Qualitas
Sistemas de Información
Una división de DMR Consulting

Manual JCL

Manual JCL

Id.Doc: Haga click y escriba

Fecha: Haga click y escriba

Índice

1 Introducción

1.1 Job Control Language

El JCL es el lenguaje especial que se utiliza para indicar al Sistema Operativo los trabajos que debe realizar, señalando:

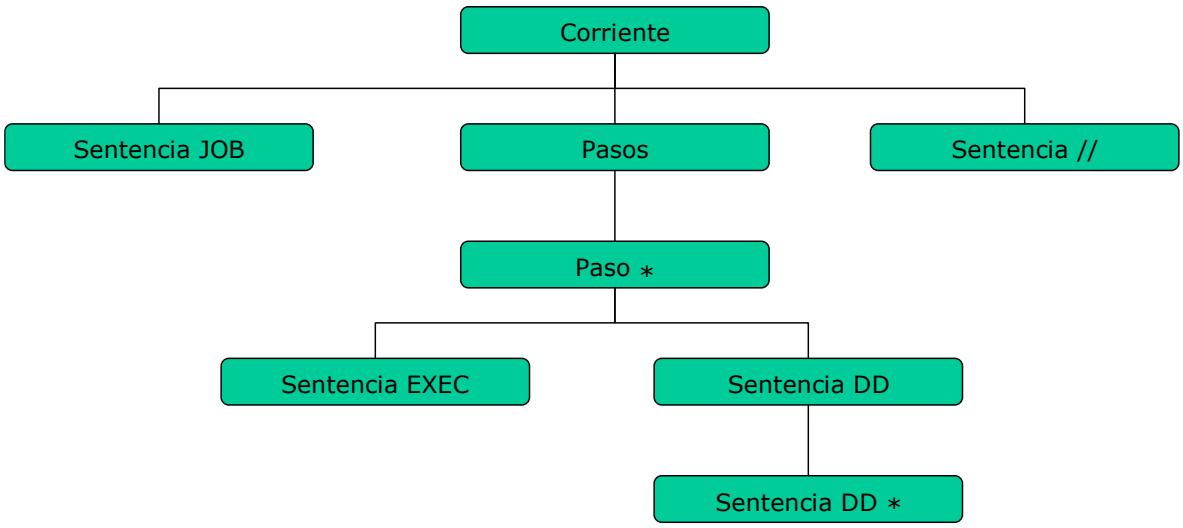
- Los programas que se deben ejecutar.
- Los recursos que necesitan los programas en ejecución.

Un conjunto de programas que realizan un determinado trabajo constituyen lo que se denomina un JOB.

Corriente de Control del JOB es el conjunto de sentencias de JCL necesarias para ejecutar un JOB determinado.

Además de las sentencias de control para el sistema se pueden incluir sentencias de control del subsistema de entrada de trabajos (JES) para obtener procesos especiales de entrada y salida.

1.2 Estructura corriente de control



1.3 Sentencias de control de trabajos del OS/VS

1.3.1 JOB

Identifica un trabajo para el sistema. Indica el principio de un JOB y el final del precedente.

1.3.2 EXEC

Identifica un paso de trabajo a ejecutar. Indica qué procedimiento se debe expandir o qué programa se llama a ejecución

1.3.3 DD (Data Definition)

Permite referenciar los ficheros con los que se va a trabajar durante la ejecución del programa, indicándole al sistema si debe crearlos en ese momento, cuáles son sus características, etc.

1.3.4 PROC

Delimita el principio de un procedimiento "in-stream" y, opcionalmente, puede estar al principio de un procedimiento catalogado.

1.3.5 PEND

Indica el final de un procedimiento "in-stream".

1.4 Sentencia de comandos

Se utiliza para introducir un comando al sistema por medio de un dispositivo de entrada de trabajos (//).

1.4.1 Sentencia nula

Indica el final de un job (//).

1.4.2 Sentencia delimitadora

Es el indicador 'standard' de fin de un fichero en fichas (/*).

1.4.3 Sentencia de comentarios

Se utiliza para insertar comentarios entre las fichas de control (//*).

1.5 Sintaxis de las sentencias de control

Se dispone de las columnas 1-71.

//NOMBREXX OPERACION OPERANDOS

- **//**Columnas 1 y 2.
- **NOMBREXX:** Identifica la sentencia. De 1 a 8 caracteres alfanuméricos (A-Z, 0-9, $, @, sostenido), sin caracteres especiales. El primer carácter debe ser no numérico. A partir de la columna 3.
- **OPERACIÓN:** Tipo de sentencia de control o de comando. Debe estar separado del campo NOMBRE al menos por un blanco.
- **OPERANDOS:** Contiene parámetros relativos a la sentencia de control, separados por comas. Los parámetros pueden ser, por este orden:
  - *Posicionales*. El significado de cada parámetro viene dado por su posición dentro del conjunto de parámetros. Su falta debe indicarse con una coma en el sitio donde debiera estar.

    Ej: //JOB1 JOB 41,LOPEZ
  - *De Palabra Clave*. El significado viene dado por una palabra especial que lo identifica.

    Ej: //JOB2 JOB CLASS=A

  A su vez un parámetro puede estar compuesto por una serie de subparámetros (posicionales o de palabra clave). Deben ir encerrados entre paréntesis, o, a veces, entre apóstrofos.

  Ej: //JOB3 JOB (41,52,007,,LOPEZ),CLASS=A

  Toda sentencia JCL puede llevar comentarios después del último parámetro y precedidos al menos por un blanco.

Continuación de las sentencias de control:

- Se interrumpe la codificación de la sentencia antes de la columna 72.
- Si hay comentarios, un carácter no blanco en la columna 72.
- En la ficha de continuación, // en las columnas 1 y 2.
- La ficha de control se continúa entre las columnas 4 y 16. Si es comentario, no se tiene en cuenta el límite de la 16.

2 SENTENCIA JOB

Identifica el principio de un trabajo para el sistema. Debe haber una sola sentencia JOB por cada job que se introduzca en el sistema.

2.1 Parámetros posicionales

2.1.1 Información de contabilidad

Para la contabilización del JOB por el sistema. Entre paréntesis o apóstrofos si hay más de un subparámetro. (Como máximo 142 caracteres incluyendo las comas de separación).

