informacionales son suprimidas. |
| ALL | OFF | Lios valores son ON, OFF O PASS. Los datos de omisión faltantes son cambiados para reintegrar records cortos de trayectoria. |
| FILENAME | - - | Cuando la palabra FILE está ausente de TABLE,DEFINE O MONEY, el nombre de este omisión nueva será usada. |
| DASS O USER | - - | Identifica al usuario para proteger el acceso al archivo. |
| BINS | 12 | El número de páginas centrales usadas para pulir la base de datos. Es computado internamente pero puede re-colocarse manualmente. |
| PRINT | ONLINE | Esto es funcionalmente equivalente a los comandos OFFLINE Y ONLINE, y una forma alternativa de dirigir la imprisión a una divisa fuera de línea (OFFLINE) es el de colcar o preparar PRINT=OFFLINE. |
| TEMPDISK | A | En dos la omisión para todo espacio de trabajo temporal es calcado internamente examinando todos los discos adheridos y seleccionando el disco usado con la cantidad mayor del espacio libre. El modo de archivo de este disco es usado en todos los FILEDEF°S de omisión. Puede ser cambiado especificamente a cualquier otro modo de disco. |

| | | |
|---|---|---|
| WIDTH | 160 | El ancho máximo de las líneas de salida de un reporte escrito. Esto es determinado automáticamente de la longitud de reporte del archivo de impresión. Sin embargo, puede hacerse más pequeño para corresponder a una terminal de impresión más angosta. |

Uso del Comando SET

El cambiar los valores de omisión requiere el comando FOCUS de SET. Sobre la misma línea y siguiendo el comando están los nombres de los parámetros para re-instalar sus valores nuevos se pueden re-colocar varios párametros sobre una línea. Son entonces separados un de otro por comas.

EJEMPLO: Cambiando Parámetros usando SET

```
>> SET LINES=61
>> SET LINES=61, TEMP DISK=59
>> SET PRINT=OFFLINE, LINES=59
>> SET PAUSE=ON
```

Descripción de los archivos FOCUS

INTRODUCCION:

Esta sección del manual del usuario está dedicado a describir, los atributos de los archivos FOCUS. Ellos son verificados y algunos de los conceptos de los archivos designados en la cual son usados en un rango de aplicaciones desde lo mas simple hasta lo mas complejo.

Convenciones Generales:

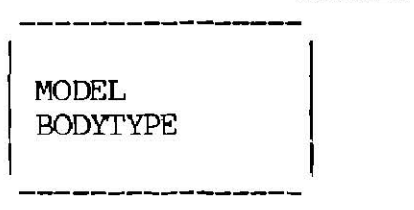
Diagramas

Una manera conveniente de ilustrar varios campos de datos y segmentos, es en un diagrama con cuadros, representando cada segmento de la estructura y flechas entre cada cuadro, mostrando el tipo de conexión. Insiden los cuadros con algunos de los campos de datos, usualmente la llave campo necesario para indentificar diferentes situaciones de los datos que pueden ser dados.

EJEMPLO: Grupo de campos representando información de un importe de carros son:

MODEL
BODYTYPE
SEATS
RETAIL COST
ENGINE CODE

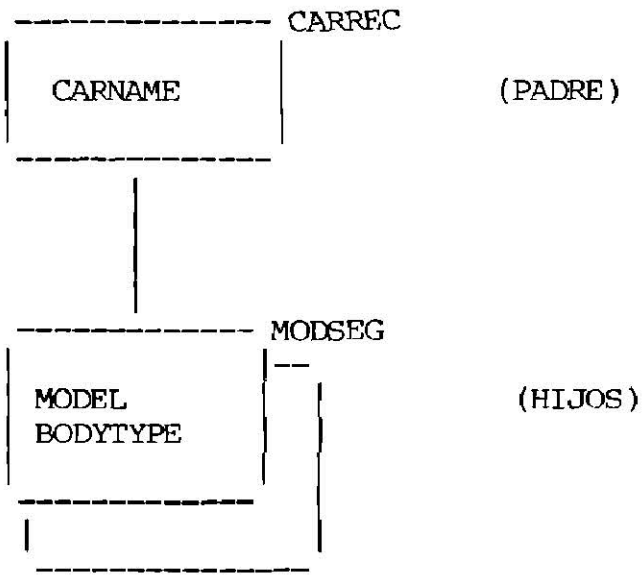
Y ellos son agrupados a la vez en un segmento MODSEG. Entonces un diagrama puede ser representado, como:




Relación de Segmentos.

Cuando los segmentos son relcionados el uno al otro, en una forma jerarquica en la cual un segmento es común a muchos segmentos descendente como un padre a su hijo, entre la información existente entonces puede ser dibujada línea del segmento padre al segmento hijo. Una flecha que el fin de la línea indica la dirección del parentesco. En un archivo FOCUS nosotros generalmente recorremos a un segmento hijo de un segmento padre.

Cuando un segmento ocurre un número múltiple de veces con relación al padre dado puede ser proyectado el cuadro.




Cross-Reference

Una linea sólida entre segmentos significa que un parentesco estructural existe entre los segmentos.

Por ejemplo si en cierto campo del segmento uno es la misma información que hay en otro campo del segmento dos entonces decimos que existe un CROSS-REFRENCE. Entonces gráficamente se veria:

Ejemplo:

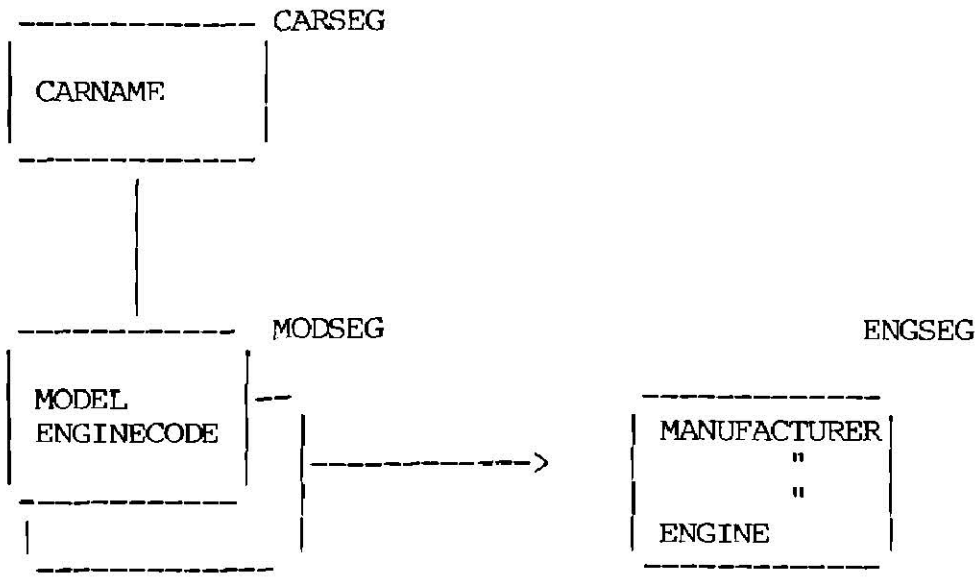
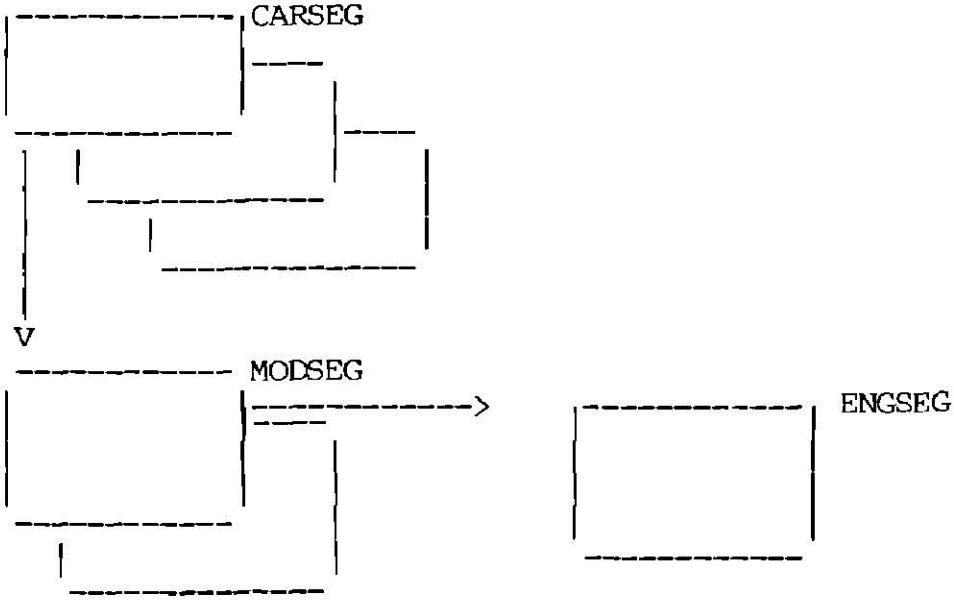



Ilustración de datos

Otro tipo de diagrama usado para ilustar datos puede actualmente ocurrir en un archivo es el "INSTANCE" diagrama.

Ejemplo: Diagrama de un Archivo



Creando Descripciones de Archivos FOCUS

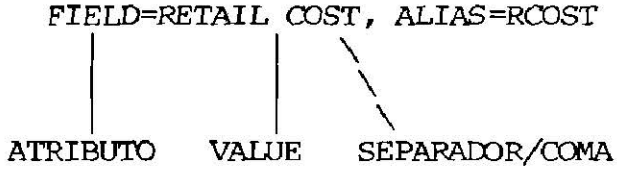
En PC/FOCUS:

Las descripciones de los archivos FOCUS son puestos en archivos que tienen una extension de "MAS". Por ejemplo, un archivo FOCUS nombrado FCARS tiene una descripción del master, almacenando en un archivo nombrado FCARS.MAS.

El PC/FOCUS edita pantallas llenas con TED ú otros editores que pueden ser creadas y modificadas.

Los atributos en la cual comprende la descripción fuente son teclados en formato libre, separado con comas entre cada atributo.

Por ejemplo, una linea de descripción original es teclado como:




Mencionado Convenciones:

Estas son tres clases de atributos usadas para describir un archivo FOCUS.

Estos son:

Atributos Archivos
Atributos Segmento
Atributos Campo.

Mencionando Convenciones (CONT'D)

Los atributos describen el archivo, segmento y campos en un archivo FOCUS puede ser escrito uno despues del otro en una forma libre delimitado por una coma. Al final de la descripción de cada campo se escribe tambien una coma y despues un signo de dollar "$" esto significa el fin de la descripcion de un campo.

Por ejemplo:

```
FIELDNAME= DEALER-COST, ALIAS=DCOST, USAGE=D8,$
FIELDNAME= RETAIL-COST, ALIAS=RCOST, USAGE=D8,$
```

Las reglas para datos de formato libre cuando empezamos a escribir la descripción del archivo son:

-Los nombres de campos y valores de campos pueden empezar en cualquier línea.
-Los espacios en blanco en frente en cualquier campo o valor son ignorados.
-Separados entre atributos con las comas.
-Los nombres de atributos pueden identificarse con su nombre o con su alias.

Por ejemplo:

FIELDNAME ó FIELD

-Al final de cada descripción del campo se coloca una coma (,) y un signo de dollar ($).

Ejemplo:

Atributo Archivo

```
FILENAME=FCAR, SUFFIX=FOC, $
```

Atributo Segmento:

```
SEGNAME = ORIGIN, SEGTYPE=S, $
```

Atributo Campo

```
FIELD= CARNAME, ALIAS=CAR,USAGE=A18, $
FIELD= CARCODE, ALIAS=CCODE, USAGE=A4, $
```

Verificando una Descripción de Archivo

Es fácil corregir una descripción de archivo usando la facilidad del sistema de edición, pero es mas dificultoso determinar si tiene algun tipo de error, o violación de reglas.

En FOCUS tenemos el siguiente comando para checar si tiene algun error una descripción de archivo.

CHECK FILE Filename [PICTURE] [HOLD] [RETRIEVE]

Los efectos de las opciones son las siguientes:

| | |
|---|---|
| PICTURE | Es un diqrama extructural |
| HOLD | Produce un archivo HOLD. Usando el comando TABLE se prepara un reporte basado en el archivo de datos HOLD. |
| RETRIEVE | Trae un diagrama como vistas por los comandos TABLE durante el proceso de la traida de datos. Note los segmentos únicos son vistos con extensiones llgicas de los segmentos padres. |

EJEMPLO:

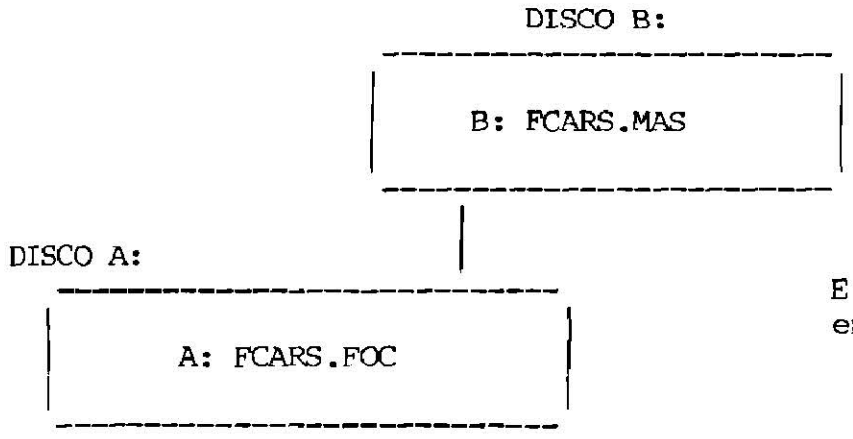
```
    CHECK FILE CARS PICTURE

>>>
    Número de errores = 0
    Número de segmentos = 7 campos = 20 index=0
```

Cambiando un Archivo Maestro

Cada dato ha sido dado de alta en el archivo y no es posible hacer cambios arbitrarios en el MAESTRO. Algunos cambios son eternamente fáciles y pueden hacerse en cualquier tiempo, otros son prohibidos, ya que pueden afectar a los datos, ya dados de alta, entonces para esto es necesario utilizar el comando REBUILD. Otros pueden hacerse si los cambios correspondientes son hechos en varios lugares.

Almacenando Archivos FOCUS y Descripción de Archivos

Generalmente la descripción de un archivo FOCUS y los datos, se encuentran en el mismo disco, pero las decripciones no tienen que estar en el mismo disco que los datos.




Ejemplo de Descripción y Datos en diferentes discos.

Cuando los datos son dados de alta por un nuevo archivo estos son almacenados en un archivo DOS el cual lleva por default el tipo de archivo FOC note que en el ejemplo anterior que el archivo nombrado FCARS, ésta descripción está en FCARS.MAS y los datos estan en FCARS.FOC

Cuando estas extensiones no son cambiadas, entonces esta acción está muy lejana, se requerirá influencia del usuario, para que estas extensiones sean cambiadas. FOCUS puede dar de alta y preparar reportes o cambiar datos fuera de cualquier FILEDEF, u otras comunicaciones.

Cuando el tipo de archivo es renombrado, los datos no se encuentran en el disco, entonces el comando FOCUS (USE) prvee el nombre del archivo de datos para el usuario.

```
>> USE
>> B:FCARS.FOC
>> END
```

Esto quiere decir que con el comando USE nosotros haremos referencia al disco en el cual se encuentra el archivo FOC.

Archivo Fuente

Los archivos Maestros no tiene que estar disponibles en su forma "fuente". La administración de la base de datos puede ser almacenada en uno o mas directorios FOCUS. Para el directorio central, las descripciones son disponibles por todos los propósitos, pero no pueden ser directamente accesados los datos por los usuarios.

ESTRUCTURAS DE LOS ARCHIVOS

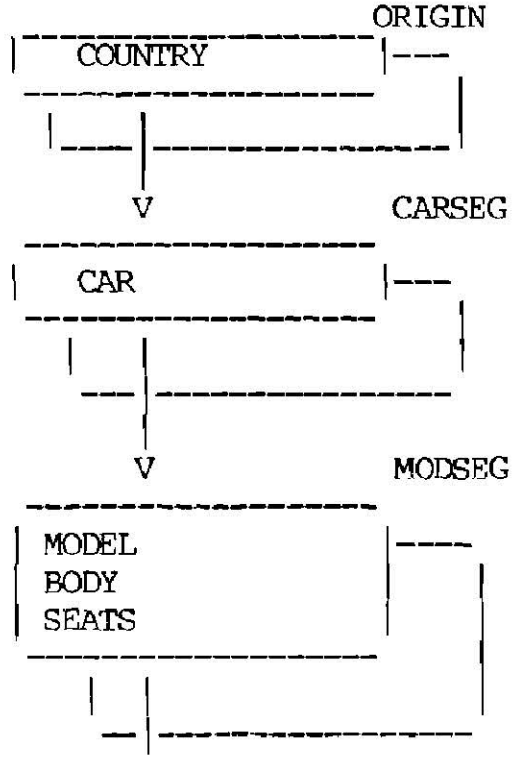
Estructura de un archivo con trayectoria simple

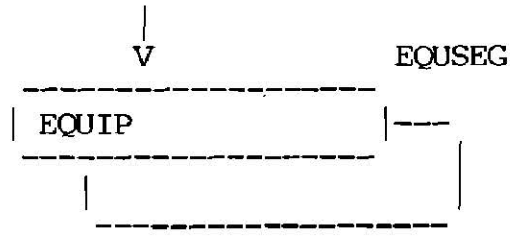
Una de las mas comunes formas de organización de una colección de datos esta dentro de una estructura jerárquica. El nivel mas alto tiene información en común a todos los niveles mas bajos.

EJEMPLO:

| CAMPO | SIGNIFICADO |
|---|---|
| COUNTRY | Ciudad de Origen |
| CAR | Nombre del Carro |
| MODEL | Nombre del modelo del carro |
| BODY | Tipo del Modelo |
| SEATS | Número de asientos en el modelo |
| EQUIP | Modelo del Equipo |

Uno de los métodos de organización de estos, pueden ser cuatro niveles jerárquicos. Un diagrama de este archivo puede ser:






Esto es llamado una trayectoria jerárquica, porque cada segmento tiene solamente un tipo de descendiente. En el diagrama anterior esta solamente una trayectoria de principio a fin.

CARACTERISTICAS DE LOS ARCHIVOS

Las características necesarias que debe llevar un archivo FOCUS son:

| Nombre | Alias | Longuitud | Significado |
|---|---|---|---|
| FILENAME | FILE | 1-8 CARAC. | Nombre del archivo FOCUS |
| FILESUFFIX | SUFFIX | 1-3 CARAC. | extension "FOC". |

CARACTERISTICAS DE LOS SEGMENTOS

Tres piezas de información son usadas para describir cada segmento en un archivo jerárquico. Estos son:

| NOMBRE | ALIAS | LONGUITUD | SIGNIFICADO |
|---|---|---|---|
| SEGNAME | SEGNAME | 1-8 CARAC. | Nombre del segmento. |
| PARENT | PARENT | 1-8 CARAC. | Nombre del Pariente del Segmento. |
| SEGTYPE | SEGTYPE | 1-4 CARAC. | Tipo de Segmento Secuenciado. |

CARACTERISTICA DE CAMPOS

Tres piezas de información son necesarias, acerca de cada campo de datos de la organización del archivo, estos son:

| NOMBRE | ALIAS | LONGUITUD | SIGNIFICADO |
|---|---|---|---|
| FIELDNAME | FIELD | 1-12 CARAC. | Nombre asignado a c/campo de datos. |
| ALIAS | SINONIMO | 1-12 CARAC. | Nombre para identificar c/campo./ |
| USAGE | FORMATO | 2-8 CARAC. | Tipo, Longuitud, y Opción a eitar los valores de datos. |

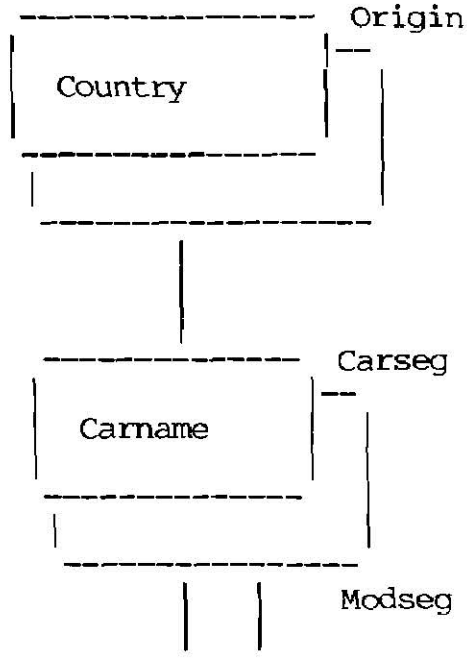
El archivo FCARS puede ser escrito en FCARS MASTER como:

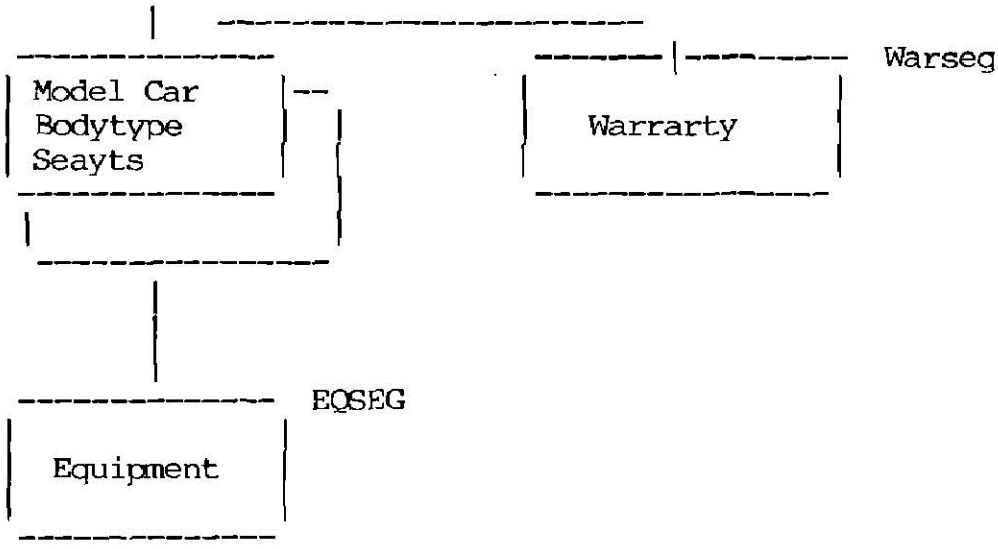
EJEMPLO: Trayectoria simple de un archivo jerárquico.

```
FILENAME=FCAR, SUFFIX=FOC, $
SEGNAME=ORIGIN, SEGTYPE=S, $
FIELD=COUNTRY, ALIAS=CTY, USAGE= A20, $
SEGNAME=CARSEG, SEGTYPE=S, PARENT= ORIGIN, $
FIELD=CARNAME, ALIAS=CAR, USAGE= A14, $
SEGNAME= MODSEG, SEGTYPE= S2, PARENT=CARSEG, $
FIELD=MDEL, ALIAS= MODEL, USAGE= A24, $
FIELD=BODYTYPE, ALIASD=BODY, USAGE= A9,$
FIELD=SEATS, ALIAS=STS, USAGE=13, $
SEGNAME=EQSEG, SEGTYPE=S, PARENT= MODSEG, $
FIELD=EQUIP, ALIAS=ECODE, USAGE= A6,$
```

Estructura de un Archivo con Multiples-Trayectorias

Una extensión de una estructura de una trayectoris simple, (La cual tiene solamente una raiz (TOP-TO-BOTTOM). Esto es sumando a la alternativa del recorrido TOP-TO-BOTTOM. Esto ocurre cuando un registro de un segmento tiene mas que un tipo de descendiente. Por ejemplo suponemos que la informacion (sobre la garantia) es común a todos los carros de una compañia dada la estructura aparece como :






Este tipo de estructura es descrita con la condicion de que

```
SEGNAME=MODSEG, PARENT=CARSEG, SEGTYPE=S, $
   "        "        "      "          "
   "        "        "      "          "
   "        "        "      "          "
SEGNAME=WARSEG, PARENT=CARSEG, SEGTYPE= , $
FEILD=WARRARTY, ALIAS= WAR, USAGA= AGO, $
```

NOTA

NOTA:que ambos MODSEG y WARSEG especifican el mismo pariente del CARSEG. En este caso son 5 segmentos en 4 niveles de la jerarquia. Un archivo puede tener 64 diferentes segmentos.

Segmento Unico

Una estructura Multi-Recorridos es alguna veces contruida por una coleccion de datos la cual están todos directamente relacionados y pueden estar en un simple recorrido. Esto ocurre por ejemplo. Cuando un segmento eta dividido entre dos ó mas partes porque:

1.-Datos para esto son raramente suplidos, hasta no hay espacio de almacaenaje para ser tomados sin ellos están en un segmento separado.