Ej: //JOBA JOB (0471,25-12-74,LUIS,'457''53.71')

2.1.2 Nombre del programador

Si se codifica debe ir tras la información de contabilidad. En el caso de que esta se omita, debe ir precedido de una coma. De longitud máxima 20, si contiene caracteres especiales debe ir entre apóstrofos. Si alguno de sus caracteres es un apóstrofo debe ir duplicado.

Ej: //JOBB JOB 074,MARTINEZ
//JOBC JOB ,MARTINEZ
//JOBD JOB ,'f:J''C'

2.2 Parámetros de palabra clave

En cualquier orden. Si existen parámetros posicionales deben ir tras ellos.

2.2.1 ADDRSPC

Especifica si ese JOB se ejecuta en memoria virtual (si hay paginación, por defecto) o real.

ADDRSPC=VIRT / REAL

2.2.2 CLASS

Asigna una cola de entrada donde lo va a colocar el JES en espera de que se ejecute.

CLASS=clase de entrada (A-Z, 0-9)

2.2.3 COND

Especifica una condición que imponemos de que los pasos de un JOB se sigan ejecutando, basada en los códigos de retorno que van devolviendo los distintos pasos tras su ejecución.

Parámetro COND	Job continua si	Job termina si
COND=(code,GT)	RC >= code	RC < code
COND=(code,GE)	RC > code	RC <= code
COND=(code,EQ)	RC ¬= code	RC = code
COND=(code,LT)	RC <= code	RC > code
COND=(code,LE)	RC < code	RC >= code
COND=(code,NE)	RC = code	RC ¬= code

2.2.4 MSGCLASS

Especifica la clase de salida donde queremos que vayan todos los mensajes del sistema y las sentencias de control.

MSGCLASS=clase de salida (A-Z, 0-9)

2.2.5 MSGLEVEL

Indica qué tipo de mensajes y de sentencias de control relacionados con nuestro job queremos que nos imprima el sistema. Por defecto el valor especificado en la instalación.

MSGLEVEL=([sentencias],[mensajes])

- **Sentencias**
  - 0 Sólo se imprime la sentencia JOB.
  - 1 Se imprimen todas las sentencias de control de entrada y las que resulten de expandir el procedimiento catalogado.
  - 2 Sólo se imprimen las sentencias de control de entrada.
- **Mensajes**
  - 0 No se imprimen los mensajes de asignación/terminación, salvo que el job termine anormalmente.
  - 1 Se imprimen todos los mensajes de asignación/terminación.

Si el subparámetro sentencias no se codifica, se debe colocar una coma en su lugar. Si es mensajes lo que no se codifica, no se necesita poner paréntesis.

2.2.6 NOTIFY

Le pide al sistema que envíe un aviso al usuario de TSO cuando termine la ejecución del trabajo introducido.

NOTIFY=identificador de usuario

2.2.7 PERFORM

Especifica a qué grupo de rendimiento queda asignado el JOB. Por defecto:

- **1** Para jobs que no son de TSO
- **2** Para jobs que son de TSO

PERFORM=n (1-125)

2.2.8 PRTY

Especifica la prioridad que se le asigna a ese JOB.

PRTY=P (valor de 1-15. La más alta es 15)

2.2.9 RD

Controla las posibilidades de rearranque del JOB.

RD= |R |
|RNC|
|NC |
|NR |

- **R** Se permite el rearranque automático de paso. Dos posibilidades:
  - a nivel de paso.
  - a nivel de checkpoint.
- **RNC** Se permite el rearranque automático de paso, no de checkpoint.
- **NC** No se permite ninguno de los dos rearranques automáticos.
- **NR** Se pueden tomar checkpoints, aunque no se permita ningún tipo de rearranque automático luego si se puede rearrancar a nivel del checkpoint tomado.

2.2.10 REGION

Con ADDRSPC=REAL indica el tamaño de memoria real que utiliza dicho JOB.

Con ADDRSPC=VIRT sirve sólo para limitar la cantidad de memoria virtual que se puede tomar con una GETMAIN variable.

REGION=valor K (debe ser un número par. Si se codifica impar,
el sistema lo pasa al par superior).

2.2.11 RESTART

Se indica desde qué punto se quiere que se empiece la ejecución en el caso de un rearranque diferido.

```
                    |*              |
RESTART=(|paso       |[,checkid])
                    |paso.paso dentro procedimiento|
```

- ***** Indica el primer paso.
- **Checkid** Opcional, queremos que el job arranque su ejecución en el checkpoint indicado por este subparámetro, dentro del paso especificado. Si no se codifica se arranca al principio del paso.

2.2.12 TIME

Tiempo máximo que el JOB puede utilizar la CPU. Por defecto 30 minutos.

TIME=(|minutos|,|segundos|)

- **Minutos** (1-1439) (1440 tiempo ilimitado)
- **Segundos** (0-59)

2.2.13 TYPRUN

Indica cómo debe considerar el sistema a nuestro JOB con vistas a ejecución.

TYRUN=|HOLD |
|SCAN |
|JCLHOLD|
|COPY |

- **HOLD** El job se coloca en cola de entrada pero no se puede llamar a ejecución hasta que lo libere el operador.
- **SCAN** Se comprueban los posibles errores que haya en las fichas de control pero no se ejecuta.
- **JCLHOLD** El job ha de ser retenido antes de ser procesado por el 'converter' del JCL; no pasa a la cola de entrada hasta que el operador lo libere.
- **COPY** Indica que las fichas de entrada se conviertan en un fichero SYSOUT y se ponga en cola para el proceso de salida.

3 SENTENCIA EXEC

Identifica un paso de trabajo dentro del JOB completo. Indica qué programa se debe ejecutar y los parámetros de un paso.

3.1 Parámetros

3.1.1 PGM

Indica el programa que se va a ejecutar y debe ser el primer parámetro de la sentencia EXEC.

```
        |programa
PGM=    |*.nombre de paso.nombre DD
        |*.nombre de paso.nombre paso dentro proc.nombre de DD
```

El primero es el caso de que conozcamos el nombre del miembro donde está el programa, o el nombre del programa.