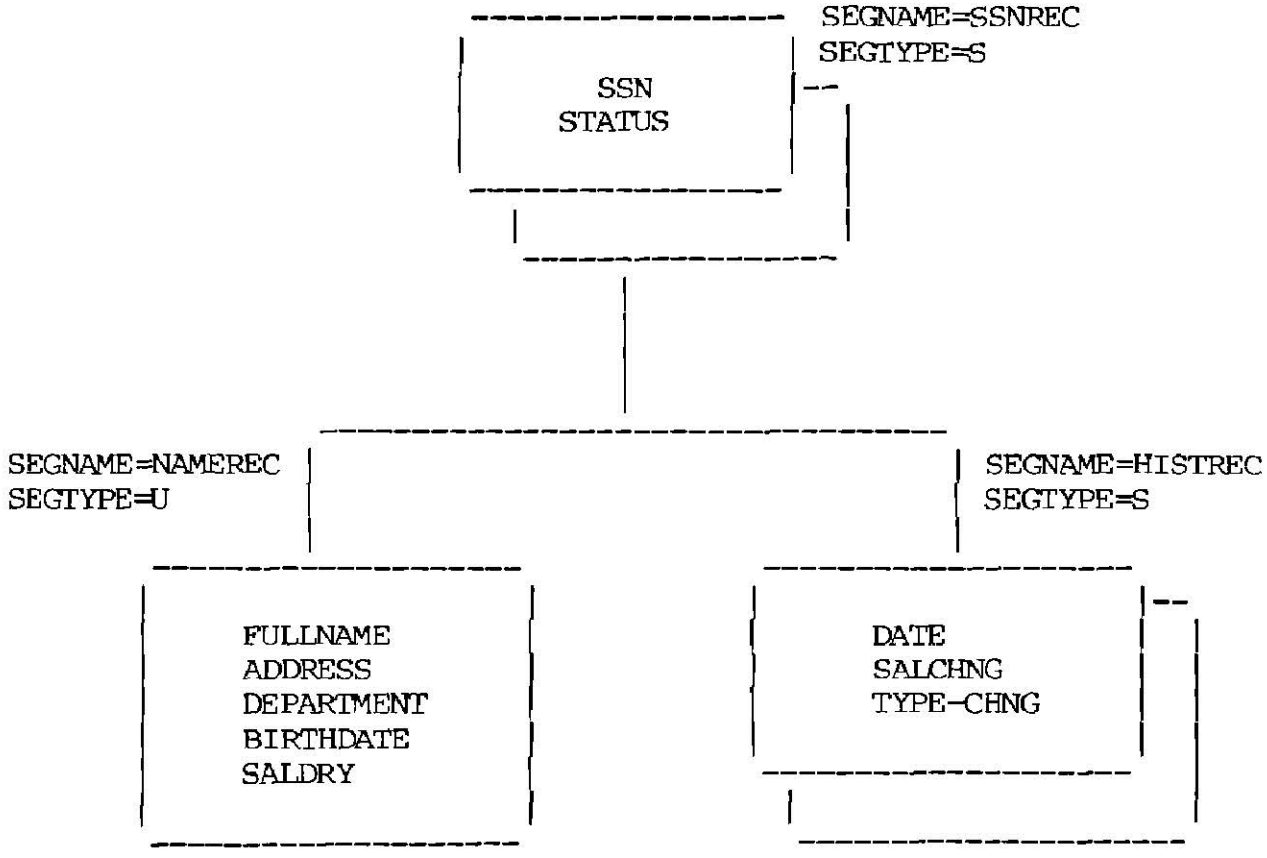
2.-Un segmento pequeño para el coampo "Llave" significa que este puede estar apartado para la misma pagina de almacaenaje, hasta en casos donde una busqueda es necesaria (The Retrievel is Faster)(La recuparacion es rapida).

Esta situacion donde un segmento es dividido entre dos ó mas partes, todavia puede ser tratado logicamente si todos ellos como si estuvieran juntos, requiere que las partes divididas sea identificadas. Para estos segmentos el atributo SEGTYPE esta dado por el valor de "U", (Para segmentos únicos).

Un segmento único es uno del cual puede solamente ocurrir cada vez que ocurra un padre. Esto es uno a uno con este padre. Cualquiera que intente almacenar dos ó mas registros para el mismo padre.

Originara un error

Un ejemplo de usar un segmento "U", supongamos en un archivo de personal. El número de Seguro Social. se encuentra en el nivel uno; y toda la informacion relacionada con el (SSN) está en el segmento descendente a él; tal como FULLNAME, ADDRESS, DEPARTMENT, ETC. Otro segmento con múltiples ocurrencias de Historia/Trabajo está en el segmento descendente a él (SSN).



Nótese en la ilustración que el HISTREC ocurrió múltiples veces pero no fue hecho un segmento descendente a él.

Es un error indicar un segmento descendente al segmento único. Porque es una propiedad especial el segmento único.

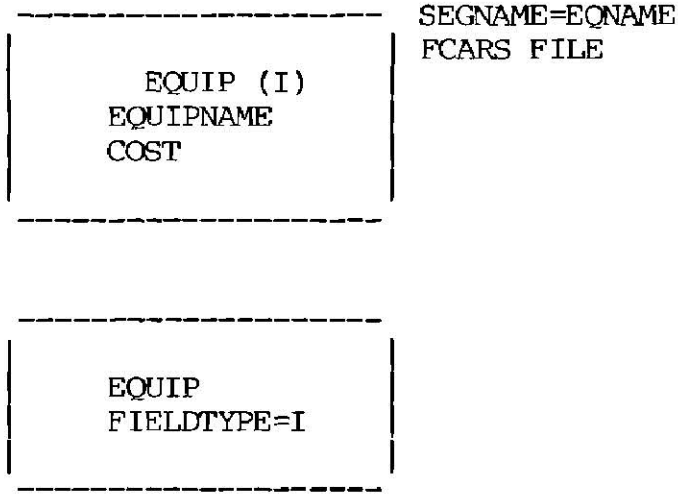
Los segmentos únicos prohiben:

-Almacenar un espacio para datos en la caul no estaran presentes.
-Control automático de múltiples ocurrencias.

Preparando un segmento por partes: FIELDTYPE=I

Un orden para un segmento que hace una CROSS-REFERENCE para uno ó mas campos tiene un FIELDTYPE=I. Esto significoa que un índice ha sido construido para los valores del campo. Através del uso de éste índice otros archivos pueden localizar y usar un segmento. Cualquier número de campos pueden estar indexados en un segmento. Solamente si estos tienen valores en común en otros archivos, estos son candidatos prácticos.

Por ejemplo, en el archivo de datos de equipo llamado STAND, los campos en el primer segmento son:




En el archivo FCARS podemos localizar el valor existente, del nombre del Campo EQUIP indicado como campo indicando en el archivo STAND por el campo nombrado tambien EQUIP. Cuando el tipo de CROSS-REFERENCE es "KU" (Keyed unique) significa que solamente un valor existente puede ser encontrado y aun segmento correspondiente.

CROSS-REFERENCE. Dinámico-Usando el comando "JOIN"

Los mecanismos para juntar estructuras de archivs separados en la cual tienen campos de datos en común pueden ser acopladas a ser ejecutadas en cualquier tiempo para propósito de reportes. El uso de el método esquema Master comprendo mostrar la relación de segmento referenciado. Un comando en FOCUS es usado para juntar estructura de archivos separados la cual éste comando nos da muchas utilerías.

La Sintáxis es:

JOIN field1 IN file1 TO [ALL] field2 IN file2 AS joinname

Donde:

field1: es cualquier campo en el file1
field2: es un campo indexado del file2 la cual tiene el mismo formato que el field1 (FIELDTYPE=I).

Joinname: es cualquier nombre arbitrario de 8 caracteres que identifica al estatuto join.
ALL: es usado cuando son mas que un posible valor existente del field2 en el file2. Esto es equivalente en FOCUS a SEGTYPE=KM (Llaves multiples) I.E.

La combinación de 4 campos pueden ser usados en lugar del Field1 I.E.

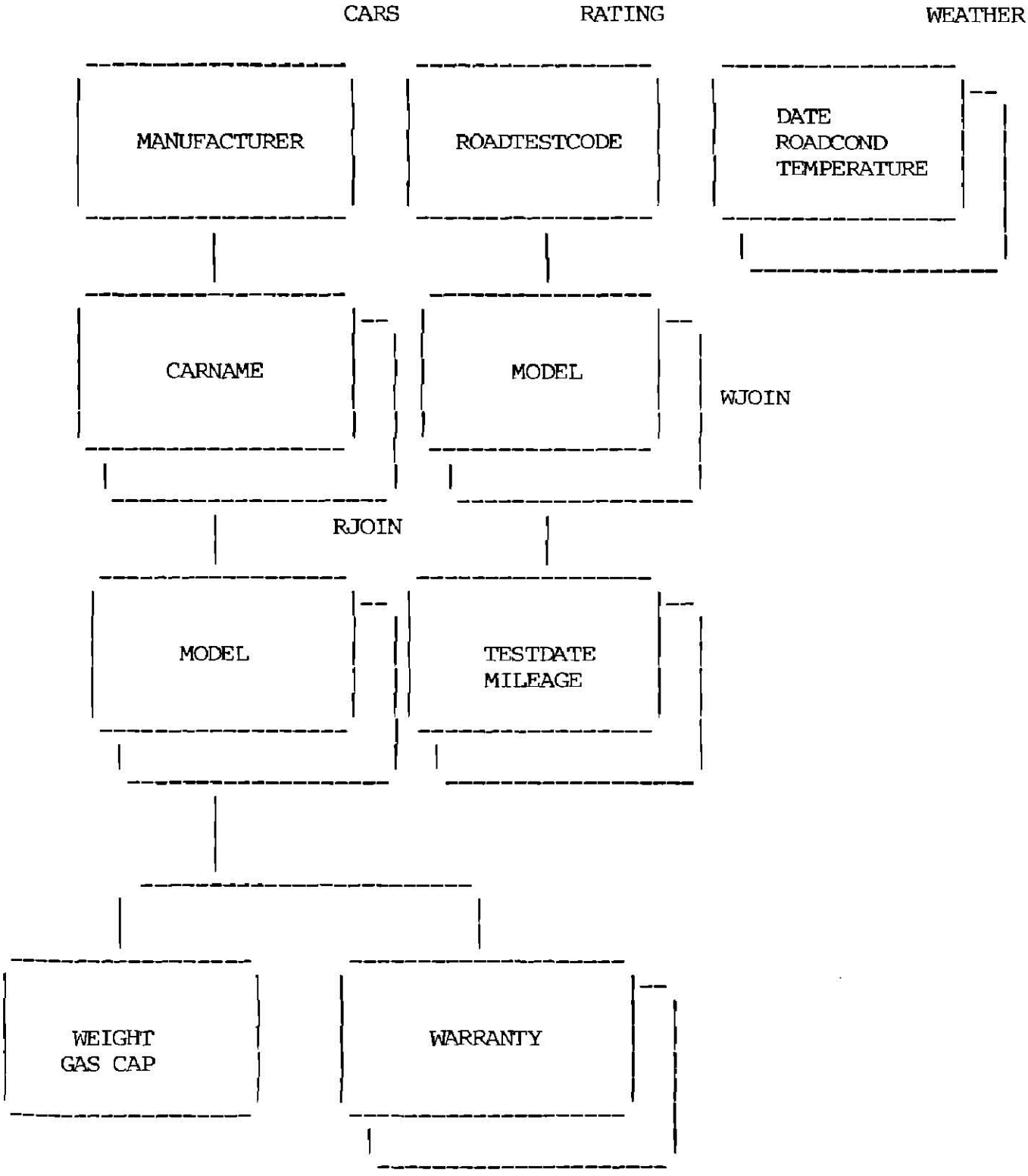
JOIN FIELD1 [Field2...] IN file1 to [ALL] field2 in file2 AS joinname.

En este caso el grupo de campos en el primer archivo es combinado y los datos concatenados son existentes en un campo en el segundo archivo. Cuatro campos pueden ser combinados, y a la vez la combinacion de todos esta disponible.

Ejemplo:

Join model in cars to all model in raiting as rsoin
Join testdate in cars to date in weather as wjoin.

El efecto de este join es combinar una entrada en una de las estructuras, en otra estructura. Gráficamente


CARS
RATING
WEATHER
MANUFACTURER
ROADTESTCODE
DATE
ROADCOND
TEMPERATURE
CARNAME
MODEL
WJOIN
RJOIN
MODEL
TESTDATE
MILEAGE
WEIGHT
GAS CAP
WARRANTY

Para desplayar la lista de Join activos actuales use el siguiente comando FOCUS:

```
>>? JOIN
```

Para poder desactivar dichos Join es necesario utilizar el siguiente comando:

```
>>JOIN CLEAR (Joinname)
```

Nota: Se puede utilizar hasta 16 Joins para cualquier combinacion de archivos en la implementacion actual.

Los Join son activos durante un Modify estando disponibles el FIND y LOOKUP. Un diagrama donde represente el Join actual, se puede obtener:

1.-Usando el comando CHECKFILE, FILENAME, PICTURE.
2.-Despues de un Join el archivo unido obtiene el nombre del primer archivo (file1) ya sea en un DEFINE, TABLEF, TABLE, ETC.)
3.-El comando Join puede ser teclado en una sola linea.
4.-Si varios campos son indexados por un Join, entonces el comando ?Join displayara solamente el primero de ellos.

Ventajas:

-No tiene que ser preplaneado.
-No ocupa espacio de archivo.
-Puede ser agregado o cancelado como se necesite.

Desventajas

-El comando Join puede ser usado en cada sesión.

Identificando archivos Fisicos y Lógicos, el comando USE

En FOCUS es muy importante que los comandos hagan referencia al nombre de un archivo.

Ej.-

```
TABLE FILE FILENAME
CHECK FILE FILENAME
```

Cuando estos comandos son usados FOCUS localiza la descripción del master, con el nombre al cual hacen referencia, entonces localizar los datos fisicos asociados con dicho archivo, FOCUS hace una serie de Assuptions por Default.

Funciones del comando USE

El comando USE es necesario cuando los datos no residen en el archivo o DDNAME ó si la información adicional se requiere ser especificada como puede ser usada. Algunas de las siguientes funciones son relevanates en el sistema operativo MS/DOS.

Las funciones del comando:

1.-Cambia el DEFAULT la extensión del archivo o letra del disco.
2.-Identifica un archivo de datos que tiene diferentes nombre.
3.-Identifica un archivo de datos que no existen todavia pero puede ser creado, en la cual no tiene un nombre por default.
4.-Protege cambios de un archivo de datos por un Modify ó comando SCAN.
5.-Concatena varios archivos fisicos dentro de un archivo lógico para propósito de reportes.
6.-Identifica la localización de un archivo que está dentro del control central de una base de datos.

Sintáxis del comando USE

```
USE [ADD]
D:FILENAME TYPE [READ NEW] [AS FILENAME] [ON CONTROLID]
     "
END
```

or

```
USE CLEAR.
```

Si el comando USE está usando los parametros ADD ó CLEAR, el listado de archivos remplaza cualquier previo USE listado. Si el parámetro ADD es suplido. Las nuevas entradas son apendiadas a la lista vieja. Si el atributo Clear es usado, todos los previos USE comand son borrados.

Cambianado el defult FILETYPE ó FILEMODE

En MS-DOS, el primer archivo identificado por un comando USE define la extensión del archivo y letra del disco para todos los comandos subsecuentes, hasta el fin deuna sesión ó hasta el USE comand son barrada.

Ej.-

```
USE
F:STAND, FOC
END
TABLE FILE STAND
```

FOCUS localiza un archivo de datos llamado F:STAND.FOC, en lugar de A:STAD,FOC.

Identificando un archivo con un nombre diferente.

Si un archivo no tiene el filename ó ddname que FOCUS indica, un USE command es necesario.

Por ej.-

```
USE
A:CARS:STP AS SPORT
END
```

El comando TABLE FILE SPORT puede ser usado a accesarse como el archivo de datos CARS usando una definición masters, SPORT.

Identificando un archivo de datos

En MS-DOS, FOCUS automáticamente crea un nuevo archivo, si el comando MODIFY FILE FILENAME es usado y si el archivo esta dado con un nombre diferente del especificado es necesario un USE comand en el atributo """NEM".

Por ej.-

```
USE
C:CARS.FOC NEW
END
```

Protegiendo un archivo de cambios.

Es posible proteer un archivo de cambios usando el atributo "READ" en un USE command.Por ej.-

```
USE
C:CARS.FOC READ
END
```

El archivo puede ser referenciado en un MODIFY ó SCAN, para cualquier opción de incluir, borrar o actualizar registros.

El ? estatuto USE

El comando ?USE causa un listado actual a ser desplegado.

Ej.-

```
?USED
FILE IN USE ARE
A:FCARS.FOC
B:EQUIP.FOC
```

Nueva base datos

En MS-DOS la primera prueba a poner dentro de un archivo usa un tipo default y un disco A.FOC (A menos que el camando USE haya sido cambiado por default)

El comando FOCUS "CREATE" como:

```
CREATE FILE FILENAME
```

Sin ningun archivo esta presente actualmente, focus dinamicamente asigna espacio en disco dentro del DDNAME especificado.

Administración de la base de datos

Introducción:

Los accesos a los datos en un archivo FOCUS puede ser restringidos. Estas restricciones dividen al usuario dentro de 2 categorías. Estas son:

-Leer el archivo solamente.
-Cambiar datos en el archivos solamente.

Usarios quienes son permitidos leer solamente pueden ser limitados por:

-No está disponible a leer todos los valores campo.
-No está disponible a leer entrada de segmentos de datos.
-No está disponible a leer cada ocurrencia de datos de estos campos y segmentos estas son permitidas a accesarse.

Estos limites son llamados restricciones de FIELD SEGMENT y VALUE y asi tambien existen restricciones para PROGRAM que son:

-No cambiar datos en 1 ó mas campos.
-No cambiar datos en cualquier campo en 1 ó mas segmentos.
-No accesa porciones seleccionadas de un archivo.
-No suma o borra segmentos, puede cambiar valores existentes. (Up date only access)

Estableciendo identidad de usuarios

Identificación de usuarios ó proveer un valor de clave de ecceso. La sintaxis del comando SET es:

```
SET PASS=NAME
```

ó

```
SET USER=NAME
```

entonces:

```
SET PASS=TOM362.
```

El comando SET puede ser usado a cualquier tiempo antes de accesarse el archivo ocurrido.

Cuando el usuario no tiene una llave o no tiene una la cual es inadecuada a proveer el tipo de acceso entonces el mensaje desplegado es:

(FOC 648)User does not have access rights to file filename.
(El usuario no tiene acceso al archivo, nombre del archivo)

Accesos de Niveles necesarios, ACCESS=

Estas son cuatro nieveles de acceso. Estas son:

ACCESS=R lee solamente
ACCESS=W escribe solamente
ACCESS=RW lee y escribe solamente
ACCESS=U actualiza solamente.

Los archivos FOCUS estan protegidos del tipo de acceso determinado por los comandos de arriba.

Los tipos de accesos necesarios en orden de procedencia con varios comandos FOCUS son mostrados en la siguiente tabla:

Tipo de accesos necesarios

| COMANDOS | R | W | R/W | U |
|---|---|---|---|---|
| TABLE | X | | X | |
| MODIFY | | X | X | X |
| SCAN | | | X | X |
| DEFINE | X | X | X | X |

ACCESS=U (Actualiza solamente)

En el modo de acceso del SCAN de "U" todos los comandos excepto DELETE,IMPUT and MOVE, hasta, REPLACE y CHANGE estan disponibles y los valores de la base de datos pueden ser cambiados.

Tipo de restricciones de cadaa valor no son activadas por operaciones SCAN, como son las restricciones campos y segmentos son activadas. Note que el acceso de "R" no se puede usar SCAN, a menos de que los usuarios tengan acceso "U".

Un usuario quien "U" puede tener acceso "R". Cualquier otra combinación de "U" no es efectiva por lo tanto:

```
USER=WALL, ACCESS=U, RESTRICT=FIELD, NAME=SALARY, $
           ACCESS=R, $
```

Dentro del modo de acceso del Modify no se permiten las operaciones INCLUDE y Delete cuando el acceso "U" solamente se puede actualizar.

ACCESS=R (Lee solamente)

Un usuario quien tiene acceso "R" puede preparaar reportes de un archivo. esto es, los comandos TABLE, TABLEF, DEFINE, etc. NO todos los comandos con los valor pueden modificar el contenido de un archiv son permitidos.

Ej.-

1.-Lee todos los campos y segmentos de datos.
```
    USER=SALLY, ACCESS=R, $
```
2.-Lee todos los campos excepto 2.
```
    USER=BILL, ACCESS=R, RESTRICT=RIELD, NAME=SALARY, $
```
3.-Lee los datos pasados por una NAME=DECREASE, $

```
USER=JOHN, ACCESS=R, RESTRICT=VALUE, NAME=IDSEG
VALLUE=CIVISION EQ "EAST" or "WEST", $
```

ACCESS=W (Escribe solamente)

Un usuario quien tiene acceso "W" puede solamente usar el comando MODIFY, nuevos datos pueden ser sumados a un archivo y viejos pueden ser actualizados, las restricciones de FIELD y SEGMENT son opcionales.

| ATRIBUTOS | REFERENCIAS |
|---|---|
| Alias | Se puede usar de 1 a 12 caracteres como un hombre alternado a identifado el campo. Se recomienda qué en el nombre no se ultiliza blancos ó caracteres especiales. |
| Crfile | El nombre del archivo referenciado en la cual el segmento KU ó KM referenciado está descrito. |
| Crkey | Cuando SEGTYPE=KU (llave única) ó KM (múltiple). |
| Crsegname | El nombre de un segmento en un archivo referenciado si es diferente del SEGNAME. |
| Definition | De 1 a 44 caracteres usada a documentarse el significado del campo. |
| Encrypt | Cuando ENCRYPT=ON todos los campos en el segmento son almacenados en una forma revuelta. Dado con un paquete de administración de datos. |
| Fieldname | De 1 a 12 caracteres a identificar un campo. El nombre puede incluir espacios en blanco. |
| Location | El nombre del archivo con un segmento de datos está localizado. |
| Parent | El nombre de los segmentos parientes. |

Mantenimiento de Archivos FOCUS

CONCEPTOS GENERALES

Para que cambie la información en una base de datos FOCUS se tiene que suministrar una nueva información ya sea para:

* Cambiar los valores de artículos existentes en registros existentes
* Agregar nuevos registros de información
* Suprimir registros antiguos

La nueva información es llamada "transacción". Estas transacciones puedan traducirse de varias formas. Estas pueden ser:

* Escritas directamente en una terminal de comunicación dentro de una base de datos.
* Almacenados en un archivo de disco temporal o en una cinta de archivo, para más tarde introducirlo en la base de datos.
* Introducido en respuesta a una interacción rápida de información.
* Introducido en una terminal CRT en una pantalla.

Independientemente de como entren la transacciones, existen 8 elementos de información sobre ellos quedeben de conocerse.

El FOCUS proporciona un lenguaje facil de usar para describir estos 8 elementos. El lenguaje sustituye comisiones cuando no se proporciona información por lo tanto reduciéndose a los ususarios un mínimo de esfuerzos. El propósito del lenguaje del procesamiento del lenguaje es el de reemplazar la necesidad de escribir un programa de computación para modificar la información en una base de datos FOCUS.

El procesador de transacción es introducirlo escribiendo el comando FOCUS de MODIFY seguido por el nombre del archivo a ser cambiado.
Por ejemplo, MODIFY FILE CARS

El sistema entonces responde escribiendo: ENTER SUBCOMMANDS:
Existen 18 subcomandos los cuales están en disposición para lograr las 8 tareas descriptivas sobre transacciones. En la mayoría de los casos solo algunos de estos subcomandos son necesitados las tareas y los subcomandos son:

Tarea 1 describe el formato físico de los campos en una transacción
FIXFORM (FORM)
subcomandos: FREEFORM
PROMPT
CRTFORM

Tarea 2 Describe el criterio de validez para aceptar una transacci0on
subcomando: VALIDATE.

Tarea 3 Describe cálculos o modificaciones por hacerse a los datos en la transacción
subcomando: COMPUTE.

Tarea 4 Describe como la transacción concuerda los registros de base de datos.
Subcomandos: MATCH o NEXT
ON NEXT ON NONEXT

Subcomandos:
ON MATCH {ACTION} ON NOMATCH {ACTION}

Tarea 5 Describe la acción que ocurrirá cuando los registros de base de datos.
subcomandos:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | VALIDATE | INCLUDE | | INCLUDE | PROMPT |
| | UPDATE | TYPE | EN | REJECT | CRT FORM |
| EN MATCH | REJECT | PROMPT | NOMATCH | COMPUTE | GOTO |
| | DELETE | CRTFORM | | VALIOTE | FIXFORM |
| | COMPUTE | GOTO | | TYPE | |
| | IF | FIXFORM | | IF | |
| | GOTO | | | | |

| | | | | |
|---|---|---|---|---|
| COMPUTE | PROMPT | COMPUTE | PROMT |
| VALIATE | CRTFORM | TYPE | FIXFORM |
| UPDATE | FIXFORM | GOTO | REJECT |
| REJECT | TYPE | IF | |
| IF | | CRTFORM | |
| GOTO | | | |

Tarea 6 Describe como los errores serán tratados y cualquier información de auditoria para transacciones aceptadas.
Subcomando CHECK ON INVALID TYPE
LOG ON INVALID GOTO
TYPE

Tarea 7 Describir de donde proviene la fuente de transacciones
subcomando: START N
STOP N
DATA [{ON VIA} DD NOMBRE]

Tarea 8 Usa los otros subcomandos para construir casos de procesamiento separados.
subcomando. CASE
ENACASE
IF (expresión) GOTO
GOTO

Por ejemplo:

```
ON MATCH UPDATE  PESO ALTURA COMBUSTIBLE BHP
ON MATCH UPDATE GALONES ANCHO RPN
```

Hay solo dos excepciones a esto cálculos que se siguen uno a otro exactamente como en el comando define e.g.,

```
ON NOMATCH VALIDATE
GOODNUM= INVNUM LE 999 AND INVNUM NE 800;
GOODCOST=RCOST GT D COST;
```

Y con teextyos libres en el subcomando TYPE el cual especifica el texto exactamente como en table HEADINGS, e, g.,

```
" L10 COSTO DE <RCOST IS IN ERROR"
" L30 RCENTER VALUE "
" L20 OR CHECK COSTO AGENCIA"
```

Las transacciones que deben modificar los archivos de datos pueden proporcionar de archivos de disco o cintas o pueden ser escritas directamente en la terminal.