Las otras posibilidades se dan cuando no conocemos ese nombre. Habrá que hacer referencia a la ficha DD que define esa librería para que el sistema pueda buscarlo por el nombre temporal que le dio cuando lo creó.

```
Ej: //LIBRDD    DD   DSN=&&LIBR(PROG)
    //PASON     EXEC PGM=*.LKED.LIBRDD
```

3.1.2 PROC

Indica que la sentencia EXEC está haciendo referencia a un procedimiento, sea 'in-stream' o catalogado, para que se expanda o ejecute todo lo indicado en él.

PROC=nombre de procedimiento,

o, simplemente:

nombre de procedimiento

3.1.3 ACCT

Especifica la información contable relativa al paso y que exige la instalación. Palabra clave, con restricciones parecidas al parámetro posicional de la sentencia JOB.

ACCT[.nombre paso dentro procedimiento]=inf.contable

```
Ej:  //PASO1  EXEC   PGM=A,ACCT=041
     //PASO2  EXEC   PGM=B,ACCT=(041,'12/12/74')
     //PASO3  EXEC   PGM=C,ACCT=(041,12/12/74)
     //PASO4  EXEC   ABC,ACCT.PASO41=(041,ABCD)
     //PASO5  EXEC   PROCED1,ACCT.PASON='135+47'
```

3.1.4 ADDRSPC

Especifica si el paso se ha de ejecutar en memoria real o virtual (por defecto).

ADDRSPC[.nombre paso dentro procedimiento]=|VIRT|
|REAL|

Si ponemos este parámetro en la ficha JOB y en las EXEC, aunque pongan cosas contradictorias, el valor que prevalece es el dado en la ficha JOB. Para poder ejecutar unos pasos en memoria real y otros en virtual, se ha de codificar en las EXEC.

3.1.5 COND

Especifica bajo qué condiciones no se ejecutara el paso, dependiendo de los códigos de retorno devueltos por uno, varios o todos los pasos anteriores.

```
                                    |código,operación|
COND[.nombre paso dentro de proc]=(  |código,operación,paso|…|,EVEN|  )
                                    |código,operación,paso|…| ONLY|
                                    | .nombre paso dentro proc |
```

- **código** Valor con el que comparar el código de retorno (0-4095).

- **operación** GT mayor que
  GE mayor o igual
  LT menor que
  LE menor o igual
  EQ igual a
  NE distinto de
- **EVEN** este paso se debe ejecutar aunque algún paso anterior terminase anormalmente.
- **ONLY** este paso se ejecutará si algún paso anterior ha terminado anormalmente.

Se pueden especificar hasta 8 condiciones distintas incluyendo la EVEN u ONLY. Si en la JOB se codificó COND, se ignoran los de la EXEC.

Ej: //PASO1 EXEC PGM=PROG1
//PASO2 EXEC PGM=PROG2,COND=(4,EQ,PASO1)
//PASO3 EXEC PGM=PROG3,COND=((8,LT,PASO1),(8,GT,PASO2))

3.1.6 DPRTY

Prioridad de este paso.

DPRTY=(|valor1|,|valor2|)

- **valor1** Número entre 0 y 14 con el que el sistema determina si este paso debe tener una prioridad de ejecución igual o distinta a la que se indicó en la sentencia /* PRIORITY del JES2, o a la que calculó por defecto el sistema a la vista de los valores dados para tiempo de ejecución..., o a la especificada por defecto por la instalación.
- **valor2** Número entre 0 y 15 que se añade al valor interno generado por el sistema a partir del valor1 para formar la prioridad real de ejecución.

Prioridad final: (16 x valor1 + valor2) = prioridad ejecución

Si no se codifica este parámetro, el paso queda asignado al grupo de prioridades automáticas (APG).

En el caso de colocar este parámetro en una sentencia EXEC que llama a un procedimiento catalogado hay dos posibilidades:

- Prioridad igual para todos los pasos de ese procedimiento.

  DPRTY=

- Se desea asignar o variar la prioridad de ejecución a uno o varios pasos del procedimiento.

  DPRTY.nombre paso dentro procedimiento=

Ej: //PASO1 EXEC PGM=A,DPRTY=(13,9)

//PASO2 EXEC ABC,DPRTY=12

//PASO3 EXEC ABC,DPRTY.PASOA =(5,4),DPRTY.PASOB =(,10)

3.1.7 DYNAMNBR

Sirve para que el iniciador asigne ficheros dinámicamente. También los propios programas de proceso del usuario pueden pedir asignación de dispositivos para sus ficheros dinámicamente, es decir, en tiempo de ejecución.

Al iniciador hay que indicarle cuantas asignaciones dinámicas se van a efectuar como máximo para que prevea espacio para tablas, bloques, etc.

DYNAMNBR[.nombre paso dentro procedimiento]=n

- **n** Número máximo asignaciones dinámicas (1-1635). Por defecto 0. 1635 no se aplica, sino que es el máximo número de ficheros que pueden asignarse en un momento determinado. Si se excede ese límite, el sistema lo asume.

3.1.8 PARM

Sirve para pasar algún tipo de información al programa cuando este se está ejecutando. Como máximo 100 caracteres.

PARM[.nombre paso dentro procedimiento]=parámetro

Si la información tiene más de una expresión, se separan por comas y se encierra el conjunto entre paréntesis o apóstrofos.

Ej: //PASO1 EXEC PGM=PROG1,PARM='PAR1,12345,PAR2=5'

//PASO2 EXEC PGM=PROG2,PARM=(PAR1,'PAR2=50')

3.1.9 PERFORM

Indica el grupo de especificaciones de rendimiento al que queda asignado el paso de trabajo.

PERFORM[.nombre paso dentro procedimiento]=n

- **n** (0-255). Por defecto:
  - **1** Pasos de un job que no es de TSO
  - **2** Pasos de un job de TSO

Si se codificó PERFORM en la sentencia JOB, todos los pasos del JOB quedan asignados al grupo de especificaciones de rendimiento asignado en el JOB.

3.1.10 RD

Controla las posibilidades de rearranque del paso donde se ha codificado.

RD[.nombre paso dentro procedimiento]= |R |
|RNC|
|NC |
|NR |

Lo mismo que para la RD de la ficha JOB.

3.1.11 REGION

Igual que el parámetro REGION de la sentencia JOB sólo que a nivel de paso de trabajo.

REGION[.nombre paso dentro procedimiento]=n K

- **nk** Número de k de memoria

Si en la sentencia JOB se codificó REGION, se ignoran todos los que se hayan codificado en las sentencias EXEC.

3.1.12 TIME

Lo mismo que para la TIME de la ficha JOB.

Si se codifica TIME=1440 (24 horas), el sistema supone que no existe limitación de tiempo de CPU para este paso.

Aunque se especificase TIME en la sentencia JOB, aquí cada una tiene validez en su entorno; pero cuando se cumpla el TIME del JOB, el trabajo terminará anormalmente.

4 SENTENCIA DD (Data Definition)

Nos va a permitir referenciar los ficheros con los que vamos a trabajar e indicarle al sistema si estos ficheros deben crearse, qué espacio necesitan, si es cinta, disco o fichas, su organización, etc. Cada sentencia DD describe un fichero.

4.1 Parámetros

4.1.1 (*) (posicional)

Indica que el fichero que define esta DD va a continuación de ella en la corriente de entrada.

El fin de un fichero se detecta mediante:

- Un delimitador standard : /* en columnas 1, 2
- Una sentencia de control: // en columnas 1, 2

Para indicar otro delimitador como fin de fichero se utilizará el parámetro DLM.

Esta sentencia no puede ir dentro de un procedimiento. Si se necesita se debe añadir al llamar al procedimiento.

Ej: //SYSIN DD *
DATO1
DATO2
…
DATON
/*

4.1.2 DATA (posicional)

Su utilización es la misma que la del anterior, con la diferencia de que se utiliza cuando entre los datos que componen el fichero figuran sentencias de control. (No puede ir dentro de un procedimiento).

Una ficha // en columnas 1 y 2 no actúa como delimitador de fin de fichero. Para indicar cual es el delimitador que actúa como delimitador de fin de fichero se utilizará el parámetro DLM.

4.1.3 DLM

Sirve para definir un delimitador distinto del standard (/*) y que no aparezcan en las dos primeras posiciones.

Si el delimitador contiene caracteres especiales, se debe encerrar entre apóstrofos.

Ej: //DATOS1 DD *.DLM=$@

//DATOS1 DD DATA.DLM='/$'

4.1.4 DUMMY (posicional)

Definimos el fichero como ficticio. El programa lo abrirá, realizará operaciones de e/s ficticias. El sistema ignorará estas peticiones, pero sin dar lugar a error de programa.

Si una sentencia DD está definida como DUMMY, todas las demás que vayan concatenadas a ella también se consideran DUMMY, las precedentes siguen siendo reales.

Ej: //DATOS1 DD DUMMY

4.1.5 DYNAM (posicional)

Asignación dinámica de ficheros. Si no se codifica el parámetro DYNAMNBR, en la sentencia EXEC, hay que colocar tantas DD DYNAM como asignaciones dinámicas vayan a existir en un momento determinado. Si se codifica, el número de asignaciones dinámicas es la suma del valor de DYNAMNBR más el número de DYNAM, y nunca debe exceder de 1635.

4.1.6 DDNAME

Para un programador COBOL, un fichero tiene tres nombres

- Nombre que el programador le da en el programa, definido en la SELECT.
- Nombre que va a tener en el sistema, especificado por el parámetro DSN.
- Nombre de enlace entre el nombre del fichero en el programa y el nombre del fichero en el sistema que es la DDNAME y que aparece en la corriente de control en la ficha DD correspondiente a ese fichero.

Ej: COBOL: SELECT nombre del fichero ASSIGN TO DDname

JCL: //DDname DD DSN=nombre en el sistema

Se utiliza cuando se desea diferir la definición del fichero para ponerla en una DD posterior y proporciona el enlace entre la DDNAME real y una DDNAME auxiliar.

Un caso muy claro se da en los casos de * y DATA que no pueden ir dentro de un procedimiento, el problema se resuelve difiriendo la definición del fichero para más adelante (fuera del procedimiento, cuando lo llamemos).

```
Ej:  //MAESTRO DD  DDNAME=AUXMAES
                (sentencia de control)
     //AUXMAES DD  *
                     (datos)
     /*
```

4.1.7 SYSOUT

Describe las características de un fichero de salida cuya trascripción va a efectuar el JES. Codificación:

```
SYSOUT=(clase)|,prog.||,form.|)
                      |,      ||,       |
```

- **clase** Clase de salida a la que queda asignado ese fichero (A-Z), (0-9), *. El * hace que la clase de salida sea la misma clase que la codificada en el parámetro MSGCLASS de la sentencia JOB.
- **programa** Nombre del programa que tendrá que escribirlo. Se sustituye por (,). Hay dos nombres reservados para el sistema:
  - **INTRDR** El fichero es tratado como si fuera una corriente de entrada de trabajos y grabado directamente sobre el 'spool' de entrada.
  - **STDWTR**
- **form** 1-4 caracteres no especiales. Identifica el tipo de formulario en que se debe escribir, perforar o grabar esa salida.

SYSOUT, es el parámetro de la sentencia DD que define un fichero de salida en impresora. Asociados a él vienen los parámetros: COPIES y HOLD.

Este parámetro es incompatible con DISP, VOLUME, LABEL.

Para las impresoras puede ser necesario codificar el DCB.

```
Ej:  //SALIDA1  DD  SYSOUT=D
     //SALIDA2  DD  SYSOUT=*
```

4.1.8 COPIES

Indica el número de copias (por defecto 1), que deseamos de un fichero de salida (impresora o grabadora).

Sólo puede ser codificado en una ficha del tipo: // DD SYSOUT=

COPIES= número (1-255)

Ej: //SALIDA1 DD SYSOUT=A,COPIES=32

4.1.9 DEST

Especifica el destino que se le quiere dar a un fichero de salida (SYSOUT). Puede ser a un dispositivo del sistema (LOCAL) o a un terminal remoto.

DEST=|Rnnn
|RMnnn
|RMTnnn
|Unnn
|LOCAL
|nombre

- **Rnnn, RMnnn, RMTnnn** Donde 'nnn' es un número de 1 a 3 dígitos que indica el terminal remoto al que se dirige la salida. R0 equivale a LOCAL.
- **Unnn** 'nnn' es un número (1-255) que indica el dispositivo local con destino especial al que va a dirigirse la salida.
- **LOCAL** Indica que el destino de salida es un dispositivo del sistema.
- **Nombre** (1-8) caracteres. Nombre del dispositivo local o remoto que va a recibir el fichero de salida.