*Transacciones en discos o cintas:*

Cuando las transacciones vienen de archivos de disco o cinta el procedimiento es terminado por la palabra END en una línea por si sola.

*Ejemplo: Transacciones en un archivo de disco.*

```
MODIFY FILE CARS
FIXFORM COUNTRY/10  X2 CAR/10 MODEL/20
RCOST/8
MATCH COUNTRY CAR MODEL
ON MATCH UPDATE RCOST
ON NOMATCH REJECT
DATA ON NEWCOSTS
END   transactions typed "live"
```

Transacciones escritas en "vivo".

Cuando se escriva en "vivo" las transacciones en una terminal por el operador, la palabra de subcomando DATA es el último subcomando procesado. La terminal entonces se abre para recibir datos y el operador escribe las transcciones (por lo general es un formato libre o por medio de una asistencia) y la palabra END sobre una línea se usa para darle terminación al procedimiento después de la última transacción haya sido escrita.

Ejemplo: Transacciones escritas en "vivo" en un formato libre"

```
MODIFY FILE CARS
MATCH COUNTRY CAR MKODEL BODY
ON NOMATCH INCLUDE
ON MATCH REJECT
DATA
COUNTRY= ITALY, CAR=FIAT, MODEL=L128 2
DOOR,
BODY= COUPE, SEATS=5, RCOST=4150,$
MODEL=128 4 DOOR, BODY=SEDAN,SEATS=
RCOST=4355, $
END
```

Transacciones entrados en una pantalla completa CRT:

Cuando se usa la pantalla CRT el subcomando MODIFY CRTFORM describe el formato de la pantalla. El tipo de CRT es por omisión una IBM 3270, puede ser anulado colocando las palabras DATA VIA crtnombre en la última línea del procedimiento. Una elección separada describe el lenguaje interactivo de entrada de datos (FIDEL).

Proceso de mantenimiento de archivo básico.

Los tres procesos de mantenimiento de archivo básicos son:

* Entrada de nuevos segmentos existentes
* Cambio de valores en segmentos existentes.
* Suapensión de segmentos existentes.

Estos pueden combinarse en un procedimiento el cual realiza procesos compuestos. El ejemplo siguiente ilustra cada uno de los procesos básicos.

Ejemplo: Entrada de segmentos nuevos.

```
CMS FILEDEF NOGOOD DISK REJECT DATA (LRECL 80 RECFM F)
MODIFY FILE CARS
FIXFORM COUNTRY/10  X2 CAR/10  MODEL/20
FIXFORM BODY/5  SEATS/3  X4  RCOST/6  DCOST/6
MATCH COUNTRY CAR MODEL BODY
ON NOMATCH INCLUDE
ON MATCH REJECT
LOG DUPL ON NOGOOD
DATA ONB NEWIN
END
```

Ejemplo: Actualizando los valores de datos existentes.

```
MODIFY FILE CARS
VALIDATE
SEATEST=(SEATS EQ 2 AND BODY EQ COUPE
FIXFORM CAR/ 10 X2 COUNTRY/10 MODEL/20
FIXFORM X4 BODY/5  SEATS/3 RCOST/6
MATCH CAR COUNTRY MODEL BODY
ON MATCH UPDATE SEATS RCOST
ON MATCH REJECT
DATA, ON UPVAL
END
```

Ejemplo: Supresión de segmentos.

```
MODIFY FILE CAR
MATCH COUNTRY
ON MATCH REJECT
ON MATCH CONTINUE
MATCH CAR
ON MATCH REJECT
ON MATCH CONTINUE
MATCH MODEL
ON MATCH DELETE
ON MATCH REJECT
DATA
   .
   .
END
```

Secuencia de subcomandos.

Es de gran utilidad tener en mente que los subcomandos dentro de la cláusula MATCH se ejecutan en la orden provista, como en la secuencia siguiente

```
MATCH CARRO
ON NOMATCH COMPUTE
MPG= MPG + 10;
ON NOMATCH TYPE "NEW CAR WITH MPG OF "<MPG>"
```

El mensaje es escrito después de haber ejecutado el cálculo, entonces el valor en este caso es mayor por 10.

Tarea 1. Describiendo transacciones
Fuentes de datos

Los datos de entrada para los procedimientos MODIFY provienen de uno de cuatro fuentes, cada uno con su propio subcomando. Los subcomandos y las descripciones de los datos son los siguientes:

- FISFORM. Este subcomando especifica los datos de un formato arreglado con cada campo apareciendo en una posición arreglada del registro.
- FREE FORM. Este subcomando especifica registros de formato variable con campos de datos delimitados por comas.
- CRTFORM. Este subcomando especifica que los datos serán proporcionados por el usuario en una pantalla CRT pre-formada (llenar los espacios en blanco.
- PROMPT. Este subcomando especifica que el usuario introducirá datos interactivamente en respuesta a indicaciones del FOCUS.

En adición, existen cinco comandos de utilidad (START, STOP,CHECK, DATA, END) que son usados para apoyar el proceso de entrada de datos.

Describiendo transacciones (formatos arreglados)

Las transacciones de formatos arreglados son aquéllos en los cuales los valores de campo ocupan la misma posición en cada registro de transacción. Para describir el formato arreglado son necesarios dos piezas de información.

- Primero- Los nombres de los campos deben darse
- Segundo- La posición de la columna del campo en el registro.

El subcomando FISFORM es usada para estos propósitos se pueden suministrar tantas líneas del comando FIXFORM como sean necesarias para describir el formato del registro de transacción.

El subcomando FIXFORM

El subcomando FIXFORM es usado para especificar los nombres de los campos de datos y el número de caracteres a ser procesados para cada campo de datos en la transacción.

La sintaxis básica es:

FIXFORM [ON DDNOMBRE] nombre de campo/longitud nombre de campo/longitud.

El nombre de campo es separado por una línea inclinada del número de caracteres para procesar sobre la transacción; y obtener el valor para el campo. Cada set de nombres- longitudes está separado por un espacio del siguiente set de nombres- longitudes.

Por ejemplo: FIXFORM CLIENTE/10 MES/12 UNIDADES/4 CANTIDAD/5

Esto indica que el registro de transacción tiene cuatro campos de datos, el primero de 10 caracteres de largo, el segundo de dos, el tercero de cuatro caracteres, y el último de 5 caracteres. Si se tienen que saltar carateres en el registro de tansacción entonces se usa 'x' seguida del número de caracteres a saltarse.

FIXFORM X3 CLIENTE/10 MES/2 X1 UNIDADES/4

Los primeros 3 caracteres son saltados y después uno es saltado . Es posible saltar caracteres hacia adelante o hacia atrás. Esto permite el uso de solo un campo de transacción como el valor de varios campos de archivos de datos.

Por ejemplo, agregarle a un total de año a la fecha y también ser itroducido como un valor de detalle . Entonces,

FIXFORM YRTODATE/4 X4 UNIDADES/4 CANTIDAD/8

Procesará cuatro caracteres, después regresarse y tomar el mismo valor de nuevo y usarlo como si fuera un párrafo de datos diferentes. Se puede usar una separación parcial de campos de transacción.

Por ejemplo, un prefijo en un campo de datos puede almacenarse en un párrafo diferente, al igual que el párrafo de datos en total, e, g.

FIXFORM PRODUCT/8 X-8 DODIGO/2 X6 FACTOR 5

Este usa los ocho caracteres como un valor de datos, después se regresa al principio del campo y se usa los primeros 2 caracteres, después se berinca sobre los restantes 6 para continuar con los otros valores.

Cualquier número de entradas separada por espacios puede proporcionarse en una línea FIXFORM dada. Si se necesitan más líneas entonces las entradas se siguen una a otra en un flujo de líneas de FIXFORM. Las siguientes dos grupos son identicos en significado.

```
SET 1
FIXFORM  MES/2  CODIGO/3  X40  UNITS/4  FACTOR/6
SET 2
FIXFORM  MES/2  CODIGO/3  X40
FIXFORM UNIDADES/4
FIXFORM FACTOR/6
```

Espacios en blanco insertados:

Si cualquier nombre de campo en el subcomando FIXFORM contiene un espacio insertado, entonces se usa ya sea su nombre de alias alterado o su truncación más corta, encerrar el elemento entre comillas sencillas.

Por ejemplo.

```
FISFORM        RETAIL   COST/9'   MODEL/18
```

comillas sencillas  espacio insertado

Por ejemplo . Actualizando un archivo usando la transacción del formato arreglado

```
MODIFY FILE CARS
FIXFORM COUNTRY/10  CAR/18  'MODEL NAME/20'  TYPE/8
FIXFORM SEATS/2 X10  RCOST/8 DCOST/8
MATCH COUNTRY CAR MODEL TYPE
ON MATCH UPDATE SEATS RCOST DCOST
ON NOMATCH REJECT
LOG NOMATCH ON BADRECS
DATA ON UPFILET
END
```

Valores de datos binarios

Si ya existenb valores de datos en la transacción en un formato interno, como un número entero binario, decimal empacado o una anotación de punto flotante, entonces el largo del subparámetro después del nombre de campo en el subcomando FORM deben citar el tipo de valor de datos.

Los tipos de datos internos son:

| Letra | tipo |
|---|---|
| P | Decimal empacada (longitud en dígitos in packed más signo. |
| I | Entero binario (4 bites) |
| D | Punto flotante de doble precisión (8 bites) |
| F | Punto flotante de precisión sencilla (4 bites) |
| Z | Decimal seccionada |

Ejemplo: Leyendo datos Binarios.

```
FIXFORM  PAIS/10     RCOSTO/P8      ASIENTOS/14    CODIGO/Z5
```

Datos en decimales en packed — datos en un decimal seccionado

Punto decimal implícito

Cuando los datos decimales seccionados (z) o in packed (P) son introducidos, la posición del punto decimal puede ser especificado incluyéndo en la longitud sub-parámetro del descriptor de campo.

Por ejemplo, MPG/26.2 indica un campo de entrada seccionada de seis biles con un valor que contiene 2 lugares decimales.

## Describiendo grupos repetitivos (formatos arreglados)

Una situación común es la frecuencia de los grupos de campos más de una vez en la transacción,

Por ejemplo un registro que contiene un nombre de producto aparece cada 12 meses. Los registros MONTH y AMOUNT aparecen 12 veces en cada ejemplo

Los registos son indicados en el subcomando FORM encerrado los campos en el registro entre paréntesis y colocando el número de frecuencias antes de de los paréntesis abiertos.

Por ejemplo.

```
FIXFORM  PRODUCTO/10  AREA/6  12 (MES/2  X4 CANTIDAD/6
```

Nótese que el largo del registro es.

10+ 6+12x (2+ 4+6) ó 160 caracteres

Los grupos múltiples pueden ser especificados pero no pueden ser colocados en orden.

Ejemplo: Entrada de grupos repetitivos.

```
MODIFY FILE CARS
MATCH COUNTRY CART MODEL
ON NOMATCH INCLUDE
MATCH WARRANTY
ON MATCH INCLUDE
FIXFORM  COUNTRY/10  3  (CAR/10  MODEL/20 ) 2(WARRANTY/30
DATA ON MYNEW
END
```

Note: Number of occurrences

## Describiendo valores de datos que pueden no estar presentes (datos condicionales)

Los valores de los datos que pueden ocurrir en una transacción pero no en una obra, se dice que están condicionalmente presentes. Si ocurren suceden en el imput ( entrada) , o en el proceso de actualización, de otra manera, son ignorados. Las prueba de la ocurrencia es un campo completamente en blanco ( esto sólo ocurre en datos alfanuméricos.

Dos usos importantes son:

1. La especificación de longitud en la línea FORM para cada campo en el grupo es precedido por la letra ~C'.

Ejemplo. Presencia condicional de datos de entrada.

```
FIXFORM PRODCODE/8  X3  12 (MONTH/C2   X1   SALES/C6
```

condicional

Sanoke data might appear as:

```
SP40721X ...01..1465.02...467.03..5148
RV431674 ...01..6400.
```

Los registros de transacción física en el ejemplo pueden tener grupos de 0 a 12 . Cada grupo que es aceptado se cuenta como uno transacción lógica. Entonces los dos registros físicos de arriba serían contados como 3 transacciones.

2. Actualizando una lista variable de campos

La especificación del largo para cada campo que puede no estar presente es precedido por la letra `C' .

Ejemplo. Actualizando un grupo variable de campos.

```
MODIFY FILE CARS
FIXFORM COUNTRY/10 CAR/15 MODEL/15 BODY/8
FIXFORM  MPG/C4  WEIGHT/C5 LENGTH/C5
FIXFORM WBASE/C5
MATCH COUNTRY CAR MODEL BODY
ON MATCH UPDATE MPG WEIGHT LENGTH WBASE
ON NOMATCH REJECT
DATA ON NEWVALS
END
```

En el ejemplo algunas transacciones pueden contener todos los campos para actualizarlos erg, MPG, PESO, LARGO, ETC. Otros pueden contener sólo PESO, O algún ser parcial del total de campos disponibles para UPDATE.

Fuentes de datos FIXFORM (ON dd nombre)

Bajo el uso normal el FIXFORM no especifica las fuentes de las transacciones de datos de entrada. En lugar, el subcomando, el DATA ON ddnombre, es colocado usualmente al final del kprocedimiento.

De manera opcional, se puede especificar la fuente en el enunciaDO fixform E.G.

```
FIXFORM ON INDATA COUNTRY/10 CAR/10
```

Por medio de esta opción un MODIFY po si, puede obtener datos para varias fuentes de transacción .

Describiendo transacciones (formato libre)

En un formato de transacción libre el operador escribe la identidad del campo de datos junto con su valor de datos. No se restringe escribir en las columnas arregladas en una línea, y cada línea puede ser enteramente diferente. Se pueden identificar los campos de datos de 4 maneras:

1. Nombre de campo
2. Nombre sinónimo
3. Posición numérica en un diccionario de archivo de datos.
4. Truncación corta única.

Ejemplo: Métodos equivalentes

Asúmase un campo de datos con los siguiente 5 campos de datos.

| Número | Nombre | Sinónimo |
|---|---|---|
| 1 | PROD-TYPE | PRT |
| 2 | PROD-NAME | PNAME |
| 3 | AREA | AR |
| 4 | CANTIDAD | AMT |
| 5 | FACTOR | FAC |

(a) PROD- TYPE= ALY, AREA= CAMIONES, CANTIDAD= 1000 ,$
(b) PRT= ALY, AR= CAMIONES, AMT = 1000 , $
(c) 1= ALY , 3 CAMIONES, 4 = 1000 ,$
(d) 1= ALY
3= CAMIONES, 1000 , $

Las cuatro formas de arriba son formas equivalentes de escribir las mismas transacción ( asumiendo que PROD-TYPE es el compu 1 en el diccionacio de archivos, etc.)

Nótese que:

(a) Los valores de datos, losk jpares de dos valores de los nombres de campo son separados por comas.
(b) El final de cada transacción es terminada por una coma seguida por un de dólares , `$'. La transacción puede continuar a cualquier número de líneas , y la última línea contiene el terminador.
(c) Una vez mencionado un campo de datos en una transacción no tiene que re-colocarse ( re-set) su valor y el valor antigu persistirá hasta que sea cambiado, haciéndose más fácil de introducirle información repetitiva. Entonces los valores de los datos pueden `traspasarse' de una transacción a otra.
(d) Los traspasos ocurren en campos no mencionados sólo hasta que sea mencionado el primer campo. Después de esto todos los campos no mencionados asumen valores de omisión de cero o espacios. Por ejemplo, si la primera transacción tiene campos con posiciones de archivo número de 1 a 6 , y la segunda transacción contiene campos de 4 y 6 solamente, entonces los valores para 1,2, y 3 se traspasan, pero el campo 5 omite.

## Espacios en blanco iniciadores

Los espacios en blanco antes o después de un valor de datos no tienen efecto y pueden quitarse. Entonces, se puede comenzar a escribir en cualquier lugar de la línea.

Por ejemplo: 2 = ALLOY
Y 2 = ALLOY
produciran el mismo resultado

## Datos conteniendo caracteres especiales

Cuando un valor de datos contiene un valor especial como una coma, un signo de dólar, o un espacio en blanco al principio, entonces el valor debe encerrarseentre comillas sencillas. Por ejemplo.

LOCATION= 'ORANGE, NEW JERSEY'
la coma insertada es ignorada y convierte en parte del valor de los datos.

Ejemplo. Traspaso de valores.

```
MODIFY FILE SAMPLE
MARCH PRT AREA AMT
DATA
PRT= ALY, PNAME= ALLOYS
AREA= TRUCKS, 1000 ,. 50, $
AREA= CARS, 500 ,. 42, $
AREA = BUSES, 1100,. 51,$
END
```

Nótese que los valores de datos para PROD-TYPE y PROD-NAME fueron proporcionados en la primera transacción y ya que no fueron proporcionados otros valores están todavía activos para la segunda y tercera transacción.

Un valor computacional puede colocarse en cero un dato alfanumérico en un espacio en bvlanco usando una coma extra como separadora. Por ejemplo,

```
1=  STL,, CARS,,, $
```

Los campos 2, 4, y 5 se ponen en cero o en blanco tanto como sea necesario.

## Subcomando FREEFORM

Los valores de datos que sean entrados en el formato libre son separados uno de otro por medio de comas. La suposición de omisión es que cada coma representa un brinco al siguiente campo en el orden en que son descritas en el disconario MASTER. Entonces un registro de transacciôm compuesto de los siguientes datos representarían 7 campos de datos:

```
A, B, C,  ,  , V, W, $

1  2  3   4  5 6  7
```

Campos:

Nôtese que después de ¯C' es tercero, cuarto y quinto campos que no tienen valor de datos se cuentan debido a las comas.

Este registro de transacción puede presentarse asi

```
A, B, C, 6 = V, W, $
```

El quiebre debido a la ausencia de los campos 4 y 5 está cubierta al nombrar explicitamente 6= ( o el nombre de campo= )

Cuando el orden narural de los campos no es una manera conveniente de escribir los valores debido a los quiebres o porque una secuencia diferente es más sencilla,;,e,3,2,1, en lugar de 1,2,3, entonces el subcomando FREEFORM es usado.

El subcomando FREEFORM espedifica el orden en el cual los campos son presentados en la transacción. Las comas son interpretadas como el principio de un campo nuevo, ninguno es más grande que el último, pero como el siguiente en la lista FREEFORM.

La sintaxis del comando es:

FREEFORM [ ON ddname] campo, campo, campo, campo.

Los argumentos del subcomando es una lista de nombres de campo en el orden en que esos campos serán presentados.

Ejemplo: Datos de formato libre.

```
MODILFY FILE CARS
* FOUR FIELDS PROVIDED
* THREE TO LOCATE RECORD
* ONE TO BE UPDATED
FREEFORM CAR COUNTRY MODEL MPG
MATCH CAR COUNTRY MODEL
ON MATCH UPDATE MPG
ON NOMATCH REJECT
DATA
FIAT, ITALY, 128 2 DOOR, 31, $
MASERATI, ITALY, BORA 2 DOOR, 29, $
DATSUN, JAPAN, 260Z, 33, $
END
```

Los valores de los datos pueden estar ausente de una transacción en el cual se denota un quiebre por un campo siendo identificado por su nombre.i,e,

```
FIAT, ITALY, 128 2 DOOR, 31, $
MODEL = 128 4 DOOR AUTO, 24, $
Traspaso de CARY COUNTRY
```

Las transacciones deben terminar con un caracter de '$' si hay un error detectado en la transacción, entonces el final de la transacción , es el primer finalizador encontrado.

Cuando el subcomando FREEFORM se usa plara anular el orden natural de la ocurrencia del campo, se debe observar una regla muy importante.

> Solo los campos mencionados en la lista FREEFORM serán proporcionados como entradas (input). Si un registro de transacción contiene una coma extra , o un nombre de campo que no está en la lista, la la transacción será rechazada.

## Subcomandos PROMPT

### Indicaciones para valores de transacciones

Cuando se usa el subcomando PROMPT el FOCUS le pedirá al operador que escriba los valore de los datos en respuesta a los mensajes indicados . El operador usa el método de delimitación de la coma de formato libre para describir los datos, pero como sólo un valor de campo es pédido a la vez, el operador necesita escribir un valor de campo y después oprimir el regreso. Si varios valores son escritos, entonces deben separarse por medio de comas. FOCUS se brincará hacia adelante y resumirá las indicaciones para los valores aún no indicados.

Ejemplo: Entrada de datos PROMPT

```
MODIFY FILE CARS
PROMPT COUNTRY CAR MODEL RCOST
MATCH COUNTRY CAR MODEL
ON MATCH UPDATE RCOST
ON NOMATCH REJECT
DATA

DATA FOR TRANS  1
COUNTRY     = ENGLAND
CAR         = AUSTIN
MODEL       = MARIANA 2 DOOR GT
RETAIL COST = 2549
DATA FOR TRANS  2
COUNTRY     = ENGLAND, AUSTIN, MARIANA 4 DOOR GT
RETAIL COST = 2499
DATA FOR TRANS 3
COUNTRY     = END
```

Cuando se usa el PROMPT es necesario nombrar todos los campos de interés en los subcomandos ya que son los únicos indicados.

Sin embargo, si no se imprime ningún subcomando MATCH, entonces la suposición es la de igualarlos todos.

El subcomando PROMPT* significa indicar todos los campos en el archivo. Ellos serán pedidos en el orden en que son descritos en el diccionario MASTER.

Para finalizar la sesión PROMPT se escribe la palabra END en respuesta a cualquier indicación. Si la palabra END se va a introducir como dato, se debe encerrar entre comillas, e,g, 'END' . La palabra RUN es sinónima a END de la misma manera que MODIFY es a TABLE.

Corrigiendo una respuesta anterior:

Hasta que la última indicación para la transacción haya sido satisfecha, el operador tiene la oportunidad de cambiar los valores proporcionados para cualquier campo. Esto se logra contestyando a la sindicación actual, después escribiendo una coma seguida por el nombre de campo= valor nuevo, en donde el nombre del campo debe ser corregido y el valor nuevo es reemplazado por el campo.

Ejemplo: Corrigiendo una respuesta indicada.

```
COUNTRY      = italy       correction
CAR          = fiat
MODEL        = bora 2 door, maserati
RETAIL COST  = 31,000
DATA FOR ...
```

Escribiendo hacia adelante:

Después de haber indicado varias transacciones, el operador puede estar finalizando con la secuencia de las indicaciones y proporcionar varios valores en un mismo tiempo, y de esta manera desviar el mensaje indicando esto se logra finalizando la indicación actual con una coma, y proporcionándole valores por lo que podrían ser las indicaciones siguientes, también entre comillas.

Ejemplo: Escribiendo después de las indicaciones.

```
MODIFY FILE CARS
PROMPT COUNTRY CAR MODEL MPG
MATCH COUNTRY CAR MODEL
ON MATCH UPDATE MPG
ON NOMATCH REJECT
COMPUTE CHANGEDATE
DATA
```

```
                DATA FOR TRANS  1
                COUNTRY       = italy
                CART          = fiat
                MODEL         = 128 2 door
                MPG           = 23
                DATA FOR TRANS 2

Note typing ahead COUNTRY     = japan, datsun, B210 4 door, 29
                DATA FOR TRANS 3
                .
                .
                .
```

Indicando para grupos de campos:
---

Normalmente todos los campos nombrados en el subcomando PROMPT serán pedidos para cada transacción. Sin embargo, uno o más grupos de campos pueden ser repetidos un número de veces arreglado. Este núemero puede ser más grande porque el joperador puede `salirse' de un grupo escribiendo el signo de exclamación (!). Indicar entonces continúa en el siguiente grupo más alto, o si no hay otro grupo, desde el principiode la lista de campos.

La sintaxis para especificar un grupo es el de encerrar la lista de campos en el subcomando PROMPT entre paréntesis con el número de replicación al frente del paréntesis , e,g.

Ejemplo. Indicando con grupo de campos.

```
MODIFY FILE CARS
PROMPT COUNTRY CAR 5 ( MODEL SEATS RCOST MPG)
MATCH COUNTRY CAR
ON MATCH REJECT
MATCH MODEL
ON NOMATCH INCLUDE
ON MATCH UPDATE SEATS RCOST MPG
DATA
```

```
DATA FOR TRASNS  1
COUNTRY       = italy
CAR           = fiat
MODEL         = 128 2 DOOR
SEAT          = 4
RETAIL COST   = 2741
MPG           = 22
DATA FOR TRANS  2
MODEL         = 128 4 DOOR
SEATS         = 4
RETAIL COST   = 3049
MPG           = 21
```

Ejemplo: Indicando con grupos de campos (cont' d )

```
DATA FOR TRANS  3
MODEL         = !      NOTE ~ break out' of group
COUNTRY       = japan
CAR           = datsun
MODEL         = B210  2 door
SEATS         = 4
RETAIL COST   = 2899
MPG           = 27
DATA FOR TRANS 4
MODEL         = B210 4 door, 5, 3184, 25  Note typing anhead
DATA FOR TRANS 5
.
.
```

## Grupos de campo agrupados:

Puede haber varios campos de grupos repetitivos, y estos pueden estar ya sea en paralelo o en grupos.

Por ejemplo el patrón :

```
PROMPT campo n (campo) m (campo)
```

es un grupo paralelo de dos grupos, y

```
PROMPT campo n ( campo m (campo ) )
```

es un set agrupado de dos grupos.

El caracter de quiebre (!) puede ocasionar que la secuencia se mueva de un grupo al siguiente, de izquierda a derecha en patrones paralelos y de adentro hacia afuera en patrones agrupados.

Respuesta de datos faltantes:

Si no hay ningún valor de datos para un campo indicado la respuesta correcta es un punbto `.' , y después presionar el `return'. Presionando el `return' solamente, provocará el eco del modo FOCUS . i,e, PROMPT. El punto marca el campo como `no presente' y si está involucrado en un update ( una actualización para ese campo sin embargo si el segmento con un campo faltante es incluido en el archivo, entonces se usa un valor de omisión de cero para números y espacios para alfanuméricos.

Ejemplo: Indicando con valores faltantes.

```
MODIFY FILE CARS
PROMPT COUNTRY CAR MODEL BODY RCOST DCOST SEATS
MATCH COUNTRY CAT MODEL BODY
ON MATCH UPDATE RCOST DCOST SEATS
ON NOMATCH REJECT
DATA

DATA FOR TRANSACTION 1
COUNTRY      = JAPAN, TOYOTA, 1000 2 DOOR COROLLA
BODY         = SEDAN
RCOST        = 4850
DCOST        = .
STATE        = .
DATA FOR TRANSACTION 2
.
.
```

Repetición der la última petición:

Si la respuesta a un valor de campo indicado es el de revisar el valor dado en el registro previo después una marca," , ( doble sencilla se escribe.

Ejemplo : Reuso del valor indicado anterior

```
DATA FOR TRANSACTION 1
COUNTRY       = W GERMANY
CAR           = BMW, C400 AUTO
RCOST         = 7800
DATA FOR TRANSACTION 2
COUNTRY       = "
CAR           = "
MODEL         = C200 2 DOOR
RCOST         = 7600
```

Texto de indicación proporcionado por el usuario:

El texto de indicación por omisión es el nombre del campo de datos, cualquier texto puede ser proporcionado, agregándolo al subcomando PROMPT en el siguiente formato.

PROMPT nombre de campo. texto. nombre de campo. texto.

Nótese el uso de un punto después del nombre de campo para separarlo del texto, y el punto de cierre para finalizar el texto. El FOCUS no proporcionará un signo de igual (=) , asi es que si se quiere uno, tiene que ser parte del texto proporcioinado.

*Ejemplo:*

```
PROMPT COUNTRY, ENTER NAME OF COUNTRY=
PROMPT CAR. NAME OF CAR=,. RCOST. RETAIL COST=.
```

This appears as:

```
ENTER NAME OF COUNTRY=
NAME OF CAR=
RETAIL COST=
```

Cancelando respuestas anteriores:

Si la respuesta a cualquier indicación es el caracter determinación, `$' , entonces la transacción actual que está siendo ensamblada es cancelada. El siguiente mensaje es desplegado:

(FOC309) TRANSACTION IN COMPLETE:

Esto le permite al operador cancelar las respuestas en curso y comenzar de nuevo. El número de transacción es incrementado y la indicación comienza desde el principio de la lista de campos.

## Calculando valores nuevos:

Se pueden computar nuevos valores de datos de los datos de valores en la transacción , se pueden insertar constantes. Estos pueden definir valor faltantes en la transacción o en los valoresde cambios sometidos. Los cálculos, por ejemplo, ueden ser usados para escalar un número multiplicándolo por una constante o proporcionar una fecha constante a todas las transacciones sometidas en esa fecha.

Los cálculos son colocados bajo el subcomando COMPUTE. La parte izquierda del signo de igualdad es el nombre del campo siendo computado. Es el mismo nombre que el de un campo de datos el cual está em ;a transacción en donde se lleva a cabo el lugar del valor de la transacción. Si el mismo que el de un campo de datos en el archivo, pero en la transacción, toma su valor calculado como si estuviera en la transacción.

La expresión usa la sintaxis descrita en la sección 1 del manual del usuario bajo cálculos ejecutados cada expresión tiene que terminar con un punto y coma i, e,

```
COMPUTE
nombre de campo= expresión ;
nombre de campo= expresión ;
```

Ejemplo: Computando valores de transacción nuevos.

```
        MODIFY FILE CARS

subcomand     COMPUTE
definition    RCOST= IF COUNTRY EQ 'ITALY' THEN RCOST/600
temporary           ELSE RCOST;
definition    BODYCODE/I 1 = ;
definition    BODYTYPE= DECODE BODYCODE (1 SEDAN
                        2 CONVERTIBLE    3 ROADSTER    4 WAGON
                        5 COUPE ELSE OTHER
definition    DATE= 670421;
layout of     FIXFORM COUNTRY/10 CAR/10 MODEL/20
transaction   FIXFORM BODYCODE/1 RCOST/6 DCOST/6
              MATCH COUNTRY CAR MODEL
              ON NOMATCH REJECT
              MATCH BODYTYPE
              ON NOMATCH REJECT
              ON MATCH UPDATE RCOST DCOST
              DATA ON JERRY
              END
```

Las funciones del usuario y todas las características del comado FOCUS DEFINE están disponibles para las expresiones COMPUTE.

El subcomando COMPUTE usa valores del registro de transacciones solamente. Las expresiones son computadas antes de cualquiera igualación o de cualquier otro proceso de la transacción si el cálculo requiere de valores al archivo de datos, entonces las opciones ON MATCH COMPUTE o el ON NOMATCH COMPUTE son usadas. Esto permite que puedan estar en una expresión tanto transacciones mezcladas como valores de archivos de datos.

## Cálculo y validez después de MATCH O NEXT

Las expresiones que siguen las acciones:

```
        MATCH
        NOMATCH       COMPUTE
ON      NEXT          VALIDATE
        NONEXT
```

Pueden referirse tanto a las transacciones de valores de campo como a valores de campo de datos. Para distinguirlos uno de otro, del prefijo`D.' es colocado enfrente del campo de archivo de datos. Por ejemplo D.RCOST se refiere al valor del camo RCOST como aobtenido del archivo de datos.

Todas las operaciones descritas en la sección 1 se permiten: todos los cálculos. Esto es, operaciones aritméticas, operaciones lógicas, la lógica IF-THEN-ELSE, funciones especiales y funciones escritas del usuario.

Ejemplo: Suma acumulativa de valores nuevos a valores existentes

```
RCOST= RCOST + D. RCOST;
```

Ejemplo: Cambia el valor de la base de datos si el valor es 0

```
RCOST= IF   D. RCOST EQ O THEN RCOST ELSE D. RCOST;
```

Ejemplo: Cambios de valores de base de datos sólo si ambos valores de transacción es no- espacio y el valor de la base de datos es un espacio.

```
ADDRESS= IF ADDRESS NE  ` ` AND D. ADDRESS EQ `
      THEN ADDRESS ELSE D. ADDRESS;
```

Ejemplo: Un valor en el registro es usado para computar valore nuevos

```
RCOST= IF D. FACTOR GT O THEN D. FACTOR * RCOST ELSE
         RCOST;
```

Ejemplo: Cómputo después de MATCH.

```
                          MODIFY FILE CARS
                          COMPUTE
Temporary field           MAR KUP/16
not in data file          FIXFORM COUNTRY/10 CAR/10 MODEL/20 BODY/6
                          MARKUP/6
                          MATCH COUNTRY CAR MODEL BODY
                          ON NOMATCH REJECT
                          ON MATCH COMPUTE
mixed calculation         RCOST= (1+ MARKUP/100) * D. DCOST;
use of new value          ON MATCH UP DATE RCOST
                          DATA ON SPEEDS
                          END
```

Nótese que un campo en la transacción que no existe en el archivo de datos le ha sido dada una definición temporal, entonces es usada para producir un campo de archivo de datos.

## Concordando transacciones a registros de base de datos

La operación básica durante el kproceso de transacción es el de concordar algunos valores de una transacción a valores correspondiente en la base de datos y tomar acciones dependiendo si hay una concordancia completa, parcial, o no haya.

La designación de los campos a concordar son proporcionados bajoi del subcomando MATCH. Los nombres de los campos están en una lista uno enseguida de otro separados por espacios en blanco.
(Si un nombre contiene un espacio en blanco insertado, se debe de encerrar entre comillas sencillas).

Se pueden suministrat tantas líneas como sean necesarias en el subcomando MATCH e,g,

```
MATCH DAIS MODELO CARRO
MATCH ASIENTOS TIPO
```

Nótese que los nombres de campo pueden siempre ser remitidos por su nombre de campo completo, un nombre alias alterados o por la truncación más pequeña única.

Entonces, MATCH CON MO CAR SE TY
puede ser equivalente al anterior.

Después del subcomando MATCH las acciones a tomar son específicas mediante los subcomandos ON MATCH Y ON NOMATCH

### Concordando los campos clave.

Los campos claves son usados ara identificar segmentos. Un segmento con el tipo SEGTYPE= SI tiene su clave en el primer campo del segento; SEGTYPE=S2 tiene la clave en los primeros dos campo. Al concordarlos, todos las claves de los campos tienen que ser especificados, y si no FOCUS se agregan los campos faltantes a una lista de aquéllos incluidos e impide un mensaje de precaución. Cuando ese campo no tiene valor de transacción, el FOCUS asume el valor de omisiónde cero para los campos numéricos o de espacios en blanco para los camos alfanuméricos.

Incluyendo los registros en niveles de detalles de un archivo.

MATCH*

Los niveles superiores de un archivo con frecuencia contienen claves usadas ara localizar registros mientras que los niveles inferiores contienen el volúmen de los campos de datos. Una manera sencilla para referirse a todos los campos en estos niveles es el de colocar un subcomando MATCH* después del control de información principal . ( el * es un símbolo de taquigrafía usada para `all'). La colocación de este subcomando en el flujo del control MODIFY es por lo general al final porque es efectivo sólo para segmentos que no han sido mencionados. Esto es, aquellos para los cuales no se ha proporcionado más información.

Ejemplo: Uso de MATCH*

```
MODIFY FILE CARS
FORM COUNTRY/10 CAR/10 MODEL/20 BODY/5 RCOST/6
FORM DCOST/6 LONG/5 WEIGH/5 WBASE/5  MPG/4
FORM WARRANTY/40
MATCH COUNTRY
ON MATCH REJECT
MATCH*
ON NOMATCH INCLUDE
DATA ON NEWCARS
END
```

Cortando campos sin clave.

No todos los segmentos tienen campos clave e.g., SEGTYPE=SO o espacio. Para seleccionar un segmento en particular, se puede usar cualquier registrode campos. Se pueden proporcionar campos clave de manera individual en un enunciado MATCH para los segmentos con clave.

Precaución: Se debe de tener precaución al seleccionar registreos de esta manera.

Nótese que el subcomando ON NOMATCH INCLUDE puede permitir la inserción no intencional de registros con claves duplicadas. Esta situación necesita expander, con frecuencia el número de claves en una descripción de archivo.

El subcomando NEXT

Una manera alternada de localizar una nueva posición es el de usar el subcomando `NEXT' . Este tiene el efecto de moverse de la posición actual en un archivo hacia el siguiente registro lógico. No se requiere de indices de datos iniciales.

La sintaxis es:

NEXT nombre de campo

Cuando el campo nombrado es el primero en el segmento. Los subcomandos asociados ON NEXT y ON NONEXT controlan lo que ocurrirá cuando existe una petición, o cuando no existe. Generalmente, los contenidos de algunos archivos deben de desplegarse como se usa NEXT, para `curiosear' dentro de un archivo. e,g.

Ejemplo: Curioseando entre un archivo.

```
MODILFY FILE CARS
PROMPT COUNTRY CAR MODEL
MATCH COUNTRY CAR MODEL
ON MATCH GOTO SHOW
ON NOMATCH GOTO TOP
CASE SHOW
     NEXT BODY
        "BODYTYPE <D. BODY MILAGE< D.MPG"
        "RETAIL COST < D. RCOST"
        ON NEXT GOTO SHOW
        ON NONEXT GOTO TOP

ENDCASE
DATA
```

Acciones que siguen a las pruebas de concordación.
acciones de concordancia.

ON MATCH. Si una transacción hace concordat todos los campos nombrando en el subcomando MATCH anterior, entonces pueden ocurrir las siguientes posibles acciones:

1. ON MATCH REJECT.

La transacción se considera como una duplicación (al menos hasta el punto de la concordancia), y por lo tanto tiene que ser rechazado.

2. ON MATCH UPDATE nombre de campo nombre de campo...etc.

Un registro de base de datos que concuerda con las claves nombradas en el subcomando anterioir, si se encuentra, tiene que tener la siguiete lista de campos puestos a la fecha (varias líneas, cada una comienzan con la palabra UPDATE puede ser usada si más de una línea es necesitada).

Ejemplo: Poniendo al corriente los registros existentes ON MATCH UPDATE.

```
MODIFY FILE CARS
MATCH COUNTRY CAR MODEL BODY
ON MATCH UPDATE RCOST DCOST
ON NOMKATCH REJECT
DATA ON UPDATE
END
```

3. ON MATCH DELETE

Desde el punto de concordancia en donde un segmento de registro omitido, y todos los registros dependientes y todas las refefencias de los campos suprimidos en todos los índices, son removidos.

Ejemplo: Supresión de registros existentes ON MATCH DELETE

```
MODIFY FILE CARS
MATCH COUNTRY CAR
ON MATCH DELETE CAR
ON NOMATCH REJECT
DATA
COUNTRY = ITALY, CAR = FIAT, $
COUNTRY = JAPAN, CAR = DATSUN,$
                 CAR = TOYOTA, $
```

4. ON MATCH COMPUTE
   campo= expresión;
   campo= expresión;

   Los campos mencionados son computados cuando las claves en la transacción concuerda con aquellos en el segmento de base de datos La expresión puede referirse a los valores de transacciones y a los valores de base de datos que están presentes en el momento de concuerdan o en un segmento pariente. Los cambios a los valores de la transacción actual son computados de acuerdo a la lógica en la expresión. Estos nuevos valores pueden poner al corriente la base de datos cuando está presente ON MATCH UPDATE.

Ejemplo: Computando un valor de transacción nuevo usando valores de base de datos ON MATCH COMPUTE.

```
MODIFY FILE CARS
COMPUTE
MILES/ 16
FREEFORM COUNTRY CAR MODEL BODY MILES
MATCH COUNTRY CAR MODEL BODY
ON MATCH COMPUTE
MPG= MILES/D.GALLON;
ON MATCH UPDATE MPG
ON NOMATCH REJECT
DATA ON AREA
END
```

5. ON MATCH INCLUDE

   Aún y cuando una transacción concuerda, un registro de base de datos tiene que estar incluído. No se pueden identificar claves únicas, o el orden de entradas que identifican los registros.

6. ON MATCH TYPE

   El texto del mensaje a ser escrito en la terminal y lo en el archivo puede ser proporcionado sobre una línea, o sobre líneas múltiples.

```
ON MATCH TYPE (ON ddnombre) "texto"
ON MATCH TYPE [ON DDnombre]
"texto"
"texto"
```

El texto del mensaje puede usar todas las opciones marcadores de lugar para poder poner en posición el texto. (ver sección 1 (apoyos para poder en posición el texto') Los valores de los datyos de la transacción pueden ser insertados en el texto encerrándolos entre comillas e.g. "nombre de campo" . Los valores de datos de la base de datos pueden ser desplegados poniéndose elk prefijo ~D! por ejemplo:

```
"COSTO ACTUAL ES < D, RCOST>'
```

ver el subcoimando TYPE para información adicional.

7. ON MATCH FREEFORM [ON ddnombre] nombre de campo nombre de campo...

   Se pueden leer datos adicionales del archivo de transacción para procesar datos adicionales que no fueron referidos en un comando anterior FREEFORM.

   La opción ~ ON ddnombre' le permite a los datos de forma libre ser leídos de dos o más archivos separados en el mismo comando MODIFY.

8. ON MATCH FIXFORM [ON ddnombre] campo/ longitud.,,

   Datos adicionales pueden ser leídos de un archivo de transacción ( o de otra archivo de transacción si la opción ~ON' es usada). Este subcomando continuará procesando el balance de los datos en un FIXFORM anterior si quedan datos, uno de los usos de ésto es para tener más eficiencia cuando se rechazan muchas transacciones. Sólo los cuantos campos necesarios para concordar (las claves) y aquellos para validar son procesados primero, después si la transacción es necesitada, se procesa el balance de los campos de transacción

Ejemplo: Leyendo una transacción en secciones.

```
MODIFY FILE CAR
FIXFORM COUNTRY/10 CAR/18 MODEL/25 BODY/18
MATCH COUNTRY CAR MODEL BODY
   ON NOMATCH REJECT
   ON MATCH FIXFORM MPG/4 FUEL/8 WEIGHT/5 HEWIGHT/5
   ON MATCH FIXFORM BHP/4 MAXSPEED/3
   ON MATCH UPDATE MPG FUEL WEIGHT HEIGHT BHP MAXSPEED
DATA ON MYDISK
END
```

Nótese que la opción de formato x- n, i.e, x-79 puede ser usado para retroceder una transacción para re- procesarlo.

9. ON MATCH CONTINUE TO nombre de campo.

   Esta opción es usada para el mantenimiento de segmentos únicos que no se encuentran directamente accesibles con un subcomando MATCH.

10. ON MATCH CONTINUE

    El procedilmiento continúa hacia el siguiente subcomando MATCH en el mismo caso. Este es normalmente la opción de omisión cuando un subcomando MATCH posterior está presente pero es más frecuentemente claro definir la acción explícitamente.

11. ON MATCH GO TO nombre de caso.

    El procedimiento va al caso el cual es nombrado y comienza a ejecutar los comandos en este caso.

12. ON MATCH IF expresión GOTO nombre de caso [ ELSE GOTO nombre de caso];

    El procedimiento se va al CASE nombrado y comienza a ejecutar el subcomando en ese caso.

Acciones cuando no concuerdan
ON NO MATCH

Si una transacción no concuerda con todos los campos nombrados en el subcomando MATCH anterior, entonces las acciones que puedieran ocurrir son:

1. ON NOMATCH REJECT

Una transacción que falla al encontrar un registro de base de datos correspondiente, es rechazado. Esto es acoplado con frecuencia con un UPDATE ya que los registros fueron cambiados y fueron rechazados los que no concordaron.

2. ON NOMATCH INCLUDE

Una transacción que falla para encontrar un registro de base de datos correspondiente, será incluído en la base de datos en sí. Este es el proceso básico para la entrada de un registro.

Ejemplo: Entrada de registros nuevos.

```
MODIFY FILE CARS
FORM COUNTRY/ 10 CAR/18  MODEL/6  SEATS/2 TYPE/19 RCOST/6
MATCH COUNTRY CAR MODEL TYPE
ON NOMATCH INCLUDE
DATA ON NEWSTUFF
END
```

La secuencia de los subcomandos MATCH y ON es significativa cuando se ejecutan procesos compuestos. Por ejemplo, un rdegistro puede ser rechazado si no concuerda hasta un cierto punto, pero desde ese punto, puede ser aceptable incluírlo en la base de datos el siguiente ejemplo ilustra esto:

Ejemplo: Concordancia parcial y poniendo al corriente o incluyendo.

```
MODIFY FILE CARS
MATCH COUNTRY
ON NOMATCH REJECT
MATCH MODEL TYPE
ON NOMATCH INCLUDE
ON MATCH UPDATE RCOST
DATA
```

3. ON MATCH COMUTE
   campo = expresión ;
   campo = expresión

Los campos nombrados son computados cuando las claves en las transacciones no concuerdan coin aquellos en el segmento de base de datos. La expresión se refiere a los valores de transacción y a los valores de base de datos que están solamente en segmentos parientes. Los cambios a los valores de transacción actuales son computados de acuerdo a la lógica en la expresión. Estos valores nuevos pueden entonces ser incluídos en la base de datos cuando un ON NOMATCH INCLUDE está presente.

Ejemplo: Usando valores de base de datos para computar valores nuevos hay un NOMATCH.

```
MODIFY FILE PROD
FIXFORM SSN/9 DATE/6 HOURS/3 TYPE/2
MATCH SSN
ON NOMATCH REJECT
MATCH DATE
ON NOMATCH COMPUTE
COST= IF TYPE EQ 'R' THEN HOURS* D. RATET
   ELSE HOURS* D. RATE2
ON NOMATCH INCLUDE
ON MATCH COMPUTE
COST= IF TYPE EQ 'R' THEN D. COST + HOURS* D. RATET
   ELSE D. COST + HOURS* D. RATE2 ;
ON MATCH UPDATE COST
DATA ON NEWIN
END
```

4. ON NOMATCH GOTO nombre de caso.

   El procedimiento va al CASE que es nombrado y comienza a ejecutar los subcomandos en ese caso.

5. ON NOMATCH TYPE

   Se pueden escribir ya se aun mensaje de una línea o de varias líneas en la terminal, en un archivo o en ambos. El texto del mensaje puede contener marcadorers de lugar para ayudara ponerlo en posición y puede contener valores de transacción de campo encerrados entre comillas.

```
ON NOMATCH TYPE [ ON ddnombre] "texto"
ON NOMATCH TYPE [ ON ddnombre]
" texto "
" texto "
```

6. ON NOMATCH CRTFORM [ LINE n ]

   Ver la sección sobre FIDEL para mayores detalles sobre procesamiento de CRTFORM.

7. ON NOMATCH FIXFORM [ON ddnombre]

   Datos adicionales pueden ser leidos de la transacción de archivos primarios o de algún otro archivo. Si está en otro archivo, entonces el primer archivo de transacción que alcance el final termina el procedimiento. El uso principal de esta opción es para eficiencia, cuando varios registros están rechazados.

Ejemplo: FIXFORM en secciones.

```
MODIFY FILE CARS
FIXFORM X79 CODE/1
MATCH CODE
   ON MATCH REJECT
   ON NOMATCH FIXFORM X-80 MONTH/4 PRODUCT/6 QTY/7
   ON NOMATCH INCLUDE
   DATA ON MYDATA
   END
```

8. ON NOMATCH IF expresión GOTO nombre de caso ELSE GOTO nombre de caso;

   El procedimiento va a un CASE nombrado, dependiendo del valor computado de una expresión.

## Acciones después de NEXT y NONEXT.

Si un comando NEXT localiza un caso `next' las acciones que pueden ocurrir son:

1. ON NEXT COMPUTE
2. ON NEXT CONTINUE ( to nombre de campo)
3. ON NEXT CRTFORM
4. ON NEXT FIXFORM ...
5. ON NEXT FREEFORM ...
6. ON NEXT GOTO nombre de caso.
7. ON NEXT INCLUDE
8. ON NEXT IF...
9. ON NEXT PROMPT
10. ON NEXT REJECT
11. ON NEXT TYPE
12. ON NEXT UPDATE
13. ON NEXT VALIDATE
14. ON NEXT DELETE

Efecto de cada uno es directamente equivalente al efecto de la acción ON MATCH equivalente.

Nótese que ON NEXT INCLUDE se permite sólo cuando el segmento tiene escrito SO o un "espacio".

Si el NEXT COMMAND no localiza otros casos `next', las siguientes acciones pueden ocurrir;

1. ON NONEXT COMPUTE
2. ON NONEXTY CRTFORM
3. ON NONEXT FIXFORM
4. ON NONEXT FREEFORM
5. ON NONEXT GOTO
6. ON NONEXT IF
7. ON NONEXT PROMPT
8. ON NONEXT REJECT
9. ON NONEXT TYPE
10. ON NONEXT VALIDATE
11. ON NONEXT INCLUDE

La acción tomada es exactamente equivalente a la acción tomada después de ON NOMATCH.

## Manteniendo segmentos únicos

Algunas consideraciones especiales son necesarias para mantener un segmento en un archivo FOCUS el cual tiene el atributo de SEGTYPE= U,(en donde `U' significa `único'), La clave del campo para identificar el segmento reside en su pariente. Entonces para localizar el segmento la concordancia tiene que estar en el nivel pariente del archivo. Las acciones tomadas en el `U' descendiente tienen que estar diferenciados de las acciones tomadas en el pariente en si. Para hacer ésto, el nombre de cualquier campo de datos en el segmento `U' se proporciona en la siguiente frase ON MATCH CONTINUE TO nombre de campo. i, e.

```
MATCH claves de campos
ON MATCH CONTINUE TO nombre de campo.
```

El ejemplo ilustra la mecánica básica. Se asumirá que el archivo tiene dos segmentos.

```
Archivo PERSL      UN SEGTYPE= SI

     ID
     NOMBRE COMP
     DEPT

                   DOS SEGTYPE= S2
     DIRECCION
     COD. OSTAL
     DOB
     SALARIO
```

Ejemplo: Incluyendo un segmento nuevo único.

```
MODIFY FILE PERSL
MATCH ID
ON NOMATCH REJECT
ON MATCH CONTINUE TO ADDRESS
ON NOMATCH INCLUDE
ON MATCH REJECT
```

Ejemplo: Suprimiendo un segmento único.

```
MODIFY FILE PERSL
MATCH ID
ON NOMATCH REJECT
ON MATCH CONTINUE TO ADDRESS
ON MATCH DELETE
```

Ejemplo: Actualizando un segmento único.

```
MODIFY FILE PERSL
MATCH ID
ON NOMATCH REJECT
ON MATCH CONTINUE TO ADDRESS
ON MATCH UPDATE ADDRESS ZIP DOB SALARY
ON NOMATCH REJECT
```

Ejemplo: Incluyendo otros valores actualizados si, no existen otros.

```
MODIFY FILE PERSL
MATCH ID
ON NOMATCH REJECT
ON MATCH CONTINUE TO ADDRESS
ON MATCH UPDATE ADDRESS ZIP DOB
ON NOMATCH INCLUDE
```

Nótese que en el último ejemplo es necesario especificar tanto la lista actualizada y el subcomando INCLUDE para ocasionar que uno o el otro le sucedan a un segmento en particular, dependiendo en que si existe un caso y que pueda ser actualizado o si no existe y tenga que incluirse.

Cuando existen varios segmentos únicos entonces se pueden usar frases CONTINUE separadas para dirigir el proceso de cada uno.