Por defecto asume el mismo terminal desde el que se leyó el JOB.

Ej: //SALIDA1 DD SYSOUT=A,DEST=R555

4.1.10 FREE

Sirve para liberar ficheros.

FREE= |END |
|CLOSE|

- **END** Se libera cuando se termina el paso.
- **CLOSE** Se libera cuando se cierra.

Por defecto asume END.

No se debe codificar en un fichero que se abre y se cierra varias veces a lo largo del paso, pues la segunda vez que se abra, el paso terminará anormalmente.

Ej: //DD1 DD DSNAME=A,DISP=(NEW,PASS),FREE=END

4.1.11 FCB

Especifica el programa de control de carro para las impresoras controladas por programa en vez de por cinta.

FCB=(imagen |,ALIGN |)
|,VERIFY|

- **imagen** Identificación del programa a cargar en el buffer de la impresora (1-4 caracteres no especiales).
- **ALIGN** Pide al operador que compruebe que el formulario esté alineado.
- **VERIFY** Se imprime la imagen de caracteres seleccionada y permite comprobar que el formulario esté alineado.

Ej: //OUTPUT DD UNIT=3211,FCB=(FCB1,ALIGN)

4.1.12 HOLD

Indica que un fichero de salida quede retenido, y no empiece a salir hasta que el operador lo libere.

HOLD=|YES|
|NO |

Por defecto es NO (no queda retenido).

Sólo puede codificarse en un fichero SYSOUT.

4.1.13 UCS

Especifica el juego de caracteres que debe utilizar una impresora para imprimir un fichero.

UCS=(juego de caracteres [,FOLD] [,VERIFY])

[,]

- **juego de caracteres** 1-4 caracteres alfanuméricos. Identifica el juego de caracteres que se desea utilizar.
- **FOLD** Especifica que determinadas configuraciones del EBCDIC, se desea que se impriman como caracteres que tienen otra configuración (minúsculas por mayúsculas).
- **VERIFY** Se imprime una representación del juego de caracteres y el operador debe confirmar que es el que se desea.

Ej: // SAL DD SYSOUT=A,UCS=(YN,,VERIFY)

4.1.14 OUTLIM

Especifica el máximo número de registros lógicos que se desea componga un fichero SYSOUT.

OUTLIM=número (1-16777215)

Si no se codifica no hay límite.

Ej: // SAL DD SYSOUT=A,OUTLIM=1000

4.1.15 DSID

Especifica el identificador de un fichero de entrada o de salida en diskette 3540.

DSID=(id [,V])

- **id** Identificador de fichero (1-8 caracteres).
- **V** El fichero ha de verificarse antes de su proceso (sólo para entrada).

Ha de codificarse en una sentencia con *, DATA o SYSOUT. En caso contrario se le ignora.

Es mutuamente excluyente con los parámetros DYNAM, DDNAME o MSVGP.

```
Ej: //SYSIN  DD     *,DSID=(FICENT,V),VOLUME=SER=123456,
    //          DCB=LRECL=80
    //SYSPRINT DD  SYSOUT=E,DCB=LRECL=128,DSID=FICSAL
```

4.1.16 DSNAME (DSN)

Indica el nombre con el que el sistema debe localizar (si ya existe) o crear (si es nuevo) el fichero definido por la sentencia de control DD.

```
                    |nombre de fichero (1)
                    |fichero (nombre de miembro)(2)
                    |fichero (número de generación) (3)
                    |fichero (nombre de área) (4)
                    |nombre temporal del fichero (5)
DSNAME=             |nombre temporal del fichero (nombre de miembro) (6)
(DSN)               |nombre temporal del fichero (nombre de área) (7)
                    |*.'nombre de DD' (8)
                    |*.'nombre de paso'.'nombre de DD' (9)
                    |*.'nombre paso'.'paso dentro proc'.'nombre de DD'(10)
```

- **Permanentes**
  - **(1)** Nombre del fichero con el que se va a trabajar.
  - **(2)** Miembro de un fichero particionado.
  - **(3)** Grupos de generación. Es el conjunto de versiones de un fichero que se crean sin destruir las anteriores para poder acceder a una u otra según las necesidades (hasta 255). Referenciamos una versión:

    ```
    //   DD     GRUPO(2)
    ```

    Todas las versiones han de estar catalogadas.
  - **(4)** Secuencial-indexados.
- **Temporales** Desaparecen al terminar el JOB. Empiezan con & o &&. Si el nombre se deja en blanco o no se especifica el parámetro DSN, el sistema asume que es temporal y le da un nombre interno.
  - **(5)** Nombre del fichero.
  - **(6)** Fichero particionado.
  - **(7)** Secuencial-indexados.

- **Permanentes cualificados**
  - **(8) (9) (10)** Un nombre no cualificado está compuesto por 1-8 caracteres no especiales (excepto el guión). Uno cualificado se compone de varios no cualificados separados por puntos, máximo 44 caracteres (35 en grupos de generación). Los ficheros temporales no admiten cualificación. Hay un nombre especial de fichero: NULLFILLE. Si una sentencia DD tiene el operando DSN = NULLFILLE , a todos los efectos es como si esa sentencia se hubiera codificado como //DD DUMMY.
- **Indexados**

```
//DD1 DD    DSN=FICINDEX(INDEX) (área índices)
//      DD    DSN=FICINDEX(PRIME) (área primaria de datos)
//      DD    DSN=FICINDEX(OVFLOW) (área de overflow independiente)
```

Si las áreas residen en volúmenes de distinto tipo de dispositivo, se codifican las tres sentencias DD, que han de ir concatenadas. Sino, sólo se codifica el nombre del fichero en una DD sin especificar áreas.