## Poniendo al corriente con transacciones de formato libre.

Cuando se proporcionan valores de datos en un formato libre (delimitados por una coma) cada transacción puede tener una lista de campos diferente a ser actualizado, si un campo que es mencionado en el subcomando UPDATE no le ha sido dado un valor en la transacción entonces no le ocurren cambios a los valores de base de datos correspondientes. Esta es la equivalencia de la opción presente condicionalmente en los formatos arreglados.

Ejemplo: Actualizando en formatos libres.

```
        MODIFY FILE CARS
        MATCH COUNTRY CAR MODEL
        ON MATCH UPDATE MPG WBASE WEIGHT
        ON NOMATCH REJECT
        DATA
        COUNTRY= JAPAN, CAR= DATSUN, MOD= B210, MPG= 33
(1)     WEIGHT=2050, $
(2)     MOD= 610, MPG= 27, WBASE=98.4,$
(3)     MOD= 710, WBASE=96.5, $
        END
```

(1) Transacción 1 Sólo MPG y WEIGHT han sido cambiados
(2) Transacción 2 Sólo MPG y WBASE han sido cambiados.
(3) Transacción 3 Sólo WBASE ha sido cambiada.

## Actualizando registros en índice.

Si un campo de datos ha sido puesto en índice,i.e, FIELDTYPE=I entonces el segmento en donde un campo de índice tiene un valor dado, puede ser localizado directamente aún y cuando no sea el primer(raíz) segmento de un archivo. El campo tiene que ser un campo clave en un segmento si. Una vez que el segmento haya sido localizado por medio de un archivo de índice, cualquier campo en el segmento puede actualizarse. En segmentos descendientes (hijos), las operaciones de incluir y omitir tanto como el de actualizart pueden ejecutarse.

El procedimiento para localizar un segmento basado en valores de índice, usa la característica o facilidad de visión de archivos' (ver visiones de archivo alternadas en la sección 3 ). La sintaxis para invocar una vista de archivos es el de añadir el nombre de campo de índice al nombre de campo e.g., MODIFY FILE, CAR. CAR. El ejemplo siguiente demuestra un uso típico de esta facilidad. La identidad de un carro en el archivo CARS está clara de su nombre y modelo. El paí de origen no se necesita, y puede no estar presente en algunos transacciones. Esquemáticamente el archivo CARS aparece como:

Ejemplo :

| | COUNTRY | CARS FILE |
|---|---|---|
| VIEWPOINT | CAR I | |
| WARRANTY | MODEL | |
| | BODY | |
| | SPECS | |

Ejemplo: Uso de visión de archivos para actualizar.

Note

```
MODIFY FILE CARS. CAR
FIXFORM CAR/18 MODEL/20 BODY/8 RCOST/6 DCOST/6
MATCH CAR
   ON NOMATCH REJECT
   ON MATCH CONTINUE
MATCH MODEL BODY
   ON NOMATCH REJECT
   ON MATCH UPDATE RCOST DCOST
DATA ON NEWSTUF
END
```

## Condiciones de omisión

### Match

Si no se proporciona ningún subcomando MATCH, entonces todos los campos que son procesados serán usados por omisión como condiciones de concordancia. Estos son útiles para la entrada de datos.

Ejemplo:

```
MODIFY FILE CARS
FORM COUNTRY/10 CAR/10 MODEL/20 SEATS/2 RCOST/5
FORM DCOST
DATA ON NEWSTUFF
END
```

Cuando un archivo tiene datos en él, entonces no se recomienda que la omisión sea usada. Es mejor seleccionar específicamente los campos a concordar.

ON NOMATCH INCLUDE.

Si una transacción no concuerda con el registro de base de datos y no se proporciona ninguna intrucción para el match o el no-match, entonces la omisión es:

```
ON NOMATCH INCLUDE
```

Tan pronto como cualquier intrucción haya sido proporcionada sobre lo que haya que hacer si hubiera una concordancia. i.e, ON MATCH UPDATE entonces la omisión cambia a ON NOMATCH REJKECT.

Entonces, un procedimiento que es UPDATE e INCLUDE, tiene que específicamente citar estas acciones. La razón usará estas omisiones que estas claramente se relacionan a tres procesos separados y simplifican la lógica de cada uno;

1. Mera inclusión de registros nuevos, no hay actualización; la omisión es ON NOIMATCH INCLUDE.
2. Mera actualización, son inclusión de registros nuevos; la omisión es ON NOMATCH REJECT.
3. Ambas actualización e inclusión, las acciones tienen que ser especificadas.

Control de registros duplicados ON MATCH INCLUDE.

La omisión del comando MODIFY es el de rechazar transacciones si existe un duplicado en la base de datos. Sin embargo ya que sólo los campos de datos mencionados en el subcomando MATCH son comparados con sus equivalencias en el archivo de datos es importante especificar claramente cuáles partes de las transacciones identifican un registro único, en lugar de ser rechazados basados en un subgrupo de campos muy pequeños. En algunas situaciones las transacciones pueden no tener valores únicos o por otras razones todas las transacciones deben ser incluídos en el archivo de datos. Estos se logra por medio del subcomando.

```
ON MATCH INCLUDE.
```

Si el MATCH rodea todos los campos de datos entonces el subcomando anterior debe imprimirse en el siguiente orden:

```
MATCH *
ON MATCH INCLUDE
```

El rechazo de los duplicados de algunos segmentos pero no de otros está controlado en la forma usual. Considérese el siguiente ejemplo.

EJEMPLO: Control de campos duplicados en algunos segmentos del archivo.

```
MODIFY FILE RIVDATA
MATCH RIVER
ON NOMATCH REJECT
MATCH DATE SHIP CARGO WEIGHT
ON MATCH REJECT
ON NOMATCH TEMPERATURE BACTCOUNT
ON NOMATCH INCLUDE
ON MATCH INCLUDE
DATA ON DAILYRPT
END
```

Errores de apunte y mensajes de error

La operación de omisión es el de desplegar las transacciones rechazadas en la terminal e imprimir la razón correspondiente del rechazo. Estas razones son:

1. NOMATCH La transacción no hace que concuerde el registro de base de datos aunque haya sido requerido por el procedimiento.
2. DUPL La transacción hace concordar un registro de base de datos aunque no haya sido requerido por el procedimiento y es considerado un duplicado.
3. INVALID Falla una prueba de criterio VALIDATE.
4. FORMAT Se detecta un error en el formato de los datos, i,e, caracteres no-numéricos en un campo numérico.

La omisión al repetir los rechazos en una terminal pueden ser cambiados, y las transacciones pueden ser escritas para apuntar archivos que hayan sido FILEDEF'S o ALLOCATED antes del inicio del procedimiento.

También, todas las transacciones aceptadas o rechazadas pueden ser apuntadas, o sólo todas las transacciones aceptadas que han modificado los archivos de datos, pueden ser apuntadas.

5. ACCEPT Una transacción aceptada.
6. TRANS Todos los registros presentados.

La sintáxis para especificar el apuntar, o sólo el control del mensaje de error es:

```
LOG  escribilr ON ddnombre MSG  on ( of off)
o solo si el apunte logging sea afectado
LOG escribir ON dd nombre.
o sólo si los mensajes sean afectados
LOG escribir MSG  ( on  off )
```

El registro de transacción completo es escrito a cualquier archivo log en el mismo formato en donde fue leído. Hasta seis archivos de apunte en un procedimiento.

```
LOG NOMATCH ON BADMAT
LOG FORMAT ON BADDAT
LOG DUPL ON GOT MSG OFF
LOG ACCEPT ON TRAIL
LOG INVALID MSG OFF
```

Los archivos de apunte tienen que dárseles un FILEDEF completo incluyendo longitud de registro.

El archivo LOG debe tener el mismo FILEDEF que el flujo de transacción original.

A los archivos de apunte se les debe dar una longitud igual a la longitud del registro lógico. Por ejemplo, si tres registros de transacción de 80 caracteres son usados para producir una transacción lógica, entonces el largo del archivo de apunte es de 240 caracteres.

EJEMPLO:Apuntando NOMATH'S durante el procedimiento MODIFY.

```
FILEDEF NOPE DISK NOPE VALUE ( LRECL 90 RECRM FB BLKSIZE 900)

FILEDEF NEWAL DISK NEWS DATA ( LRECL 90 LRECL FB BLKSIZE 900)
.
.
MODIFY FILE CARS
FORM COUNTRY/10 X10 CAR/15 MODEL/20 X5 MPG/5 X25
LOG NOMATCH ON NOPE MSG OFF
ON NOMATCH INCLUDE
DATA ON NEWAL
END
```

## El subcomando TYPE.

El subcomando TYPE le escribe un mensaje a la terminal y/o un archivo en ocasiones controladas durante el proceso de transacción.

Estas ocasiones son:

a). Al comienzo de un proceso

```
TYPE AT START [ ON    ddnombre]
"Texto"
"Texto"
```

b). Durante el proceso de cada transacción

```
TYPE [ ON ddnombre]
" texto"
" texto"
```

c). En posiciones seleccionadas MATCH O NOMATCH

```
ON MATCH TYPE [ ON ddnombre]
" texto"
ON NOMATCH TYPE [ ON ddnombre]
" texto"
```

d). Al finalizar un proceso

```
TYPE AT END [ ON ddnombre]
" texto"
```

e). En el momento en que se detecta una condición inválida,o una condición válida.

```
ON INVALID TYPE [ ON ddnombre]
" texto"
ON VALID TYPE [ON ddnombre]
" texto "
```

La opción [ ON ddnombre ] puede nombrar cualquiuer grupo de datos *extremos para escribir sobre ellos mientras que un FILEDEF (O ALLOC)propio* haya sido impreso para él. Si no es mencionada una opción `ON' el mensaje se escribe en la terminal.

El texto del mensaje puede tener cualquier número de líneas de longitud, cada línea puede comenzar y terminar con signos de comillas de dobles o sólo con comillas al principio de la primera línea o al final de la última línea. Este último caso, las líneas están en pares.

por ejemplo

```
"texto
 texto
 texto
 texto"
```

Las líneas 1 con la 2 y la 3 con la 4 producen un mensaje de dos líneas

Se pueden usar marcadores de lugar para colocar el texto.

Por ejemplo

```
TYPE
"<20 PAIS DER ORIGEN <PAIS> + <MODELO>"
```

Los nombres de campo pueden ser encerrados entre <> en el texto y el valor de transacción de estos campos, desplegado. Cuando la opción ON MATCH TYPE es usada entonces los valores de la base de datos también están disponibles. El prefijo `Di' antes del nombre de campo de la base de datos, desplegará su valor actual.

EJEMPLO: Desplegando antes y después de los valores actualizados.

```
MODIFY FILE CARS/
FIXFORM COUNTRY/10 CAR/10 MODEL/20 BODY RCOST/8
MATCH COUNTRY CAR MODEL BODY
ON NOMATCH REJECT
ON MATCH TYPE
" <COUNTRY <CAR<MODEL<BODY  CURRENT COST <D.RCOST"
" NEW COST <RCOST "
ON MATCH UPDATE RCOST
DATA ON INDATA
END
```

Escribiendo un mensaje al final del proceso permite un " total desmesurado" o que cualquier otro final del cálculo del proceso sea desplegado. El siguiente ejemplo: ilustra como un total puede desplegarse.

EJEMPLO: Computando un total de tanda.

```
MODIFY FILE CARS
FIXFORM COUNTRY/ 10 CAR/10  MODEL/15  BODY/6
FIXFORM RCOST/6 DCOST/6
COMPUTE
TOTAL R= TOTALR + RCOST;
MATCH COUNTRY CAR MODEL BODY
   ON NOMATCH REJECT
   ON MATCH UPDATE RCOST DCOST
TYPE AT END
"TOTAL ALUE OF ALL RCOST SUBMITTED <TOTALR>"
DATA ON NEWVAL
END
```

Proceso de transacción.

Subcomandos de control de transacción

Las transacciones pueden suministrarse en cuatro maneras.

1. Escribirlas directamente sobre la terminal ya sea en formato libre,o en un formato arreglado.

2. Almacenado en archivos de cinta o disco temporales. (de tarjetas, cinta de papel, cassetes,, o de programas),

3. Escritoas directamente en la terminal, en respuesta a preguntas de indicación para cada valor de datos o de modos de pantalla completa.

4. Proporcionada vía va programa especial.

El último subcomando que tiene que proporcionarse como es, denota el final de la lista de subcomandos, y el rincipio de la transferencia de datos es ya sea;

```
DATA
 O
DATA ON ddnombre
 o
DATA VIA  programa
```

Cuando el subcomando es sólo la palabra DATA significa que la transacción seguirá en la terminal lo de un procedimiento catalogado.

Cuando el subcomando es;

```
DATA ON ddnombre
```

En donde ddnombre es el nombre usado en el enunciado FILEDEF para ayudar a localizar transacciones almacenadas entonces las transacciones son leidas de este archivo.

EJEMPLO. Leyendo un archivo de transacción

```
FILEDEF MYDATA DISK C: TODAY. UP,
        ------

.
.
FOCUS
MODIFY FILE CARS
.
.
.
DATA ON MYDATA
        -------
```

EN MS-DOS si el archivo de transacción de es dado una impresión de archivo (filetype) de . DAT entonces no se requiere un FILEDEF y solo se necesita un nombre de archivo.

## Fuentes de transacciones mixtas

Dos de los subcomandos de formato de datos pueden también identificar la fuente de datos . Estos son:

```
FIXFORM [ON ddnombre]
FREEFORM[ON ddnombre]
```

El principal uso de esta característica es el de mezclar dos métodos de captura de datos. Estos pueden ser FIXFORM CON CRTFORM con FREEFORM con PROMPT. Considérese una situación en donde un grupo de registros de base de datos pueden ser almacenados en un archivo separado. Estos pueden ser seleccionados automáticamente, uno a uno y de cada uno, se pueden usar una secuencia PROMPT para obtener actualizaciones. (Las claves pueden ser colocadas en archivos separados vía procesos FOCUS TABLE y SAVES.

EJEMPLO. Mezclando FREEFORM con PROMPT.

```
FILEDEF OLDEQUIP DISK C: OLDVAL.DAT
MODIFY FILE EQUIP
FREEFORM ON OLDEQUIP
MATCH CODE
  ON NOMATCH REJECT
  ON MATCH PROMPT COST LOCATION
  ON MATCH UPDATE COST LOCATION
DATA
```

El subcomando END

El subcomando END designa el final del procedimiento MODIFY.
Si se esta proporcionando datos por medio de un `DATA ON ddnombre `o' del subcomando `DATA via programas, entonces el subcomando END debe seguir inmediatamente, e,g.,

```
DATA ON MY SOURCE
END
```

Sin embargo, los datos son proporcionados en la terminal, sería parte del MODIFY, y después de la última línea de datos, la palabra END finaliza el procedimiento, e.g.,

```
DATA
PAIS= JAPON, CARRO= TOYOTA, RCOST= 10 000, $
END
```

STARD

La primera transacción a ser procesada es por omisión, primera leída. Esto puede cambiarse, y la transacción no puede ser la primera procesada. Esto permite un flujo de transacción a ser procesada en partes, o la repetición de run desde un punto determinado. La sintaxis es;

START N

en donde N es un número entero.

Los números se refieren a registros físicos, entonces si un registro lógico está compuesto de varios registros físicos, esto debe multilicarse entre el número start. Por ejemplo, si 4 registros físicos de 80 caracteres, cada una compone una transacción lógica, entonces. La transacción # 20 comienza en el 4x 20 + 1 = 81 registro físico.
Entonces se usaría el START 81.

STOP

La útima transacció procesada partilcularmente para propósitos de prueba, se pueden especificarse por:

STOP N

en donde N es un número entero

Este número registra "físico' a ser leído'.

EJEMPLO:

```
MODIFY FILE CARS
FORM COUNTRY/10 CAR/10 MODEL/20  RCOST/6
MATCH *
START 80
STOP 200
DATA ON NEWTAPE
END
```

## CASO LOGICO

## INTRODUCCION

Con frecuencia, cuando se describe una situación de procesamiento de transacción se usan palabras como : ~ en este caso,,, en el otro caso...! Lo que significa es que un grupo separado de procesos lógicos se deben de ejecutar cuando un registro particular de circunstancias aparecen. Los subcomandos ~MODIFY' explicaron, hasta ahora, implícitamente como se permitió hacer la mayoría de estos procesos lógicos por separado. Esto hace que el MODIFY tan fácil de usar y sin embargo tan poderosa en capacidad. Sin embargo, conforme el proceso se convierte más compleja, es importante ser capaces de dividirlos en procesos sub-modify separados más explícitos. Esto mantiene la complejidad dentro de territorios razonables dividiéndolo la lógica en sub-secciones identificables los cuales usan MODIFY aprendidos hasta ahora.

## Los Subcomandos CASE y ENCASE.

Cada una de las secciones del sub- modify son llamadas ~caso' y el procedimiento se mueve de un caso a otro. Los subcomandos para identificar el principio y el final de una cosa son:

```
CASE nombre de caso
subcomando
subcomando
.
.
ENDCASE
```

El nombre de caso es cualquier registro de 1 a 12 caracteres exceptuando los caracteres especiales de ~+ - / * & $ ' . El caso principal o caso de control está siempre arriba del procedimiento, así es que se le da el nombre por omisión , de ~TOP' , Entonces el nombre del caso no tiene que ser mencionado.

## Reglas del caso

Cada caso está compuesto de subcomando MODIFY y deben ser capaces de pararce por si solas como un MODIFY y no dar mensajes de error. En particular, si un caso contiene subcomandos ON MATCH o ON NOMATCH, o cualquier referencia a campos D. ( o a composit . en FIDEL ) debe contener un subcomando MATCH en el mismo caso.

Los casos no pueden ser agrupados. Esto es, la palabra ENCASE debe aparecer antes de comenzar otro caso.

Al final del proceso de un caso, si no se ha dado ninguna instrucción, de donde ir entonces el caso principal o `TOP' es entrado.

La tarea de la captura de datos puede mezclar el PROMPT, y FREEFORM en diferentes casos, en el mismo procedimiento, y de la misma manera mezclar FIXFORM y CRTFORM pero no una de cada una (i.e., PROMPT y FIXFORM).

Ciertos subcomandos globales MODIFY como START, STOP, LOG, DATA y CHECK no son dependientes del caso. Permanece en efecto el último uso de estos subcomandos.

Sólo se reservan dos nombres de casos. `TOP' significa la raiz o caso principal. No tiene que aparecer, trabajará el GOTO TOP . De igual manera, el nombre de caso ` EXIT' no debe aparecer, en otro lugar que en un GOTO EXIT, en donde el procedimiento es terminado como si se hubiera alcanzado el número de transacción del subcomando STOP.

## El caso TOP

El FOCUS proporciona el nombre de omisión TOP para el primer caso en el MODIFY, Los comandos de utilidad ALL y cualquier subcomando no asignado es colocado en este caso TOP.

## Procesado secuencias

El FOCUS adelanta un procdimiento MODIFY al principio del caso TOP, le transfiere control a otros casos de acuerdo a las subcomandos GOTO que puede tomar una de las siguientes formas.

```
GOTO CASENAME
IF expresión GOTO nombre de caso;
IF expresión THEN GOTO  nombre de caso ELSE GOTO nombre de caso;
IF expresión THEN GOTO nombre de caso ELSE ILF expresión
```

```
ON MATCH
ON NOMATCH
ON NEXT       GOTO  nombre de caso
ON NONEXT     if expresión...;
ON MATCH / NOMATCH

ON VALID GOTO nombre de caso
ON INVALID GOTO   nombre de caso
```

En donde `nombre de caso' es el nombre asignado a un caso en el procedimiento y ` expresión ' es cualquier expresión lógica legítima, usando valores de transacción , campos computados y/o valores de base de datos (prefijos con D.),

Los subcomandos GOTO son ejecutados en el orden en que hayan sido escritos. Si la secuencia de ejecución alcanza el final de un caso sin seguir un subcomando GOTO, MODIFY automáticamante le transfiere al caso TOP (y no al caso siguiente en secuencia).

E nombre EXIT se reserva y no podrá ser usado como un nombre de caso. El subcomando GOTO EXIT finaliza el procedimiento y es útil cuando se desea tener una manera de parar (detener) el procedimiento antes de alcanzar la condición "end- of- file' en los datos de transacción el cual es el punto normal de terminación.

## Mantenimiento del registro con casos Múltiples

La lógica de casos proporcionaran opciones de mantenimiento de registros incluyendo la habilidad de actualizar varios registros paralelos de una transacción sencilla de manera que no puedan ser obtenidos con grupos repetitivos en un caso sencillo y la habilidad de incluir, actualizar y suprimir registros en un procedimiento.

## Estableciendo una posición en el archivo

Un caso que contiene cualquiera de los subcomandos ON MATCH, ON NOMATCH , ON NEXT o ON NOMATCH deben ser precedidos por un subcomando MATCH o NEXT en el mismo caso. Esto también es cierto si el caso contiene cualquier referencia a un campo D. ó T..

En otras palabras, cada uno debe tomar la forma de un procedimiento MODIFY legal.

Si en su ejecución, el procedimiento establece una posición sin un segmento pariente en un caso, entonces se ramifica a un segundo caso, el segundo caso puede establecer una posición en un segmento hijo re- establecer la posición en un segmento hijo sin re- establecer la posición en el pariente.

## Concordando segmentos múltiples en un comando sencillo

Aún y cuando un subcomando MATCH pueda referirse a campos en varios segmento, en ocasiones necesario en un procedimiento con casos múltiples concordar en cada segmento separadamente.

Cuando varios segmentos son concordados en un subcomando MATCH , un subcomando `ON NOMATCH INCLUDE' toma efecto cuando el primer segmento es identificado A menos que un subcomando (ON MATCH GOTO' sea dado explícitamente a cada nivel en la estructura de datos, el control es regresado automáticamente al caso TOP . Considèrese los siguiente ejemplos:

EJEMPLO:

```
>> MODIFY FILE CARS
>> FIXFORM COUNTRY/10 CAR/10 MODEL/20 BODY/5 SEATS/3
>> MATCH COUNTRY CAR MODEL BODY
>>      ON NOMATCH INCLUDE
>>      ON NOMATCH GOTO SECOND
>>      ON MATCH GOTO SECOND
>> CASE SECOND
>>      FIXFORM RCOST/6 DCOST/6
>>      COMPUTE
>>      BODY=BODY;
>>      MATCH BODY
>>      ON MATCH UPDATE RCOST DCOST
>> ENDCASE
>> DATA ON INREC
>> END
```

El procedimiento MODIFY procesa cada uno de los segmentos de la estructura de datos a cambio, e incluirá los datos tan pronto como sean identificado un instrucción NOMATCH. La siguiente versión del mismo procedimiento demuestra explícitamente la lógica.

```
>> MODIFY FILE CARS
>> FIXFORM COUNTRY/10 CAR/10 MODEL/20 BODY/5 SEATS/3
>> MATCH COUNTRY
>>     ON NOMATCH INCLUDE
>>     ON NOMATCH GOTO TOP
>>     ON MATCH CONTINUE
```

```
>> MATCH CAR MODEL
>>     ON NOMATCH INCLUDE
>>     ON NOMATCH GOTO TOP
>>     ON MATHC GOTO SECOND
>> MATCH BODY
>>     ON NOMATCH INCLUDE
>>     ON NOMATHC GOTO SECOND
>>     ON MATCH GOTO SECOND
>> CASE SECOND
>>     FIXFORM RCOST/6 DCOST/6
>>     COMPUTE
>>     BODY=BODY;
>>     MATCH BODY
>>     ON MATHC UPDATE RCOST DCOST
>> ENDCASE
>> DATA ON INREC
>> END
```

Nótese que con la sintáxis original, una transacción que falla al encontrarle una concordancia a COUNTRY, CAR o MODEL se incluye, pero para tales transacciones, el CASE SECOND no es procesado. Si la intención es de procesar el CASE SECOND despues de cualquier condición NOMATCH, entonces los segmentos separados tienen que ser concordados por separado, y el comando tiene que ser incluído en cada nivel.

```
ON NOMATCH GOTO SECOND
```

```
>> MODIFY FILE CARS
>> FIXFORM COUNTRY/10 CARS/10 MODEL/20 BODY/5 SEATS/3
>> MATCH COUNTRY
>>     ON NOMATCH INCLUDE
>>     ON NOMATCH GOTO SECOND
>> MATCH CARS MODEL
>> ETC..
```

Activando Campos

En un procedimiento MODIFY, el FOCUS sigue huella de cuales campos y segmentos en un archivo de datos estén activos para el mantenimiento de datos en cualquier momento. Los subcomandos MATCH, INCLUDE, UPDATE y DELETE solo se aplican a los campos que están activos en el momento.

Los siguientes subcomandos activan un campo: Cualquier FIXFORM, FREEFORM, PROMPT o CRTFORM que hace referencia a un campo en el archivo MAESTRO. Cualquier COMPUTE o VALIDATE que asignan valores a un campo en el archivo MAESTRO. Los siguientes subcomandos in-activan un campo: Un INCLUDE in-activa todos los campos y segmentos, Un UPDATE in-activa todos los campos en listados en el subcomando UPDATE. Un GOTO TOP implicito o explicito in-activa todos los campos y segmentos.

FOCUS necesita mantener la huella de campos activos e inactivos para manejar aplicaciones con datos faltantes y procesar precisamente aquellos cambios para los cuales existen datos disponible.

Sin embargo, es algunas veces necesario reativar los campos o segmentos explícitmente, la necesidad de una acción explícita sigue a un INCLUDE que acaba de in-activar todos los campos y el procedimiento continua mandando otras acciones en uno de los segmentos recientemente incluídos. Para re-establecer una posición en el archivo de datos, se necesita un nuevo comando MATCH y antes del nuevo MATCH se tiene que reactivar el campo donde se hace el MATCH. Entonces en el ejemplo anterior, el comando

```
COMPUTE
BODY = BODY;
```

Fue incluído, para reactivar el campo BODY, antes de concordar en el nuevo.

Fuente de Datos con Casos Múltiples

El uso de la lógica del CASE proporciona una flexibilidad adicional en el uso de fuentes de datos, ya que le permite a los usuarios procesar diferentes partes de una transacción sencilla, de diferentes maneras y el de mezclar datos de fuentes múltiples en un procedimiento de formas no posibles en un CASE sencillo.

Procesamiento Parcial de Transacciones FIXFORMS

Cuando el MODIFY se encuentra inicialmente con una transacción FIXFORM lee un registro completo del archivo de transacciones de entrada (especificado por el subcomando DATA ON dd nombre). El subcomando inicial FIXFORM puede procesar todo o solo parte del registro y subcomandos subsecuentes FIXFORM puede continuar procesando el mismo registro (y aun partes de él mas de una vez). Mientras que un subcomando subsecuente FIXFORM no defina un campo extendiendose mas alla del final del registro, el FOCUS no leerá otra transacción hasta que el control no regrese al CASE en donde un FIXFORM fue ejecutado primeramente y todos los grupos repetidos hayan sido procesados. Si un subcomando FIXFORM lee mas alla del final de una transacción o si control es transferido el CASE con el FIXFORM inicial, entonces MODIFY lee la nueva transacción.

Esta facilidad se aplica solo a transacciones FIXFORM primaria, aquel especificado por 'DATA ON ddnombre'. No se aplica a flujos de transacciones que son leídos por los subcomandos 'FIXFORM ON ddnombre'. Cualquier grupo nuevo de subcomandos 'FIXFORM ON ddnombre' adyacentes resultan en un registro nuevo que siempre es leído.

Nótese que para saltarse hacia atras sobre parte de una transacción, el 'X-N' no debe ser especificado al final de un subcomando FIXFORM. En lugar, lo coloca al principio del siguiente FIXFORM, evitense tales contruciones como:

```
FIXFORM COUNTRY/10 CARS/10 MODEL/20
```

Mezclando tipo de Transacciones

MODIFY les permite a los subcomandos FREEFORM y PROMPT ser mezclados en un procedimiento sencillo o en los subcomandos FIXFORM y CRTFORMS. Estas combinaciones son permitidas en un CASE sencillo, pero la disponibilidad de la lógica del CASE se expande del rango de acciones que el FOCUS puede manejar.

Las combinaciones del FIXFORM y CRTFORM proporciona un alcance mayor para el diseñdor del procedimiento, sujeto a las siguientes reglas:

1.- Se permite un subcomando FIXFORM . Tiene que especificar el nombre del archivo de transacción(FIXFORM ON ddnombre) y tiene que aparecer en el procedimiento MODIFY antes del primer subcomando CRTFORM.

2.- Se permite hasta 20 CRTFORMS en una pantalla (como en cualquier procedimiento FIDEL).

3.- Cuando menos un CRTFORM tiene que ser procesado antes de que cualquier comando TYPE o cualquier otro mensaje sean generados.

4.- Si las transacciones no se aputan, el largo del registro del archivo LOG tiene que ser mayor de las longuitudes del registro asociado con el FIXFORM y éste asociado con todos los CRTFORMS.

5.- No se aplican START y STOP.

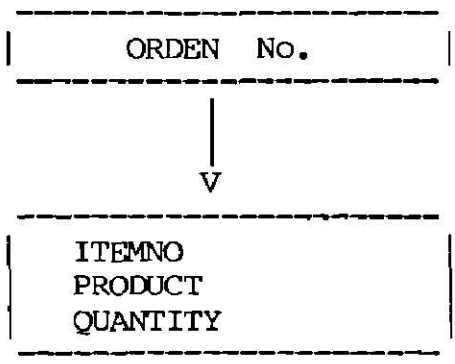
Ejemplos de Lógica de CASE

En el primer ejemplo el operador indica uno de tres valores, una "A" que significa agregar un registro "D", que significa suprimir un registro o una "C" para cambiar un registro. El procedimiento cambia al caso apropiado.

EJEMPLO: LOGICA DE CASE

```
MODIFY FILE CARS
COMPUTE
MYCASE/A1=;
COMPUTE
RECCNT/I14=RECCNT+1;
IS RECCNT EQ 4 THEN GOTO EXIT;
PROMPT MYCASE
      IF MYCASE EQ 'A' THEN GOTO ADDCASE;
      IF MYCASE EQ 'B' THEN GOTO CHGCASE;
      IF MYCASE EQ 'C' THEN GOTO DELCASE;
CASE ADDCASE
   TYPE "ADDCASE"
   PROMPT COUNTRY CAR MODEL
   MATCH COUNTRY CAR MODEL
     ON NOMATCH INCLUDE
     ON MATCH CONTINUE
ENDCASE
CASE CHGCASE
   TYPE "CHGCASE"
   PROMPT COUNTRY CAR MODEL BODY RC
   MATCH COUNTRY CAR MODEL BODY RC
     ON NOMATCH REJECT
     ON MATCH UPDATE RC
ENDCASE
CASE DELCASE
   TYPE "DELCASE"
   PROMPT COUNTRY CAR
   MATCH COUNTRY
     ON NOMATCH REJECT
   MATCH CAR
     ON NOMATHC REJECT
     ON MATCH DELETE
ENDCASE
DATA
```

El siguiente ejemplo es una versión simplificada de una necesidad común. Serán desplegados un grupo de registros en lugar de uno sencillo; entonces ocurrira alguna acción en uno o mas de ellos. El diseño del archivo tine dos segmentos, un número de orden seguido por líneas de párrafos múltiples para cada orden.




El procedimiento indica un número de orden, despues despliega todas las líneas de párrafos, productos y cantidades para esa orden. Despues pregunta si existen cambios a la QUANTITY y si es asi, la línea del párrafo el cual hay que cambiar el campo de QUNATITY. El procedimiento se muestra abajo y una muestra de la ejecución en el ejemplo que le sigue:

```
MODIFY FILE PROMTEST
COMPUTE
UPD/A1=;
ITEMNO=1;
PROMPT ORDERNO
MATCH ORDERNO
   ON NOMATCH REJECT
   ON MATCH CONINUE
   ON MATHC GOTO TWO
CASE TWO
MATCH ITEMNO
   ON MATCH TYPE
      "ITEM <ITEMNO PRODUCT<D.PRODUCT QUANTITY<D.QTY "
   ON MATCH COMPUTE
      ITEMNO=ITEMNO+1;
   ON MATCH GOTO TWO
   ON NOMATCH GOTO THREE
ENDCASE
CASE THREE
PROMPT UPD.DO YUO WANT TO CHANGE ANY QUANTITY "
IF UPD EQ 'N' THEN GOTO TOP;
PROMPT ITEMNO QTY
MATCH ITEMNO
   ON MATCH UPDATE QTY
   ON MATCH GOTO THREE
   ON NOMATCH GOTO THREE
ENDCASE
DATA
```

EJEMPLO: EJECUCION DEL PROCEDIMIENTO PREVIO

```
DATA FOR TRANSACTION      1

ORDERNO              => 12345
ITEM                 1 PRODUCT APPLES QUANTITY  300
ITEM                 2 PRODUCT PEARS  QUANTITY  400
DO YOU WANT TO CHANGE ANY QUANTITY> N

DATA FOR TRANSACTION      2
```

Nótese la importancia del caso TOP; cuando diferentes casos piden datos, no es obvio lo que constituye una transacción. La regla general que sigue FOCUS es de cuando se introduce el caso TOP el contador de transacciones es siempre incrementado.

Reconstruyendo un archivo FOCUS - Utilizando el Comando REBUILD

El comando REBUILD proporciona opciones para recontruir un archivo FOCUS, reorganizando un archivo FOCUS y para indicar los campos en un archivo FOCUS.

1.- Un archivo FOCUS dejando todos o algunos segmentos seleccionados a un espacio de trabajo y despues recargandolos. El nuevo archivo es compactado mediante la renovación de espacio suprimido no usado y almacenes adyacentes de registros lógicamente secuenciados.

2.- Un archivo FOCUS es reorganizado dejando todos o algunos segmentos seleccionados conforme vayan existiendo o recargandolos bajo una nueva descripción de archivos. La nueva descripción de archivos puede:

a).- Agregar nuevos campos de datos al final de los segmentos existentes.
b).- Remover campos de datos.
c).- Cambiar el orden de los campos de datos dentro de un segmento o degradar campos de datos a segmentos descendientes.
d).- Agregar nuevos segmentos como descendientes de segmentos existentes.
e).- Remover segmentos.
f).- Indicar campos diferentes.
g).- Diminuir el tamaño de un campo de datos alfanumérico.
h).- Promover campos de un descendiente único a su segmento pariente.

3.- Un campo en un archivo FOCUS es indicado despues de que los datos hayan sido entrados, cambiando la descripción del archivo MAESTRO y colocando el FIELDTYPE=I para el campo a ser indicado. Esta facilidad permite que la conexión interarchivos sea ejecutada para los archivos FOCUS existentes, para los cuales no fue proporcionado al principio.

De igual manera, el indicar un archivo despues de haber cargado inicialmente los datos, puede desearse porque los procesos separados de la carga de datos y después indicarlos, puede ser mas rápido que la ejecución de ambos procesos durante el mismo tiempo que dura la creación de un archivo.

a.- Se presentan cuatro opciones cuando se imprime el comando. Estas son REBUILD, REORG, INDEX, CHECK. La opción REBUILD realiza tanto el DUMP y LOAD y existe cuando se recontruye un archivo se recomienda crear una compia de refuerzo conforme sean escritos los datos originales. La opción REORG pide las fases LOAD o DUMP y ejecuta ambas por separado bajo el nombre del archivo proporcionado en ese momento. La opción INDEX puede que el campo sea indicado y contruye un ídice para ese campo.

b.- El espacio de trabajo no tiene que ser un FILEDEF o ALLOCATED antes de entrar al REBUILD. FOCUS imprimirá su propio FILEDEF para este propósito, sin embargo tiene que haber suficiente espacio de trabajo para el uso temporal del almacenaje del disco adherido. Como una regla, se debe tener un espacio de 10% a 20% mas grande que el tamaño del archivo existente disponible para las opciones REBUILD y REORG.

La opción INDEX usa el sistema de sorteo de programas y debe de disponerse de espacio para escribir en el disco.

c.- Las condiciones de prueba usadas para seleccionar los registros deseados usa la misma sintáxis que el subcomando LOCATE del modo FOCUS SCAN. Las relaciones de prueba son: EQ, NE, LE, GE, LT, GT, CO (CONTINE), OM (OMITE), Las pruebas son conectadas con la palabra AND y pueden conectarse OR . Una coma seguida de por un signo de dolar termina la prueba que puede ocupar varias líneas, por ejemplo:

```
A EQ A1 OR A2 AND B LT 100 AND
C GT 400 AND D CO 'CUR',  $
```

d.- Los archivos protegidos mediante el Paquete de Administración de Datos (DAP) requiere que la palabra clave DBA sea impresa antes de cualquier actividad REBUILD.

e.- Cuando la opción INDEX es usada, ya sea que un campo especifico sea indicado o que todos los campos que tengan descripción MAESTRA con FILEDEF=I sean indicados. Este último es útil al cargar inicialmente datos, ya que la carga de datos sin índice es mas rápido contruírlos todos al mismo tiempo.

USO:

El comando es impreso es FOCUS escribiendo:

```
REBUILD
```

EJEMPLO:

```
>> REBUILD
    ENTER OPTION (REBUILD, REORG, INDEX or CHECK)=> REORG
    ENTER THE REORG PHASE (DUMP OR LOAD)=> DUMP
    ENTER NAME OF FOCUS FILE? (FN FT FM)=> CARS FOCUS
    ANY RECORD SELECTION TESTS? (YES/NO)=> NO
    STARTING

    FILEDEF REBUILD DISK  REBUILD FOCTEMP A4 (LRECL 00064 BLKSIZE
    06404  RECFM  VB
    NUMBER OF SEGMENTS RETRIVED = 102
>> REBUILD
    ENTER OPTION (REBUILD, REORG, OR INDEX)=> REORG
    ENTER THE REORG PHASE (DUMP OR LOAD)=> LOAD
    ENTER NAME OF FOCUS FILE (FN FT FM)=> NEWCARS FOCUS
    STARTING
    $NUMBER OF SEGMENTS INPUT = 102
>>
```

EJEMPLO 2:

```
>> REBUILD
    ENTER OPTION (REBUILD, REORG OR INDEX)=> REBUILD
    ENTER NAME OF FOCUS FILE (FN FT FM)=> CARS FOCUS
    ANY RECOD SELECTION TESTS? (YES/NO)=> NO
    STARTING...

    FILEDEF REBUILD DISK   REBUILD FOCTEMP D4 (LRECL 0064 BLKSIZE
    06404  RECFM  FB
    $NUMBER OF SEGMENTS RETRIEVED = 102
    NUMBER OF SEGMENTS INPUT = 102
    FILE HAS BEEN REBUILT
```

EJEMPLO 3

```
>> REBUILD
    ENTER OPTION (REBUILD, REORG OR INDEX)=>INDEX
    ENTER NAME OF FOCUS FILE (FN FT FM)=> NEWCARS FOCUS
    ENTER NAME OF FIELD TO INDEX (OR * FOR ALL)=> BODY
    STRATING....

    (FOC319)  WARNING...FIELD IS INDEX AFTER FILE CREATED?:
    BODY
    INDEX VALUES RETRIEVED= 18
    $SORT COMPLETE...RET CODE 0
    INDEX INITIALIZED FOR:BODY
    INDEX VALUES INCLUDED=
>>
```

Anotando Procedimientos MODIFY

Si un asterico es el primer caracter no-blanco en una línea en el flujo del subcomando MODIFY, entonces, la línea es ignorada. Entonces, un procedimiento que es catalogado, por ejemplo, puede ser anotado con comentarios que sirven para documentar el procedimiento.

```
MODIFY FILE CARS
*
* THIS PROCEDURE MATCHES EXISTING CARS
* AND UPDATES VARIOUS ITEMS OF SPECIFICATIONS
*
MATCH COUNTRY/10 CAR/10 MODEL/20
*
* NOTE ONLY EXISTING RECORDS ACCEPTED
*
ON NOMATCH REJECT
LOG NOMATCH ON BADIES
ON MATCH UPDATE LENGTH WEIGHT WBASE
ON MATCH UPDATE MPG SECONDS BHP
DATA ON NEWSPEC
END
```

Borrando un Archivo FOCUS

En MS-DOS

Un archivo FOCUS es un archivo MS-DOS y puede ser borrado imprimiendo el comando DOS ERASE. Por ejemplo, para borrar un archivo FOCUS llamado PRODUCTS FOCUS en el disco A, el comando es:

```
DOS ERASE A: PRODUCTS.FOC
```

Editor Interactivo (SCAN)

Uso del SCAN

El estatuto SCAN permite observar y editar archivos FOCUS mediante el uso de subcomandos:

Con SCAN se puede:

1.- Agregar registros a archivos FOCUS ya sean nuevos o ya existentes.
2.- Borrar registros de un archivo FOCUS.
3.- Cambiar los valores de los campos en un archivo FOCUS.
4.- Búsqueda en archivos FOCUS basada en un cierto criterio.
5.- Desplegar el contenido de los registros de un archivo FOCUS mostrando ya sea todos o un subconjunto de los valores de los campos.
6.- Mover segmentos de registros y todos sus descendientes desde un padre a otro cuñado el archivo FOCUS tiene estructura de un árbol.

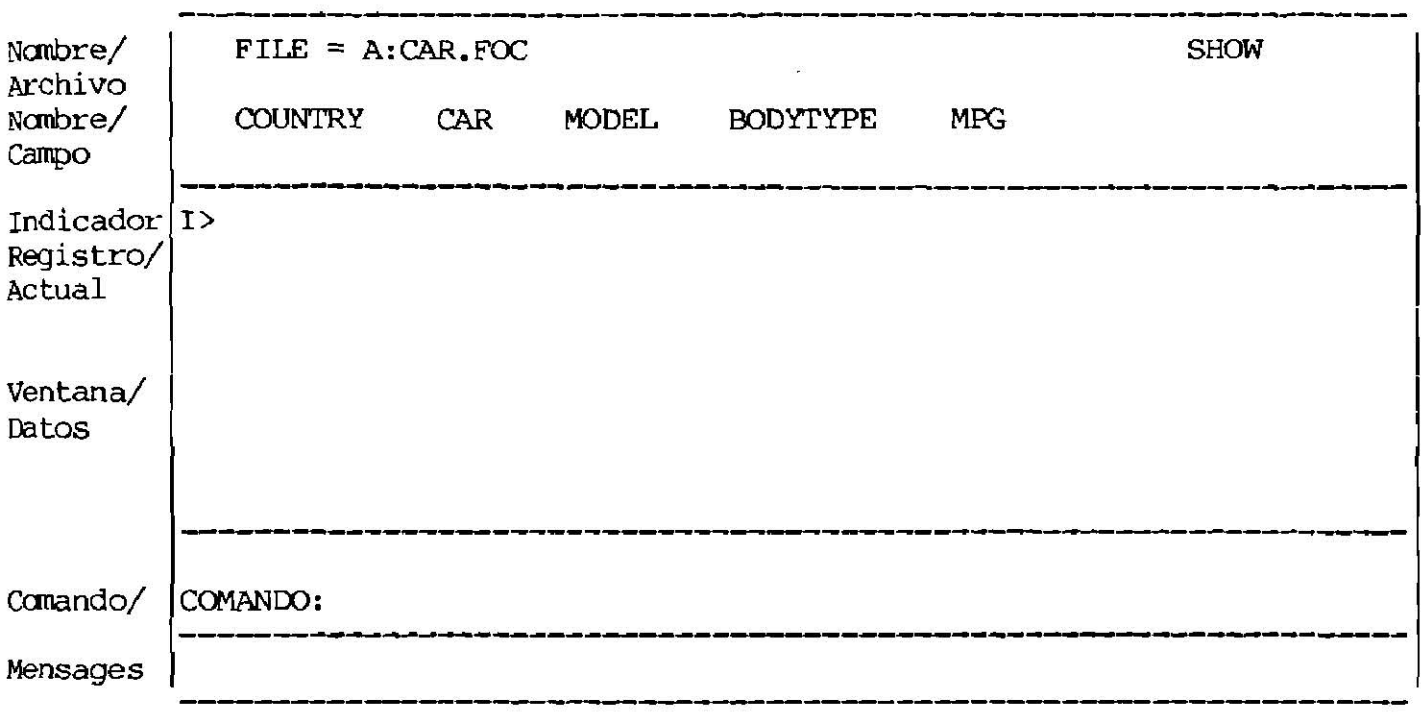
Entrar a SCAN

Se entra a las facilidades del SCAN tecleando:

```
>> SCAN FILE Nombre del Archivo
```

Donde nombre del archivo es el nombre del archivo FOCUS.

La pantalla es formateada presentando un area de control y una ventana. El nombre de cada campo del archivo aparecerá en la parte superior de la pantalla. Y en la parte inferior se encuentra el area de comandos. Por ejemplo, si se teclea, SCAN FILE CAR, la pantalla aparecera como:




Si se quiere recorrer archivos FOCUS se puede hacer utilizando cualquiera de los comandos del SCAN. Por Ejemplo, si se teclea la letra 'N' para el subcomando NEXT en el area de comandos y se presiona ENTER, los datos del primer registro aparecerán. También se puede dar un número despues del subcomando 'NEXT'.

Controlando la Pantalla

Los nombres de los campos en la pantalla superior de la ventana son seleccionados por default y estos pueden ser mostrados en dos pantallas (160 caracteres). Se puede cambiar la selección de los campos a desplegar usando el subcomando SHOW. Tecleando en el area de comandos.

SHOW Campo Campo .....Campo

Por ejemplo, si se teclea:

SHOW CAR MODEL MPG RETAIL.COST SALES

los datos de estos 5 campos aparecerán.

Si se teclea:

SHOW *

Todos los campos del archivo seran desplegados.

Si se teclea:

```
SHOW CAR * MPG
```

Todos los campos desde CAR hasta MPG inclusive, son desplegados.

Si el area necesaria para desplegar los campos excede a los 80 caracteres un flecha derecha aparecerá en la esquina inferior derecha de la línea del estatuto. Y se podrá obtener otra pantalla especial de 80 caracteres usando las llaves especiales.

Las llaves especiales que permiten moverse a través de la pantalla son:

1.- Moverse un caracter a la derecha es: F10
2.- Moverse una página posteior es: F10 y ALT
3.- Moverse un caracter a la izquierda es: F9
4.- Moverse una página anterior es: F9 y ALT
5.- Borra todo desde la posición actual del cursor hasta el final de la línea: END

Desplegado en Forma de Filas

La manera normal de desplegar los campos de un archivo FOCUS es presentar hasta 14 registros en la ventana de datos de la pantalla. Pero a veces es conveniente poder mostrar cada campo (o un subconjunto de campos) de un solo registro a la vez. La forma de desplegar los datos por hileras se puede obtener usando el comando SCAN:

```
CRTFORM Campo Campo .....Campo
```

o solo:

```
CRTFORM
```

Cuando ya se han especificado los campos a mostrar el estatuo SHOW. Para regresar al desplegado normal se hace tecleando el comando SHOW.

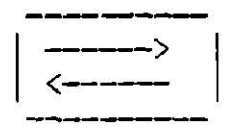
Por ejemplo, el comando CRTFORM COUNTRY CAR MODEL BODY MPG seguido del comando NEXT cambia la pantalla la forma:

```
FILE A:CAR.FOC                                          CRTFORM
-------------------------------------------------------------

COUNTRY    :   ENGLAND
CAR        :   JAGUAR
MODEL      :   SXJ 4 DOOR
BODY       :   SEDAN
MPG        :   22


-------------------------------------------------------------

COMANDO:

-------------------------------------------------------------
```

Actualizando Mediante la Forma de Desplegado por Hileras

Cuando en parte superior derecha de la pantalla aparece CRTFORM esto permite posicionar el cursor directamente en cualquier dato mostrado en la ventana y reemplazar su valor. La tecla

HOME Se posicionará en el primer campo y la tecla




permitirá moverse rápidamente de un campo a otro.

Si se requiere de modificar datos en vez de poderlos observar en la pantalla la forma CRTFORM resulta mas conveniente.

Si se tiene mas de 17 lineas puede moverse para arriba y para abajo estando en la forma CRTFORM usando las teclas:

1.- F8 para moverse para abajo.

2.- F9 para moverse para arriba.

Reteniendo Comandos

Es a veces conveniente repetir el comando antes usado. Normalmente los comandos desaparecen del area de comandos despues de ser ejecutado. Se puede hacer que el comando permanezca en el area de comandos repitiendo su primer letra. Por ejemplo:

Next

Los pueden ser abreviados a su mas pequeña forma que generalmente es un caracter, se puede teclear fácilmente

NN

para repetir el comando NEXT

Por ejemplo, presionando ENTER se puede mover de registro a registro dentro del archivo. Similarmente, el comando LL ( repetira el comando LOCATE), es recomendable cuando se requiere encontrar registro tras registro los cuales cumplan con una condición dada. Por ejemplo:

LL MPG GT 21

Concepto de Posición Actual

Un archivo FOCUS tiene que estar secuencialmente un registro seguido de otro registro. Por lo tanto, la noción de archivos secuenciales no es aplicable en archivos FOCUS. Sin embargo un efecto similar se realiza viendo desde un aspecto jerárquico al archivo FOCUS y en la forma de derecha-izquierda, arriba-abajo.

Posicionamiento en Registros Específicos

Los subcomandos LOCATE o TLOCATE operan de la misma manera que como se hace en la estructura principal de FOCUS. Se dan las condiciones de selección en el area de comandos.

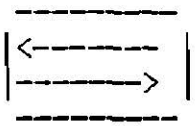
Por Ejemplo:

L MODEL CO AUTO AND BODY EQ SEDAN, $ *

El asterisco indica que todos los registros que cumplan con el criterio de busqueda son requeridos. Si el número de registros revaza a tener una ventana llena la palabra MORE aparecerá en la línea de estatus. Presionando ENTER se continuara desplegando los registros encontrados.

Cambiando Valores en Campos

Los subcomandos CHANGE y REPLACE trabajan de la misma manera que en la estructura principal de FOCUS excepto que en el registro en que se esta trabajando es indicado por una flecha en el frente de este al lado izquierdo de la pantalla. Cuando se le hace un cambio a un registro, todo el nuevo registro aparece en la pantalla y este pasa a ser el registro actual. Tal vez es mas fácil cambiar valores de los datos usando la forma CRTFORM. Para llegar fácilmente al valor que se desea cambiar se puede usar la tecla TAB.



y ya una vez que se ha cambiado el valor deseado se debe presionar la tecla ENTER.

Para cambiar valores de campos en archivos hay que entrar al área de comandos:

```
REP  KEY .....
0
CHA  KEY .....
```

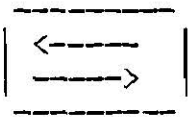
Los campos que son usados como llaves control del sorteo son de color mas intenso que los demás o tienen un asterisco en frente de sus nombres.

Borrando Registros

El borrar registros no es reversible para evitar accidentes, se debe teclear la letra "D" en el area de comandos seguida por el nombre de campos o segmento que se quiere borrar. El registro actual es indicado por una flecha al lado izquierdo, es el registro que fue borrado. En un archivo estructurado el segmento actual y todos sus segmentos hijos son borrados.