4.1.17 DISP

Indica el estado en que se encuentra el fichero cuando se inicia el paso de trabajo y como se quiere que quede tras el paso.

```
                  [,DELETE  ]
            [NEW] [,KEEP    ] [,DELETE  ]
            [OLD ] [,PASS    ] [,KEEP    ]
DISP=(      [SHR ] [,CATLG   ] [,CATLG   ]    )
            [MOD] [,UNCATLG] [,UNCATLG]
            [,    ] [,          ]
```

- El primer subparámetro indica el estado del fichero cuando se arranca el paso (por defecto NEW):
  - **NEW** El fichero se crea en este paso.
  - **OLD** El fichero ya existe y no lo compartimos, ninguna otra tarea puede acceder a él.
  - **SHR** El fichero ya existe y lo compartimos.
  - **MOD** Si existe no lo compartimos y se posiciona al final del fichero. No reposiciona en caso de prog-check.

- El segundo parámetro indica como queremos que quede el fichero en caso de terminación normal del paso:
  - **DELETE** Borrarlo.
  - **KEEP** Guardar el fichero.
  - **PASS** El fichero se va a utilizar en pasos posteriores.
  - **CATLG** Guardar el fichero y catalogarlo en el catálogo del sistema o en el de usuario.
  - **UNCATLG** El fichero se guarda, pero se borran las entradas del catálogo.

  Por defecto asume DELETE para estado NEW y KEEP para OLD.

  Con PASS se conserva toda la información sobre el fichero. Con KEEP, hay que volver a introducirla.

  Al trabajar con cintas magnéticas si se utiliza KEEP, el sistema descarga la cinta al final del paso, y la volverá a pedir en pasos siguientes si se vuelve a utilizar. Con PASS, el sistema no la descarga y además guarda información sobre qué orden tenía el fichero utilizado dentro de la cinta, en el caso de cinta multifichero.

  En los ficheros temporales no se admite KEEP, si así se codifica, el sistema lo modifica y coloca PASS.

  En el caso de un fichero particionado, lo que indique este parámetro se aplica al fichero completo como tal, no a un miembro concreto.

- El tercer subparámetro indica como queremos que quede el fichero en el caso de que se produzca una terminación anormal del paso. Si se omite se asume el valor del segundo.

Si se omite el parámetro DISP, asume: DISP=(NEW,DELETE,DELETE)

4.1.18 SPACE

Indica cuánto espacio queremos asignar en un volumen de acceso directo para un fichero que se va a crear.

Es obligatorio codificarlo cuando se define un fichero en un dispositivo de acceso directo con DISP=NEW.

Dos posibilidades de codificación:

```
SPACE=( |TRK  |                              [,CONTIG]            )
        |CYL  |,(c.p[,c.s][,dir.])[,RLSE ][,MXIG   ] [,ROUND]
        |l.bloque|   [,c.s][,ind.][,       ][,ALX    ]
                                             [,        ]
SPACE=(ABSTR,(cant.primaria, direccion  [,directorio]))
                                             [,indice]
```

- Primera forma
  - El primer subparámetro indica la unidad de medida del espacio perdido.
    - TRK pistas
    - CYL cilindros
    - long-bloque el espacio se medirá en unidades de bloques
  - El segundo subparámetro da la cantidad de espacio que debe asignarse en principio para el fichero.
  - El tercer subparámetro da la cantidad de espacio que se le debe añadir al fichero si se necesita más espacio (máximo 16 extensiones en cada volumen).
  - El sentido del cuarto subparámetro cambia según se trate de un fichero:
    - Particionado. Número de bloques de 256 octetos que contendrá el directorio del fichero.
    - Secuencial-indexado. Número de cilindros que debe ocupar el área de índices del fichero.

  Por omisión es particionado.

    - **RLSE** Indica que el espacio que no se haya utilizado en la creación del fichero debe liberarse cuando se cierra.
    - **CONTIG** Indica que el espacio asignado para la petición primaria debe estar contiguo.
    - **MXIG** Se debe asignar para la partición primaria la mayor cantidad de espacio libre contiguo que haya en el volumen, siempre que sea igual o mayor que la cantidad primaria pedida.

- **ALX** Se debe asignar para la petición primaria hasta cinco áreas contiguas de tamaño igual o mayor que la cantidad pedida.
- **ROUND** Si se ha pedido espacio en unidades de longitud de bloque, obliga al sistema a redondearlo a un n. entero de cilindros.

❑ Segunda forma

- **ABSTR** El fichero se debe colocar en una dirección específica del volumen.
- **cant-primaria** Número de pistas que se deben asignar.
- **dirección** Dirección de la primera pista que debe asignarse.
- **[,directorio]**

  [,índice] Igual que en la primera forma.

4.1.19 VOLUME

Identifica el volumen donde el fichero reside o residirá si es nuevo.

```
                                                    [SER=n. serie]
VOLUME=([PRIVATE][,RETAIN][,secuencia][,cta][,]     [REF=fichero ])
VOL                                                 [REF=*.nombre de DD]
```

❑ **PRIVATE** El sistema no puede asignar espacio para ficheros en ese volumen a menos que la sentencia DD del fichero indique ese volumen. El sistema, salvo que se codifique RETAIN o se pase el fichero, lo desmontará al terminar su utilización en el paso de trabajo.

❑ **RETAIN** Indica que el volumen en cinta, definido como PRIVATE, no sea desmontado al final del paso.

❑ **secuencia** Indica, dentro de un fichero multivolumen, qué volumen deseamos utilizar en nuestro proceso.

❑ **cuenta** Número de volúmenes que necesita un fichero multivolumen de salida.

❑ **SER** Número de serie del volumen que deseamos.

- **REF** Deseamos el mismo volumen que el que:
  - **nombre de fichero** Contiene un fichero que vino de otro paso con DISP=(-PASS), o un fichero catalogado.
  - ***.'nombre de DD'** Que se utilizó en una sentencia DD anterior.

La información de volumen no es necesaria cuando se utilizan ficheros ya existentes catalogados.

4.1.20 UNIT

Se utiliza para especificar el tipo y número de unidades que se necesitan para el fichero que se está definiendo.

```
                [dirección      ]      [,cuenta]
UNIT=(          [tipo           ]      [,P    ]      [,DEFER])
                [nombre generico]      [,     ]
UNIT=AFF=nombre de DD
```

- Primera forma
  - **dirección** Dirección con la que identificar una unidad en particular.
  - **tipo** Nombre del sistema que identifica una serie de unidades (tipo de unidad)
  - **nombre genérico** Nombre (1-8 caracteres) que identifica a una serie de dispositivos. Se definen en generación.