Agregando Registros

Al teclear la letra "I" en area de comandos y presionar la tecla ENTER. El cursor se posicionará en la linea inmediata anterior al área de comandos. Para moverse de campos a campo se puede usar la tecla TAB.



Por Ejemplo:

```
    COUNTRY   CAR  MODEL  BODY  MPG
-----------------------------------------

              X3J  SEDAN    36
-----------------------------------------

COMANDO:  I
-----------------------------------------
```

Después de terminar de insertar registros se debe presionar la tecla ENTER. El nuevo registro es movido a la ventana y pasa a ser el registro actual.

Borrando la Pantalla

Presionando las teclas CTRL y HOME

Se borrará el area de desplegados de la pantalla

Recuperación después de haber borrado una pantalla

Si se da el comando "TY" (TYPE) el registro actual es traído. Esto es usual después de haber borrado una pantalla.

Tecleando TY n (Donde n es Entero) se desplegará los siguientes n registros.

El Subcomando DISPLAY

El subcomando DISPLAY es usado para desplegar valores de los campos los cuales no son en listados en los campos actuales mostrados en el comando SHOW.

Si se tienen muchos campos de datos, los comandos X y Y puede ser usados para desplegar y almacenar sus valores. Por ejemplo:

X DIS CITY STATE ZIP

Cuando se requiere obtener estos valores almacenados en X tecleando X en el area de comandos hara que estos sean traidos. Tambien se puede tener otra lista de valores almacenada con el uso del comando "Y".

Terminando la Sección SCAN

Existen 3 maneras de terminar la sección SCAN. Esto es, se puede hacer tecleando cualquiera de los siguientes comandos en el area de comandos; END, FILE o QUIT. Las primeras 2 preservan los cambios hechos en el archivo y el otro no guardará los últimos cambios hechos al archivo que no han sido grabados en el disco.

Presiona la tecla F3, esta funciona como el QUIT.

SUMARIO DE LOS SUBCOMANDOS DEL SCAN

| SUBCOMANDO | PARAMETROS | FUNCION |
|---|---|---|
| AGAIN | | Repite el último subcomando. |
| BACK | | Regresa al registro marcado anterior. |
| CHANGE | Field= /string/srting/n | Cambia un string N veces. |
| DOS | | Pemite usar los comandos del DOS desde el SCAN. |
| DELETE | Field N | Borra N ocurrencias del campo. |
| DISPLAY | Field Field | Despliega valores de Campos. |
| FILE | | Termina la sección SCAN guardando todos los resultados. |
| INPUT | | Entra al modo INPUT. |
| JUMP | Field N | Brinca hasta la N.ava ocurrencia de un campo. |
| LOCATE | Field relation Value | Busca registros que cumplan una cierta condición. |
| MARK | | Marca un registro. |
| MOVE | Fieldname | Se mueve a otro segmento padre. |

| | | |
|---|---|---|
| NEXT | N | Moversae N registros. |
| ? | | Imprime el último subcomando. |
| QUIT | | Termina la sección SCAN sin guardar los cambios hechos. |
| REPLACE | Field=value | Reemplaza un valor de un campo. |
| SAVE | | Salva todos los campos. |
| SHOW | Field Field | Despliega una lista de campos. |
| TLOCATE | Field relation Value | Se va al inicio del archivo y después hace un LOCATE NORMAL. |
| TOP | | Se posiciona al inicio del archivo. |
| TYPE | | Despliega N registros. |
| UP | | Se va al primer registro del segmento padre. |
| X | | Asigna un subcomando SCAN o ejecuta uno. |
| Y | | Asigna un subcomando SCAN o ejecuta uno. |

DIALOGUE MANAGER

Introducción:

El propósito del Dialogue Manager es ayudar a los usuarios FOCUS en la ejecución de procedimientos los cuales son requeridos para diferentes usos.

Un archivo de procedimientos es el que se compone de comandos FOCUS los cuales son requeridos ya sea para obtener un reporte o para el mantenimineto de archivos. Asi en lugar de tener que teclear los comandos en la terminal, cada vez que se necesiten, el operador solo tendra que teclear el nombre del archivo que contiene los comandos.

Partes de estos procedimientos, son omitidos y se requiere de su complementación al tiempo de ejecución de estos. Esto permite que el operador pueda tener información variable y hace que el procedimiento sea mas flexible. Un uso tipico de esto puede ser un reporte, en el cual, se especifican el formato, encabezados, titulos, etc.. pero que requiere que el operador reeplazca las condiciones de selección de registros.

El proceso de hacer fácil el procedimiento de obtener estos valores requeridos durante el tiempo de ejecución es uno de los principales propósitos del Dialogue Manager.

El Dialogue Manager consiste de presentar al operador mensajes que le guiaran a que hacer, entonces el respondera a lo que se le pide, presentandose una revisión de las respuestas con el formato esperado.

Las variables omitidas pueden ser dadas después del nombre del procedimiento y de esta manera se evita el tener que estar pidiendo estos valores durante el tiempo de ejecución del procedimiento.

El Dialogue Manager provee facilidades para poner valores default a variables omitidas o permite hacer referencia a veriables del sistema como lo son la fecha y la hora.

Los procedimientos para ejecutar comandos FOCUS por el Dialoguer Manager son creados a través del uso del editor del sistema. Y el mantenimiento de ésto es hecho de la misma manera.

Nombre de los Archivos FOCUS

Los procedimientos FOCUS son todos FOCEXEC'S y pueden ser almacenados bajo cualquier nombre y tipo de archivo bajo MS-DOS. Sin embargo para hacer mas fácil la identificación de estos procedimientos se usará FEX.

El modo del archivo no es proveído, es tomado como "*", lo que significa que los valores default son tomados. Por lo tanto los archivos de procedimientos FOCUS deben de residir en discos centrales, accesibles a varios usuarios a la vez.

Ejecutando Archivos FOCUS

El Dialogue Manager es llamado desde FOCUS tecleando el comando EXEC (EX) seguido del nombre del procedimiento.

EJEMPLO:

>> EX LOCKUP  El procedimeinto llamado LOCKUP.FEX es ejecutado.

Si el tipo de archivo default (FEX) no es usado entonces el nombre completo y su tipo deben ser tecleados, por ejemplo, EX nombre del procedimiento.tipo del procedimiento.

EJEMPLO:

>> EX DAILY.RPT  El procedimiento llamado DAILY.RPT es ejecutado.

Procedimientos Omitidos

Un archivo compuesto de comandos FOCUS sin valores omitidos aparecerá exactamente igual que si se hiciera directamente, tecleando los comandos FOCUS en una terminal.

EJEMPLO: Procedimiento Almacenado

Un archivo DOS nombrado WEEKLY.FEX consiste de las lineas de abajo:

```
>> TABLE FILE CAR
>> " SUMMARY OF SALES...FOREIGN  CARS"
>> SUM SALES AND PCT.SALES AND COLUMN-TOTAL
>> BY CAR BY MODEL
>> END
```

Entonces el procedimiento se ejecuta de la siguiente manera:

```
>> EX WEEKLY
```

Si un archivo no es terminado con la palabra END todas las líneas son ejecutadas pero en la terminal se le presentaŕ al operador la oportunidad de agregar mas líneas a solamente especificar la finalización de este.

EJEMPLO: Procedimiento Parcial - END proporcionado por el operador

```
>> DEFINE FILE CARS
>> MARGIN=100*(RCOST-DCOST)/DCOST;
```

El procedimiento es ejecutado como:

```
>> EX COMP        Procedimiento pedido

>> GAS-RATION = MPG/WEIGHT ;     Operador sumando una línea
>> END                           Operador completa procedimientos.
```

Procedimientos con Valores Omitidos

Un archivo FOCUS puede contener variables las cuales se requiere que sean reeplazadas al tiempo de ejecución. Estos son valores señalados con un ampersand (&) como su primer caracter.

Reglas para nombres a las variables sustituibles:

1.- El primer caracter que procede al nombre de la variable esun ampersand (&).
2.- El número máximo de caracteres en el nombre, excluyendo el ampersand es de 12.
3.- No se permiten blancos intermedios en el nombre, ni caracteres especiales como (+,-,*,/,&,).

EJEMPLO: Procedimiento con Valores a Sustituir

```
>> SET MSG=OFF, PAGE=OFF
>> TABLE FILE CARS
>> " TOTAL UNITS SOLD BY COUNTRY OF ORIGIN "
>> "     BODY TYPE IS &BODYTYPE            "
>> SUM SALES AND PCT.SALES AND COLUMN-TOTAL
>> BY COUNTRY  IF BODY IS &BODYTYPE
>> IF SEATS FROM &NUMSEATS
>> END
```

Nombres de Variables Posicionales

Las variables sustituibles pueden ser asignadas a números en el orden en el que son encontradas y llamarlas simplemente, &1,&2,&3,etc... El número debe estar entre 1 y 225.

EJEMPLO: Variables Posicionales

```
>> MODIFY FILE CARS
>> FIXFORM CAR/10 COUNTRY/10 MODEL/10 RCOST/6
>> MATCH CAR COUNTRY MODEL
>> ON MATCH UPDATE RCOST
>> ON NOMATCH REJECT
>> LOG NOMATCH ON &1  MSG  &2
>> DATA ON &3
>> END
```

Reemplazando Valores para Variables Sustituibles

El Dialoguer Manager provee dos maneras de reemplazar valores para variables sustituibles.

1.- Tecleando valores después del nombre del procedimiento.

2.- Sustituyendo los valores que se requieran para las variables omitidas.

Reemplazando Valores en la Línea EXEC

Cuando el operador conoce los valores que requerirá durante la ejecución de un procedimiento, entonces, estas pueden ser dadas en la línea EXEC despues del nombre del procedimiento.

Se pueden tener:

Valores dados
Valores Obtenidos
Combinación de Valores Dados y Obtenidos

Continuando la Línea EXEC

Cuando la lista de de valores a reemplazar excede al ancho de la línea, usando una coma como último caracter de está, significara que la línea continuará en la sig. línea.

EJEMPLO:

```
>> EX LOOKUP  DIVISION=EAST COAST, UPLIMIT=40,
>> LOWLIMIT=30, DEST=OFF
```

Reglas para Sustituir Valores

Las variables las cuales son identificadas por sus nombres requieren del nombre seguido por un signo de igual (=) y seguido por el valor asignar. El conjunto de valores es separado por comas.

```
NAME=VALUE, NAME=VALUE, NAME=VALUE
```

Los valores a ser sustituidos pueden ser números, letras, etc. de cualquier longuitud. Pueden contener blancos intermedios o caracteres especiales, si estos contienen ya sea una coma o un signo de igual estos deben estar entre comillas.

```
>> EX NEWLIST ADDRESS='NY, NY', ZIP=11421
```

Valores Sustituibles

Cuando un procedimiento es ejecutado dentro del Dialoguer Manager, las variables precedidas por un ampersand (&), cuyos valores no han sido dados en la línea EXEC, seran presentados en la teminal y se necesitara que éstos sean proporcionados por el operador.

Hay tres maneras de pedir los datos:

1.- Valores Requeridos: Estos son pedidos en el momento en que se requiere dentro del procedimiento.

2.- Valores Directos: El comando PROMPT del Dialoguer Manager es usado para la obtención de estos, y se puede hacer uso de este donde sea necesario dentro del procedimiento.

3.- Entrada de Datos en Toda la Pantalla (CRT) : Una pantalla completa es presentada al operador con areas que debera llenar. Los valores son tomados al oprimir la tecla ENTER. Para controlar el formato de la pantalla se usara el comando CRTFORM del Dialoguer Manager.

Sintáxis para las Varibles Sutituibles

Una variable sustituible consiste en tres partes:

1.- Nombre de la Variable
2.- Longuitud y Tipo a Sustituir
3.- Texto del Mensaje a Desplegar

Cada uno de estos componenetes es separado por un punto. El formato y el texto son opcionales.

&Nombre.Formato.Texto

1.- Nombre de la Variable:

El nombre de una variable debe ser de 12 carcteres o menos. Este es precedido por un ampersand. No debe contener los caracteres especiales (+,-,*,/,=).

EJEMPLO:
&LOCATION aparecerá como LOCATION =
&COSTLIMIT aparecerá como COSTLIMIT =

2.- Formato de una Variable:

El formato está compuesto de dos partes, el tipo y la longuitud. Solo los tipos A o I son permitidos.

| TIPOS | LONGUITUD |
|---|---|
| A ALFANUMERICO | 1 a 255 caracteres |
| I ENTERO | 1 a 10 dígitos (2**31 - 1) |

EJEMPLO:

&COST.I5 Aparecerá como COST = el valor a reemplazar debe ser de 5 dígitos o menos.

&LOCATION.A3 Aparecerá como LOCATION = el valor a reemplazar debe ser de 3 caracteres o menos.

Si el valor a reemplazar viola el formato especificado, un mensaje de error es desplegado y se volvera a pedir el valor a reemplazar. Para que el operador pueda terminar el procedimiento debe sustituir el valor a reemplazar por QUIT.

3.- Texto del mensaje a pedir en valores sustituibles:

Si el texto es dado despues del nombre o del formato, entonces el texto es desplegado en lugar del nombre de la variable.

EJEMPLO:

```
&DIV.I4      Teclee 4 digitos para el número.
&LOC.A5      El nombre de la localización es:.
```

Contestación Implicada

Cuando una variable sustituible no tiene valor en el punto donde es requerido el Dialoguer Manager automáticamente pide el valor desde la terminal.

EJEMPLO: Contestación Implicada

```
>> SET PRINT=&PRINT, MSG=OFF
>> TABLE FILE CARS
>> SUM SALES AND COLUMN-TOTAL
>> ACROSS BODY-TYPE
>> BY DEALER IF CAR IS &CARNAME
>> IF 'RETAIL COST' EXCEEDS &FROMCOST
>> END
```

La sesión de la terminal es:

```
>> EX WHATSUP
   PLEASE SUPPLY VALUE REQUESTED
   PRINT= OFFLINE
   CARNAME=FIAT
   FROMCOST=3000
      .
      .
```

Preguntas Directas

Para pedir valores de variables no es necesario tener un procedimiento. Los valores pueden ser pedidos al momento en que sean necesitados. Esto es ejecutado mediante el uso del comando -PROMPT del Dialoguer Manager.

```
-PROMPT  &Name.Formato.Texto.
```

Cada pregunta es una línea

```
-PROMPT &DIV.A10. ENTER NAME OF DIVISION=
-PROMPT &DEST.A8. ONLINE OR OFFLINE PRINTING.
SET PRINT=&DEST
TABLE FILE SALES
SUM UNITS AND COLUMN-TOTAL
BY MONTH IF LOCATION IS &DIV
END
```

Uso de una Línea de Respuestas en el Modo Directo

La opción de poder pedir respuestas en modo directo en el Dialoguer Manager también permite tener una lista de posibles respuestas a ser especificadas en la línea de comandos. La sintáxis es:

```
_PROMPT &NAME.(LIST).TEXT.

-PROMPT &NAME.(&LIST).TEXT.
```

donde 'LIST' es el conjunto de posibles respuestas separadas por comas cada una y &LIST es una variable sustituible previamente asignada.

EJEMPLO:

```
-PROMPT DO WHAT. (SI,S,NO,N). DESEA CONTINUAR?.
```

Si el operador no responde con una de las posibles respuestas un mensaje de error es mandado, seguido de las posibles respuestas para ayudar al operador

Cuando una variable ampersand es usada para sustituir la lista de respuestas esta es inicializada con el comando -SET.

```
-SET &LIST = (YES,NO,OK,Y,N);
```

Puesto que el comando -SET puede ser extendido a muchas líneas es conveniente para aquellos casos donde un gran número de variables son necesitados. Es también conveniente cuando la misma lista es usada en mas de un estatuto -PROMPT.

La lista de variables es dada entre parèntesis. Si dentro del texto contiene parentesis, entonces se pueden usar las comillas para enmarcarlo.

Una manera alternativa de validar los valores a sustituir es usando el estatuto -IF. Este es necesario si una serie de valores numericos, o condiciones de prueba son dados.

Variables del Sistema

Una lista de variables son reservadas para el uso del sistema. Los valores para estos son sustituídos automáticamente al ser encontrados.

Hay 2 tipos de variables del Sistema y Variables Estadísticas.

| VARIABLES/SISTEMA | SIGNIFICADO |
|---|---|
| &DATE | Fecha actual. Esta desplegada como MM/DD/AA. |
| &TOD | Hora del dia. Esta es desplegada como HH:MM:SS. |
| &MDY | Fecha actual en la forma MMDDAA usuales comparaciones numéricas. |
| &ECHO | Modo de prueba. |
| &DMY | Fecha en forma DDMMYY. |
| &YMD | Fecha en la forma YYMMDD. |
| &IORETURN | Valor retornado despues de la última operación lectura/escritura en el Dialogue Manager. |

| VARIABLES ESTADISTICAS | SIGNIFICADO |
|---|---|
| &LINES | No. de lineas en el último reporte. |
| &RECORDS | No. de registros recuperados en el último reporte. |
| &TRANS | No. de transacciones hechas en el último procedimiento MODIFY. |
| &ACCEPTS | No. de transacciones aceptadas en el último procedimiento MODIFY. |
| &NOMATCH | No. de transacciones rechazadas por no coincidir. |
| &FORMAT | No. de transacciones rechazadas por error de formato. |
| &INVALID | No. de transacciones rechazadas por causa de una condición inválida. |
| &DUPLS | No. de transacciones rechazadas por duplicados. |
| &INPUT | No. de segmentos dados de alta. |
| &CHNGD | No. de segmentos actualizados. |
| &DELTD | No. de segmentos borrados. |
| &RETCODE | Valor retornado por el S.O. despues de un comando CMS. |
| &BASEIO | No. de operaciones de entrada/salida. |
| &READS | No. de lecturas fisicas desde un archivo externo. |
| &REJECTS | No. de otras transacciones rechazadas |

Variables Globales

Las variables ampersand desaparecen al final de un procedimiento. Los mismos nombres pueden por lo tanto ser usados de nuevo por otros procedimientos.

Ocasionalmente es usual preservar el valor de una variable sustituible para que un procedimiento FOCEXEC pueda comunicarse con otro; posiblemente un procedimiento que este dentro de otro. Otro uso de esto es el de tener un procedimiento al inicio de la sesión el cual recoge valores que seran usados por varios procedimientos que seran ejecutados despues. Esto ahorra la necesidad de volver a dar valores a las mismas variables en cada procedimiento. La preservación de valores es ejecutada usando el doble ampersand '&&' al frente del nombre de la variable, en lugar de tener un solo ampersand '&'. Debido a que sus valores son preservados a traves de los procedimientos, estas variables son llamadas globales (las variables ampersand son variables locales).

La sintáxis para una variable global es:

```
&&NOMBRE.FORMATO.TEXTO.
```

Donde el nombre puede estar compuesto de cualquier 10 caracteres o letras, números y los caracteres especiales excluyendo +,-,*,/,&. El formato y texto son partes opcionales. Si el texto es omitido, el nombre de la variable será usado al requerir de su valor.

Preguntando por los valores de las Variables Globales:
Debido a que las variables globales no pierden sus valores al terminar un procedimiento es conveniente poder desplegar estos valores. El comando interrogativo FOCUS (?) es usado.

Estatutos de Control del Dialoguer Manager

Un procedimiento puede estar compuesto de dos tipos de líneas. Aquellas que seran pasadas a FOCUS, y aquellas que son usadas solamente por el Dialoguer Manager. Los estatutos de control del Dialoguer Manager siempre aparecen con el signo (-) antes puesto como su primer caracter en la posición 1 o 2 de la línea.

| ESTATUTOS DE CONTROL | SIGNIFICADO |
|---|---|
| -* | Linea de comentarios, no tiene acción |
| -DEFAULTS | Poner valores iniciales para variables sustituibles. |
| -EXIT | Termina y ejecuta los comandos FOCUS |
| -GOTO | Salto incondicional. |
| -IF | Estatuto de Prueba y Salto. |
| -LABEL | Etiqueta que puede ser usada en un estatuto de salto. |
| -PROMPT | Despliega un mensaje y lee la respuesta. |
| -QUIT | Termina pero no ejecuta el procedimiento. |
| -RUN | Ejecuta estatutos FOCUS en este punto y regresa. |
| -SET | Asigna un valor a una variable. |
| -TYPE | Despliega un mensaje. |
| -TYPE1 | Despliega un mensaje y despues brinca una página. |
| -READ | Lee registros de un archivo externo. |
| -WRITE | Graba registros en un archivo externo |
| -CRTFORM | Inicializa el modo screen llemno del Dialoguer Manager. |

Conceptos de Ejecución

El Dialoguer Manager lee las líneas de un procedimiento, junto con los valores de las variables que serán sustituídas en el tiempo de corrida y las almacena para ser ejecutadas por FOCUS. Solo las líneas que empiezan con el símbolo '-' en la columna 1 o 2 no son almacenadas. Se asume que estas líneas son estatutos de control del D.M. o son etiquetas. Los estatutos de control son ejecutados inmediatamente, ya que su propósito es el de mejorar el almacenamiento de las líneas. Cuando un procedimiento sale, todas las líneas que fueron almacenadas son pasadas al procesador de estatutos FOCUS regular, como una alternativa para el flujo de entrada de datos. FOCUS se encargará de leer desde esta pila de almacenamiento hasta que se llena, entonces, ya sea que regrese a leer las entradas que definidas FILEDEF como que regrese al procesador EXEC. Si un estatuto de control -RUN es la causa de la salida del procedimiento y FOCUS esta listo para leer un nuevo estatuto, si está en el nivel comando, entonces el procedimiento re-entra al punto despues del RUN.

-GOTO etiqueta (GOTO incondicional)

Un salto incondicional es un estatuto del D.M. que empieza con -GOTO. La sintáxis es:

```
-GOTO label
```

Una etiqueta puede encontrarse ya sea antes o despues de un brinco dentro del procedimiento, cuando esta antes del brinco se llama BACKWARDS BRANCH. Si la etiqueta no es encontrada se genera un error y el procedimiento no es ejecutado.

EJEMPLO: Salto Incondicional

```
-TYPE WHEN FINISHED TYPE QUIT
-START
-PROMPT &SCREW. PRODUCT NUMBER=.
TABLE FILE PARTS
"PRODUCT NAME <PNAME ON &DATE"
"LOCATION     <LOC "
"INVENTORY    <INV
IF PRODCODE  IS   &SCREW
END
-RUN
-GOTO START
```

-IF comando (GOTO condicional)

La ejecución secuencial de un estatuto del D.M. tras otro puede ser alterada por uso de estatutos de salto. Un salto es una linea que empieza con un -IF. La sintáxis es:

```
-IF expression [THEN] GOTO [label] [ELSE GOTO label];
```

La cláusula ELSE es opcional, si no esta presente entonces la siguiente línea despues del branch es la localización por default del ELSE.

Expresiones Compuestas

Una expresión puede contener múltiples IF-THEN cada uno con su etiqueta:

```
-IF &COST GT 1000 GOTO HIGH ELSE IF &MPG
-LT 30 GOTO MID ELSE GOTO REG;
```

La continuación de una línea de muchas expresiones empieza con un signo (-).

Probando la Longuitud, Tipo y Existencia de una Variable

Un procedimiento en el que el procedimiento de salto es involucrado, o varias de las variables deben ser sustituídas en la línea inicial EXEC, entonces, es necesario poder probar cada valor sustituído para cada variable, y saber que tipo de valor tiene numérico o no-numérico, y que longuitud tiene el valor. Por ejemplo, habrá un error al hacer un cálculo numérico con una variable cuyo valor no es numérico.

Hay tres funciones de prueba. Estas son puestas despues del nombre de la variable a ser probadas.

Existencia de un Valor

Sintáxis:

```
&NAME.EXIST
```

El valor es un cero si no ha sido dado el valor de la variable y un valor de diferente de cero al ocurrir lo contrario.

EJEMPLO:

```
-IF &RCOST.EXIST EQ 0 GOTO ASKFOR
-ELSE GOTO HAVE;
```

Longuitud de un Valor

Sintáxis

```
&NAME.LENGHT
```

El valor es el número de caracteres que forman a la variable, o cero si no lo ha sido dado.

EJEMPLO:

```
-IF &DIVISION.LENGHT GT 15 GOTO NOTGOOD
-ELSE GOTO OK;
```

Tipo de Valor

Sintáxis

```
&NAME.TYPE
```

El resultado es una 'A' si es no-numérico de lo contrario es un 'N' numérico.

EJEMPLO:

```
-IF &DCOST.TYPE NE 'N' GOTO NOGOOD
```

Las caracteristicas de una variable global pueden ser probadas refiriendose a:

```
              LENGHT
&&NAME.       TYPE
              EXIST
```

Si todos los caracteres son válidos entonces se regresará un 'N' o un 'A' . El valor de existencia es cero si no se ha dado valor a la variable. La prueba de existencia puede hacerse de la siguiente manera:

```
-IF &&WHO.EXIST EQ 0 GOTO ONE ELSE TWO;
```

Comando TYPE

Los mensajes pueden ser desplegados desde un procedimiento del D.M. usando la palabra -TYPE al inicio de la línea

```
-TYPE texto
```

Las variables sustituibles pueden estar dentro de un texto.

```
-TYPE REPORT FOR &REGION ON &DATE
```

Comando SET

A las variables sustituibles pueden asignarseles valores que resultan de cálculos aritméticos o expresiones lógicas.

La sintáxis es:

```
-SET &NOMBRE=EXPRESION;
```

Un punto y coma debe de ir al final de la expresión. Si es necesario una expresión puede ocupar una o mas líneas.

EJEMPLO: Definiendo una Nueva Variable

```
-TYPE DO YOU WANT POSITIVE VALUES
-PROMPT &YN.ENTER YES OR NO.
-SET &VAL=IF &YN EQ 'YES' THEN 1 ELSE 0;
TABLE FILE POINTS
SUM NUMBER BY CLASS
IF POSITION IS &VAL
END
```

Terminación de un Procedimiento

-EXIT

Cuando la etiqueta -EXIT es encontrada la ejecución de las líneas del procedimiento es terminada y se comienza a ejecutar las líneas de expansión del procedimiento.

Si la última línea de un procedimiento es encontrada y no hay mas líneas un -EXIT implícito.

La etiqueta -EXIT es usual para terminar un procedimiento desde diferentes puntos, cuando los estatutos de salto son usados.

-QUIT

La etiqueta -QUIT causa la salida inmedita del procedimiento y las líneas que esperan en una pila para su ejecución no son ejecutadas.

Si la palabra FOCUS es agregada al QUIT

-QUIT FOCUS

Ocurre una salida inmediata de FOCUS. El código del sistema operativo es puesto en 8.

-RUN

La etiqueta -RUN causa la salida inmediata de un procedimiento. Las líneas que estan en la pila son ejecutadas, una vez terminado el procedimiento regresa a la línea despuéS del -RUN.

Escribiendo Datos en Archivos

-WRITE

Adicionalmente a manejar un diálogo entre operador-trminal el D.M. puede leer información desde archivos externos y a su vez escribir información con ellos.

La sintáxis para escribir en archivos externos al cual debe ser FILEDEF para poder usar es:

-WRITE ddname text

donde text es una combinación de variables ampersand y un texto.

ddname es el nombre del archivo al que se estará escribiendo.

EJEMPLO:

```
-WRITE WFILE &DIV &RED &TEST RESULT IS,
-&RECORDS AT END OF RUN
```

Todos los archivos los cuales fueron actualizados son cerrados al salir del procedimiento de cualquier forma ya sea con -EXIT -RUN o que sea la última línea. Si un archivo va a ser expandido entonces se debe usar el atributo de

DISP=MOD en el estatuto FILEDEF

Comando READ

El formato para leer datos desde archivos externos es:

```
READ ddname text
```

donde ddname es el nombre del archivo a ser leído y text es el conjunto de variables ampersand y un texto.

Dos tipos de información puede ser leída:

Formato Fijo: En los cuales todas las variables estan en columnas fijas y en un formato libre en donde las variables son delimitadas por comas.

El contenido de las líneas de un -READ es un conjunto de variables ampersand que se conocerá su valor mas delante. Esto es ejecutando mejor usando el estatuto -DEFAULT. De esta manera, el número de caracteres esperado para cada variable es conocido.

El formato de lectura libre es indicado separando todas las variables a leer con comas.

```
-READ ddname &A,&B,&C,&D
```

Si la lista de variables no cabe en una línea al final de esta debe haber una coma (,) y la línea siguiente debe empezar con el signo (-).

```
-READ ddname, &A,&B,&C,
-&D,&E
```

Las variables que son leídas usando una lista de variables separadas por comas alojadas en variables ampersand basadas en la separación de estas por las comas.

## Otras Facilidades del Dialoguer Manager

### Ejecución Automática de un Procedimiento (PROFILE FOCEXEC)

Si un procedimiento llamado PROFILE esta presente, este sera ejecutado automáticamente cada vez que FOCUS este activado. Este es usual para especificar parámetros ambientales como son: FOCUS, USE, SET. La definición de variables complejas puede ser automáticamente ejecutada y así poder disponer de ellas.

EJEMPLO: Profile FOCEXEC

```
USE
C:PERS.FOC
END
SET PAUSE=ON, MSG=OFF
FILEDEF MYSAV DISK C:SAVE.FTM
DEFINE FILE CARS
SPREAD/I6=IF RCOST GT 0 THEN (RCOST-DCOST)/DCOST
                        ELSE 1.30*DCOST;

END
-TYPE FOCUS SESSION ON &DATE AT &TOD
```

MS-DOS

El procedimiento puede ser llamado PROFILE.FEX y puede estar en el disco default una vez que ha sido activado de PC/FOCUS.

### Suprimiendo la Ejecución

Todas las líneas que fueron almacenadas normalmente son pasadas a FOCUS por un procedimiento pueden ser eliminados. Esto permite probar la lógica de los estatutos de control FOCEXEC.

Para desactivarla:

```
-SET &STACK=OFF
```

Ocultando Procedimiento FOCEXEC

Es a veces conveniente proteger un procedimiento almacenado en un archivo FOCEXEC un permitiendo asi, ser visto por el usuario. Pueden existir datos confidenciales dentro de un procedimiento o un password que no se quiere que este pueda ser alterado por el usurio. Esto se puede usuar ejecutando un procedimiento FOCEXEC protegido o oculto. Solo la persona que lo protegio puede desprotegerlo. El password.

Ejemplo.-

```
SET PASS=DOHIDE
ENCRYPT FILE DAILY.FEX.
```

Cualquiera puede usar:

```
EX DAILY
```

Para desprotegerlo:

```
SET pass=dohide
DESCRYPT FILE DAILY.FEX
```

Se le da adicionalmente la palabra FEX a los comandos ENCRYPT y DECRYPT usa dos en archivos MASTER.

Cuando un procedimiento FOCEXEC esta protegido su contenido no puede ser visto, el modo ECHO=ON no tiene efecto.

Los procedimientos que se quieren proteger o desproteger no pueden tener formato concatenado.

Borrando la pantalla para la selección de un menu.

CRTCLEAR command-borrando la pantalla

Los usuarios de FOCUS operando FOCUS desde una IBM3270 trabajando desde una terminal o de una IBM PC puede borrar la pantala usando la etiqueta -CRTCLEAR.

Entrada de los datos en la forma de Pantalla llena.

El D.M. soporta la forma de entradaa de datos con pantalla llena, con la cual el operador mueve el cursor de una posición a otra y llenando los valores necesarios. El lenguaje usado para esta forma es similar a la del FIDEL.

(FOCUS interactive Data Entry Languaje)

Para activar esta forma se usa la linea de control:

```
-CRTFORM
```

y cada línea de esta forma es dada entre comillas y con el signo (-) al principio.

Por ejemplo:

```
-CRTFORM
-"       DATE:<D.&DATE       TIME:<D.&TOD  "
-"</4  "
-"  PLEASE GIVE THE FIELDNAMES BELOW  "
-" "
-" <&FIELD  "
-"  PLEASE GIVE THE SORTING OPTIONG BELOW  "
-"  <&SPORTT  "
-"              PRESS ENTER TO EXECUTE, PF1 TO QUIT
```

Las variables pueden ser predeclaradas y asi FIDEL es lamado para preparar area para estas. (Se usa el comando -SET para hacer esto)

La forma de salirse de este modo con una linea que no empieza (-). En este momento, FIDEL es llamado para desplegar la lera. pantalla.

Cuando el ENTER es presionado los nuevos valores de las variables son actualizadas y la pantalla desaparece.

Si se presiona la tecla F3 es equivalente a un "END".

Las variables a ser llenadas son identificadas por el signo < que aparece enfrente de ellas. Ej. <&MPG. Cuando a la variable se le prefija una T esto significa que el valor actual de la variable será desplegado y se acepta un nuevo valor. Por ejemlo: <T.&WHERE. Cuando se le antepone una "D" significa que el valor actual de la variable es desplegado no permitendo hecerle ningun cambio.

FIDEL

Lenguaje Interactivo para entrada de datos FOCUS

Descripcion de la facilidad

El lenguaje interactivo para entrada de datos FOCUS (Fidel) es diseñado por los usuarios de terminales CRT los cuales tienen conocimiento del control del cursor y que puede operar en un (modo block) de tansmición de datos. En estas terminales los ususarios pueden especificar el texto que aparecerá en la pantalla y en donde la nueva informción sera llenada. El operador de la terminal puede entonces proceder a dar entrada de los datos usando el método visual "llevando de datas necesarios". El movimiento dentro de la pantalla es ejecutado presionando la tecla, tab la cual posicionará el cursor al inicio de cada nueva locallización en donde se dará entrada a los datos en la pantalla. Las teclas para moverse hacia arriva, abajo a la derecha y a la izquierda pueden ser usadas tambien por eloperador para controlar el posicionamento del cursor. El operador puede moverse desde un desplegado al otro presionando las teclas F7 y F8.

Después de que la máscara principal es presentada esta continua en la pantalla hasta que el aperador señala el final de la entrada de datos. Solo los datos tecleados por el operador son transmitidos a la computadora. Despues de cada transacción satisfactoria, las posiciones de entrada de datos en la pantalla son borradas dejando solo la máscara.

Si un error es encontrado en la transaccion un mensaje de error aparecerá al final de la pnatalla. Una campana sonara (beep) y la pantalla no será borrada, esto da al operador la oportunidad de corregir el error y retrasmitirlo.

La pantalla puede contener 3 tips de areas de variables. El primer ripo da la posicion para nuevos valores de datos; el segundo tipo despliega los valores actuales de los campos de la base de datos, el tercer tipo, despliega los datos actuales y acepta el reemplazamiento de valores. La facilidad del desplegar los valores actuales de los campos de una base de datos los cuales pueden ser recuperados en una manera aleatoria, permite usar al FIDEL para propósitos de investigación así como para actualizar la base de datos.

Los datos de la pantalla son transmitidos después de haber sido llenada por el operador presionando la tecla ENTER.

El final de la entrada de datos es enseñada por el operador tecleando 'END' en la primer area de trabsnucuión de datos. En la IBM PC presionar la tecla F3 es equivalente al 'END'.

Al final de esta sección se dará información de esto en la terminal CRT. Esta lista esta siendo continuamente expandida y el representante de FOCUS debe estar en contacto con la información dada en la terminal y no en la lista.

Describiendo la pantalla CRT

El subcomando MOFIY del CRTFORM, seguido por el layout(?) de la pantalla, es usado para atener una forma mas visible de la pntalla. Para posicionar los textos e identificar los campos de datos se usa la misma sintaxis que la usada al escribir encabezados en la tabla de reportes (TABLE). Cada línea que aparecera en la pantalla es dada entre comillas. Por ejemplo:

"Esta es una línea texto"

la posicion en donde los datos seran puestos es dada por el símbolo "<" seguido del nombre del campo y seguido opcionalmente por el símbolo ">". Por ejemplo:

"CAR NAME <CAR> MODEL NAME <MODEL>"

En la pnatalla los blancos son provistos dependiendo de la longuitud el campo. Por ejemplo, si el campo CAR tiene los caracteres, el campo es justificado a la izquierda por los espacios.

Cada línea que aparecerá en pantalla es dfada una tras otra. Las otras opciones de control del subcomando MODIFY pueden usarse antes o después de estas líneas como cuando es usado el subcomando FIXFORM.

Como un ejemplo de estas técnicas basicas, supongase que se deseaactualizar varios campos del archivo CARS y asegurarse de que el número de asientos y el MPG cumplan con ciertos límites.

El siguiente programa producirá la pantalla deseada.

Ejemplo:Uso de CRTFORM

```
MODIFY FILE CARS
CRTFORM
"   IMPORTED CAR UP DATE    "
"   COUNTRY OF ORIGIN:<COUNTRY>CAR  NAME:<CAR>    "
"   MODEL NAME:        <MODEL>  BODY TYPE:<TUPE>  "
"   SEATS: <SEATS>   MILEAGE: <MPG>    "
RETAIL COST: <RCOST> DEALER  COST:<DCOST>  "
```

```
VALIDATE
SEATEST=SEATS GT 0 AND SEATS LT 9;
MPGTEST=MPG LT60;
MATCH COUNTRY
    ON NOMATCH REJECT
    ON MATCH CONTINUE
MATCH CAR
    ON NOMATCH REJECT
    ON MATCH CONTINUE
MATCH MODEL
    ON NOMATCH REJECT
    ON MATCH CONTINUE
MATCH TYPE
    ON NOMATCH REJECT
    ON MATCH UPDATE RCOST DCOST MPG SEATS
DATA
```

La pantalla aparecera como:

```
IMPORTED CAR UPDATE
COUNTRY OF ORIGIN:          CAR NAME:
MODEL NAME:                 BODY TYPE:
SEATS:        MILEAGE:
RETAIL COST:     DEALER COST:
```

Note que los textos no estan alineados uniformemente debido las diferentes longuitudes de los campos. La opción de marcado de texto (sport marker) es descrita haciendo esto fácil de realizar.

Recursos para el posicionamiento de textos-Sport Markers:

Los Sport Markers estan disponibles para ayudar en el posicionamento, tanto de textos mo de campos de datos. Estos siempre empiezan con "<" seguido de un número o de un caracter especial.

Los efectos de los Sport Markers son los siguientes:

| MARCADOR | EJEMPLO | USO |
|---|---|---|
| <n o <n> | <50 | El siguiente caracter empieza en la columna n. |
| <+n o <n> | <+4 | El siguiente caracter empieza n columna a partir del ultimo caracter diferente de blanco escrito. |
| </n o </n> | </2 | Brinca n lineas. |

Sustitución de Campos de datos

En un campo de datos, para el cual se le tendrá que dar un valor tiene el nombre encerrado entre caretas izquierdad (<) o derechas(>). Ejemplo:

```
<CAR> <MPG
```

En ambos casos un valor para el campo será dado. Si el valor no se da se asume un balnco o un cero.

Al tener (<) significa que el valor no es obligatorio darlo. Si no se le da valor al campo, este es ignorado.

Estas reglas sin análogos a las de la descripción de entrada de datos con el subcomando FIXFORM. Por ejemplo:

```
 "<CAR>" es equivalente en FIXFORM a CAR/10
y"<CAR>"  "      "       "   "     " CAR/C10
```

Grupos de campos:

Un grupo de campos se puede repetir de la forma. Poe ejemplo:

```
"MODELNAME   BODYTYPE   MILEAGE"
"<MODEL      <TYPE         <MPG"
"<MODEL      <TYPE         <MPG"
"<MODEL      <TYPE         <MPG"
```

Esto es equivalente a:

```
FIXFORM  3(MODEL/C16  TYOPE/C8  MPG/C5)
```

Si hay varios grupos en la pantalla FIDEL los procesará, uno por uno como si estos tuvieran patrón.

*n(group 1)  n(group 2)*

Si los grupos están anidados dentro de otros se requiere de varios estatutos CRTFORM. Ejemplo:

```
CRTFORM
"YEAR <YR MONTHS <MO <MO <MO <MO"

CRTFORM
"YEAR <YR MONTHS <MO <MO <MO <MO"
```

Si el 2do. estatuto CRTFORM, el conjunto de datos sería mal interpretado como 2años, 8meses. De otra forma la interpretación es año 4(mo) año 4(Mo). Actualmente cada CRTFORM con todas las lineas que este contenga son ejecutadas como si fuera una unidad y subsecuentes CRTFORM no son procesadas hasta terminar el procesamiento del CRTFORM anterior.

Cuando hay mas patrones de repetición de datos más complejos, el uso del CASE es recomendado.

El CASE ciclo, repite y llama a otros CASE dependiendo de los datos a capturar.

Limpiando la Pantalla

El proceso por default despues, de cada pantalla es limpiar el area de datos de esta. Este procedimiento puede ser omitido y dejar a la pantalla inicial de cada transnsmición.

Esta es una caracteristica muy util cuando se tiene una gran cantidad de datos los cules son tranferidos de una pantala a otra.

Cada campo en la pnatalla que no ha sido cambiado se retendrá y será reetransmitido como parte de la siguiente transacción.

El comando CRTFORM para cumplir con esta funcion es:

```
                ¶ blanco   ¶
CRTFORM         ¶ CLEAR    ¶
                ¶ NOCLEAR  ¶
                  --     --
```

Cuando el NOCLEAR es usado la pantalla no es limpiada, y cuando se ua CLEAR la pantalla esta simpre llimpia aun después de un error.

Usando minusculas

Todo texto es dado en la terminal es normalmente transoformado en mayusculas. Esta es la opción mas común pero ambos casos ya sea preservados si se le da la palabra LOWER después del subcomando CRTFORM.

El formato para la entrada de datos en minusculas es:

```
                  --     --
                ¶ blanco   ¶
CRTFORM         ¶ CLEAR    ¶
                ¶ UPPER    ¶
                  --     --
```

Pantallas Continuas

Hasta esta momento hemos estado asumiendo que una 'forma' y una 'pantalla' son sinonimos, actulamente son dosn entidades diferentes. La 'forma' es una unidad lógica que contiene todos los textos y las areas de datos requeridos para una aplicacion. Un simple CRTFORM puede contener hasta 1280 linead. La pantalla es un dispositivo disico cuyo tamaño es generalmente de 24 lineas. FIDEL usa las últimas cuatro lineas para los mensajes de error. Asi que varias pantallas son necesarios para mostrar la forma entera al operador. Esto se efectua de la siguiente manera.

Al operador se le muestran las primeras 20 lineas. Oprimiendo la tecla F8 las segundas 20 lineas aparecen automáticamente y asi sucesivamente. En cualquier momento el operador antes de teclear ENTER y de que los datos sean procesados el operador puede retroceder o avanzar na pantalla. Para retroceder una pantalla la tecla a uasr es F7. Para pasar a otra pantalla se usa la tecla F8, todos los datos previamente dados pueden ser llamados.

Algunos puntos deban ser aclarados concernientes a las pantallas continuas.

a. Una segunda pantalla es forzada llenando la pantalla anterior con lineas en blanco, un simple forma de hacer esto es usando </n.

b. Cuando la opcion NOCLEAR es usada esta es aplicada a todas las pantallas con un solo comando CRTFORM.

Una de las principales características del uso de esta es colocar las instrucciones para ejecutar el procedimiento en la segunda patalla. El operador puede usar la tecla F8 para que aparezcan cuando esto le sea necesario.

Ejemplo: Pantallas continuas

```
MODIFY FILE CARS
CRTFORM
" FIRST PART OF SCREEN "
" COUNTRY <20 <COUNTRY "
" CAR     <20 <CAR "
" MODEL   <20 <MODEL "
" BODY    <20 <BODY "
" </15              <----Note
" SECOND PART OF SCREEN "
" RETAIL COST <20 <RCOST "
" .
" .
  etc.
```

Usando Pantallas a lo largo

FIDEL tiene una pantalla default de 24 lneas con 80caracteres por línea. Esta convención default deja un máximo de 20 líneas para ser usadas en la máscara de la pantalla. Cuatro líneas son dejadas libres para los mensajes de error. En cada línea 2 caracteres son reservados y un máximo de 78 caracteres pueden ser usados por textos y datos. Si una pantalla de CRT excede estos límites un mensaje de error es desplegado y el procedimiento es rechazado. Estos limites dependiendo del tipo de terminal y si son doferentes estos pueden ser especificados a FIDEL usando el siguiente formato.

```
                 ----    ----        -----     -----
CRTFORM     ¶ WIDTH nnn  ¶       ¶  HEIGHT nnn    ¶
            ¶    80      ¶       ¶       24       ¶
                 ----    ----        -----     -----
```

donde WIDTH es el número de caractéres por línea y HEIGHT es el número total de líneas en la pantalla.

Localización de los mensajes de Error

Las últimas 4 líneas de la pantalla son utilizadas para desplegar mensajes de error [un BEEP se oirá si la terminal soporta esta característica] y la pantalla no puede no ser borrada. Esto presenta una oportunidad para poder corregir los datos que causaron el error y estos son retransmitidos a la pantalla. Unicamente los campos de error deben ser cambiados. Si la opción CLEAR es usada después de CRTFORM la pantalla será borradda aun después de un mensaje de error.

Opciones de Procesamiento

El procedimiento MODIFY es procesado en el orden en que los subcomandos son presentados. Generalmente la secuencia es, recopilar datos,validarlos, revisando que los datos cumplan con los datos dados en el archivo; ejecutar validaciones adicionales; y después actualizar los campos existentes a dar de alta nuevos registros o la remoción de registros.

FIDEL presenta datos adicionales que pueden ser utilizados por el operador durante el proceso de validación. Por ejemplo los subcomandos

ON MATCH CRTFORM

ó ON NOMATCH CRTFORM

ó MATCH/NOMATCH CRTFORM

Pueden ser usados. Dado que la posición de los campos de los archivos ha sido establecida, los valores de los campos de los registros existentes estan disponibles para ser usados como referencia o para ser modeificados.

1.- DISPLAY (solo para información)
La identificación del valor del dato es:

D.Fieldname

El prefijo 'D.' es puesto antes del nombre del campo que ha sido recuperado. El valor actual del campo aparece en la pantalla, pero es protegido de culaquier tipo de cambio.

2.- Dentro del Campo (para desplegar y cambiar)
El valor de identificación del dato es:

T.Fieldname

El prefijo 'T' es puesto antes del nombre del campo que ha sido recuperado. El valor actual del campo aparece en la pantalla, si no se la da valor a esta instrucción es ignorada, pero el operador puede cambiar el valor del campo. En este caso el nuevo valor es dado y una nueva condición puede ser aplicada a este.

Si el símbolo (>) no es usado el T.FIELDNAME es tratado como condicional y al usar VALIDATE o COMPUTE estiman este valor como cero o blanco si el valor del campo no ha sido cambiado. Si el símbolo (>) es usado en el T.FIELDNAME es tomado como presente aun sin haber sido dado.

Para desplegar los campor 'D.' o T.FIELDNAME una validación debe haber ocurrido en los segmentos que contienen los campos desplegados, en el caso que sean desplegados. Para desplegar los campos de un segmento único el comando ON MATCH CONTINUE TO.... debe ser ejecutado.

El siguiente ejemplo ilustra el desplegado y la actualización de datos. El primer CRTFORM pide el campo, en este caso el número del seguro social de un empleado. Si un registro correcto es obtenido, entonces varios valores son desplegados y otros son actualizados.

```
MODIFY FILE PERSON
CRTFORM
"  ENTER SSN: <SSN>  :
MATCH SSN
   ON MATCH TYPE
   " ON DATA FOUND FOR <D.SSN> PLEASE RE-ENTER"
   ON NOMATCH REJECT
   ON MATCH CRTFORM
   " FULL NAME: <D.FIRST  <D.MIDDLE  <D.LASTIN  "
   " ADDRESS: <T.ADDRESS  CITY-STATE:<T.CITY <T.STATE  "
   " DEPARTAMENT: <T.DEPT  "
   ON MATCH UPDATE ADDRESS CITY STATE DEPT
   LOG NOMATCH MSG OFF
DATA
```

Después de que el operador halla llenado el primer CRTFORM, la tecla ENTER debe ser presionada y este transmitído, si este es un registro válido los valores de los campos en el segundo CRTFORM y el símbolo '' es posicionado al inicio de la primera línea de la segunda forma. El operador entonces puede llenar la segunda forma y transmitirla.

Cualquier número de procedimientos ON MATCH CRTFORM o NOMATCH CRTFORM puede ser utilizado, todas la formas pueden ser mandadas a la pantalla y guardadas en su lugar una después de otra.

Esta pantalla aparace como:

```
ENTER SSN:
FULLNAME:
ADDRESS:          CITY-STATE:
DEPARTAMENT:
```

Note que ambas formas se muestran en la pantalla.

OPCION MATCH/NOMATCH

El subcomando MATCH/NOMATCH CRTFORM puede ser usado cuando una condición de validación existe pero este puede ser usado aú no existiendo condición. Por ejemplo, si un valor ya existe se puede usar NOMATCH y si existe para actualizarlo MATCH.

MULTIPLES CRTFORMS

Usando las facilidades de la opcin del MODIFY y el comando CASE LOGIC como padre de muchos CRTFORMS pueden ser desplegados en un procedimiento. Cada CASE describe sus propias pantallas las cuales tienen que ser completamente reemplazadas desde el primer CASE o puede empezar en una línea especifica, esto deja varias líneas en la pantalla .

Cuando existen varios CRTFORMS el formato es:

```
CRTFORMS [LINE nn]
```

Generalmente la línea 1 es usada para reemplazar una pantalla por otra.

La idea básica del CASE LOGICO es separar completamente los procesos del MODIFY pudiendo ser ejecutados en un solo procedimiento. Cada uno de estos es un CASE y el procedimiento contiene direcciones a ejecutar, el CASE adecuado de acuerdo a ciertas circunstancias.

Combinación de pantallas "Opción Line"

Una pantalla combinada es una combinación de dos o mas CRTFORM individuales, cada uno de los cuales ocupa líneas específicas y que pueden ser reemplazadas independientemente una de otra. Una pantalla combinada es obtenida poniendo el número de línea inicial en el CRTFORM. La sintáxis es:

```
CRTFORM  [line nn]
```

Por Ejemplo:

```
CRTFORM LINE 6
```

Una pantalla es reemplazada completamente cuando ambas pantallas inician en la misma línea.

Las pantallas combinadas son múltiples CRTFORMS pero que usan la opción LINE para reservar varias líneas en la pantalla. Las pantallas combinadas pueden ser usadas en el caso de CASE LOGICO donde el CASE especifica una pantalla con diferentes MATCH o NOMATCH.

Una pantalla puede escribir parte de otra. Esto no es buena idea, pero puede ser usual en algunas aplicaciones.

Cuando el número de línea no es usado, cada CRTFORMS es puesto secuencialmente debajo del último procedimiento al leer el procedimiento MODIFY.

BIBLIOGRAFIA
==========


I.- PC/FOCUS [INFORMATION BUILDERS,INC]
USER MANUAL
RELEASE 1.5

II.- PC/FOCUS [INFORMATION BUILDERS, INC]
GUIDE TO OPERATIONS
GETTING STARTED
TABLE TALK

PC/FOCUS
INFORMATION BUILDERS, INC
1250 BROADWAY, NEW YORK,
NY 10001 (212) 736-4433

ENTERING PC/FOCUS

III.- MICROCOMPUTER DICTIONARY
BY CHARLES J. SIPPL
SECOND EDITION
RADIO SHACK
T/C TANDY CORPORATION COMPANY