    Ej: SYSDA (acceso directo) SISSQ (secuenciales)
  - **cuenta** Número de unidades del tipo genérico indicado que necesitamos para ese fichero.
  - **P** Indica que los volúmenes que se indican para ese fichero deben montarse en unidades distintas y a la vez.
  - **DEFER** El sistema asignará una o varias unidades para ese fichero, pero no pedirá que se le monten los volúmenes hasta el momento en que se abra el fichero.
- Segunda forma
  - **AFF** Indica que a este fichero se le debe asignar la misma unidad que al fichero definido en una sentencia DD anterior, pero dentro del mismo paso.

Este parámetro no es necesario si se trata de un fichero ya existente y catalogado.

Es obligatorio para ficheros que se crean en el paso. Si el fichero es temporal y en cinta es el único parámetro necesario de la DD.

4.1.21 LABEL

Especifica el tipo de etiquetas asociadas con el fichero de la sentencia DD donde se encuentre (sólo cintas).

LABEL=([secuencia][,etiquetas][,'password'][,proceso][,expiración])

- **secuencia** Posición relativa del fichero dentro de un volumen de cinta magnética. Por defecto 1.
- **etiquetas** Indica el tipo de etiquetas y el proceso a efectuar con ellas:
  - **SL** Standard IBM (por defecto).
  - **SUL** Standard IBM y del usuario.
  - **AL** Etiquetas ANS.
  - **AUL** ANS y del usuario.
  - **NSL** No tiene etiquetas standard.
  - **NL** Ningún tipo de etiquetas.
  - **LTM** El sistema debe ignorar una marca de cinta de relleno en la cabecera de la cinta, si la encuentra en cintas sin etiquetas.
  - **BLP** El sistema no debe procesar etiquetas de fichero de cinta.
- **'password'** Especifica la protección que deseamos que tenga el fichero que se está creando.
  - **PASSWORD** El fichero no puede ser leído, cambiado, escrito o borrado a menos que se introduzca la clave de protección.
  - **NOPWREAD** La clave de protección sólo afecta a escritura, actualización o borrado.
- **Proceso**
  - **IN** Un fichero BSAM no puede ser abierto más que para lectura.
  - **OUT** Un fichero BSAM no puede ser abierto más que para escritura.
- **expiración** Sirve para indicar hasta que fecha no se puede borrar o recubrir por otro fichero. EXPDT -> aaddd RETPD -> dddd (n. de días)

4.1.22 QNAME

Permite que un usuario tenga acceso a mensajes recibidos vía TCAM para que los procese un programa de aplicación.

QNAME=nombre(cola de destino de mensajes codificado en una macro TPROCESS)

Con este parámetro sólo se codifica DCB.

4.1.23 TERM

Indica que el fichero de entrada o de salida que se define con esa DD viene (o va) de un terminal de TSO.

TERM=TS

4.1.24 DCB

Describe internamente el fichero. Su codificación depende del método de acceso a utilizar y de la organización del fichero.

DCB=(operando1,operando2,...)

- Operandos
  - **RECFM** Tipo de registros del fichero.
    - **U** Indefinido.
    - **V** Longitud variable.
    - **VB** Longitud variable bloqueados.
    - **F** Longitud fija.
    - **FB** Longitud fija bloqueados.
  - **LRECL** Longitud en octetos del registro lógico.
  - **BLKSIZE** Longitud en octetos del bloque.
  - **DSORG** Organización del fichero.
    - **PS** Secuencial.
    - **IS** Secuencial-indexado.
    - **DA** Directa.
    - **PO** Particionado.
  - **DEN** Densidad de la cinta magnética. Las opciones más usuales son: 2 para 800 BPI, 3 para 1600 BPI, 4 para 6250 BPI.

- DCB ficheros IS Se utilizan los siguientes subparámetros:
  - **KEYLEN=número** Longitud en octetos de la clave del fichero.
  - **RKP=número** Número de octetos en el registro anteriores al primero de la clave
  - **OPTCD** Se utiliza para informar al sistema que el fichero se procesa con opciones especiales. Si se desean varias opciones se codificarán seguidas, sin comas ni blancos entre ellas.
    - **OPTCD=L** Indica que se borren los registros marcados con HIGH_VALUE en la primera posición cuando su espacio se necesite para otros registros. Y que no se recuperen los registros marcados con HIGH-VALUE en la primera posición cuando se lea el fichero secuencialmente. Para usar esta opción es necesario que el RKP sea > 0 en registro de longitud fija.
    - **OPTCD=M** El fichero tendrá un índice maestro si es necesario por su volumen. Al codificar esta opción se debe codificar también el subparámetro NTM.
    - **OPTCD=I** El fichero tiene un área de excedentes independiente.
    - **OPTCD=Y** Indica que se reserve espacio para áreas de excedentes de cilindro. Se usa en unión del parámetro CYLOFL. Por defecto asume Y.
  - **NTM=n. de pistas** Indica el número de pistas máximo que puede tener el índice de cilindros sin que se cree un índice maestro. Cada entrada el índice maestro apunta a una pista del índice de cilindros. Es necesario cuando se ha indicado la opción OPTC=M, si NTM no se codifica, se ignora la opción de índice maestro.
  - **CYLOFL=n. pistas** Indica el número de pistas de cada cilindro que se deben reservar como área de excedentes del cilindro. Es necesario si se ha codificado OPTCD=Y.
- DCB ficheros DA Se utilizan los siguientes subparámetros:
  - **LIMCTN=n. pistas** Indica el número de pistas máximo sobre el que se debe hacer una búsqueda extendida del registro a recuperar o espacio para grabar uno nuevo.
  - **OPTCD=E** Indica que se debe realizar una búsqueda extendida del registro

  Se utilizan ambas simultáneamente.

4.1.25 AMP

Similar al DCB pero para ficheros VSAM.

AMP=(AMORG,resto operandos)

4.1.26 CHKPT

Indica que se deben tomar puntos de control del fichero de la DD, cada vez que se encuentre una condición de fin de volumen.

CHKPT=EOV

Sólo para ficheros multivolumen con proceso BSAM o QSAM.

4.1.27 MSVGP

Especifica la identificación de un grupo de almacenamiento masivo que reside en un dispositivo de un sistema de almacenamiento masivo (MSS).

MSVGP=id

- **id** Identificador (1-8 caracteres) que define el grupo de volúmenes de almacenamiento masivo.

5 SENTENCIAS DD CON NOMBRES ESPECIALES

Estos nombres no se pueden utilizar en una sentencia DD normal ya que cada uno de ellos indica la existencia de una DD con una finalidad concreta.

- En definición de librerías:
  - JOBLIB
  - STEPLIB
- En definición de catálogos:
  - JOBCAT
  - STEPCAT
- En definición de volcados :
  - SYSABEND
  - SYSUDUMP

5.1 JOBLIB

Cuando el sistema encuentra una EXEC, busca ese programa en la librería standard del sistema SYS1.LINKLIB. Si queremos que la busque en una o más librerías distintas del sistema, se debe codificar una DD con nombre JOBLIB, y las concatenadas a ella necesarias.

- JOBLIB debe ser la primera sentencia DD del programa.
- No se debe usar JOBLIB dentro de un procedimiento catalogado.

Ej:
```
//CADENA01 JOB EJEMPLO,CLASS=A
//JOBLIB DD      DSNAME=PRUEBA.LIB,DISP=SHR
//PASO1  EXEC PGM=PROG1
//PASO2  EXEC PGM=PROG2
```

5.2 STEPLIB

Es lo mismo que la JOBLIB, sólo que a nivel de paso de programa.

- Si en un paso se codifica STEPLIB, esta anula la JOBLIB durante el paso, en caso de que hubiera sido codificada.
- Dentro de un procedimiento catalogado sí puede haber STEPLIB.
- Dentro de un paso la STEPLIB no tiene por qué ser la primera DD.

Ej: //PASO1 EXEC PGM=PROG1

//STEPLIB DD DSNAME=PRUEBA.LIB,DISP=SHR

5.3 JOBCAT

Se da este nombre a la sentencia DD que define al catálogo que ha de utilizar el sistema en la búsqueda de los ficheros de nuestro programa que se indiquen que están catalogados.

Si no los encuentra ahí, buscará en un catálogo privado caracterizado por la primera cualificación del nombre, o bien en el catálogo del sistema.

- Si se codifica una JOBCAT, debe ir precedida por JOBLIB, pero esta debe estar antes de la primera sentencia EXEC.

5.4 STEPCAT

Es lo mismo que la JOBCAT, pero a nivel de paso de programa.

- Si se codifica STEPCAT en un paso, se ignora la JOBCAT en este paso, si existiese.
- Puede parecer en cualquier punto dentro de las sentencias de control del paso.
- No se pueden definir CVOLs (catálogos OS privados) con STEPCAT. La única forma de hacerlo es por medio del catálogo maestro. Las sentencias JOBCAT y STEPCAT sólo pueden referirse a catálogos del usuario VSAM.

5.5 SYSABEND y SYSUDUMP

Nos definen ficheros donde se va a efectuar un vuelco de memoria en caso de que el paso termine anormalmente.

La diferencia que existe entre ambas reside en que parte de la memoria se vuelca el fichero:

SYSABEND	SYSUDUMP
Núcleo del sistema	Sólo el área del programa problema.
Área del programa problema	
Tabla de 'trace'	

El programa de aplicaciones utilizará en general SYSUDUMP.

5.6 SYSCHK

Define el fichero donde se grabaron los 'checkpoints' de un programa en una pasada anterior, y que ahora queremos rearrancar.

La DD con este nombre debe preceder a la primera EXEC.

6 CONCATENACIÓN DE SENTENCIAS DD

Tras una sentencia DD con nombre pueden codificarse otras DD sin nombre. El sistema asume que los ficheros descritos en las DD sin nombre son continuación del descrito en la DD con nombre.

```
//ENTRADA   DD   descripción fichero1
//          DD   descripción fichero2
//          DD   descripción fichero3
```

El programa que haga referencia a la DD con nombre ENTRADA tratará el fichero1, a continuación el 2 y finalmente el 3.

Las DD concatenadas únicamente tienen sentido para ficheros de entrada a un programa.

7 SENTENCIAS DE CONTROL DE COMANDOS

Formato: // <comando> <operandos del comando>

No todos los comandos pueden introducirse en el sistema por fichas. Los que pueden entrar por medio de sentencias de control son:

CANCEL	RELEASE	UNLOAD	STOPMN
DISPLAY	REPLY	VARY	
HOLD	RESET	WRITELOG	
LOG	SET	CHNGDUMP	
MODIFY	START	MONITOR	
MOUNT	STOP	SEND	

8 PROGRAMA PRODUCTO SORT/MERGE

Su función básica es la de clasificar ficheros de acuerdo a unas especificaciones determinadas o la de intercalar ficheros previamente ordenados (Se pueden ordenar hasta 255 ficheros secuenciales o 1 VSAM y se pueden intercalar de 2 a 16 ficheros que deberán estar ordenados según los mismos criterios).

Para la ejecución del sort/merge es necesario preparar dos tipos de sentencias:

- Sentencias de JCL.
- Sentencias de control para el sort.

Las primeras sirven para describir al sistema los ficheros necesarios para la ejecución del programa (librería sort, salida de mensajes, ficheros a clasificar/intercalar, fichero clasificado o de salida y ficheros de trabajo).

Las segundas sirven para controlar la ejecución del programa.

8.1 Sentencias de control de ordenación

Para una ordenación normal solamente se necesitan tres sentencias:

- Sentencia Sort. Proporciona la información sobre los campos de control y el tamaño del fichero de entrada y es obligatoria en todo proceso de ordenación.
- Sentencia Record. Sirve para indicar el formato y longitudes de los registros que trata el programa.
- Sentencia End. Señala el final de un grupo de sentencias de control del programa y se precisa si dichas sentencias no van seguidas de una ficha '/*'.

El programa verifica la validez de cada sentencia antes de empezar el proceso y si detecta un error envía un mensaje y cancela la ejecución.

Cada sentencia de control del programa SORT/MERGE se identifica por una palabra clave.

En un tipo de sentencia sólo debe aparecer una vez la palabra clave que identifica dicha sentencia.

La palabra clave debe empezar a partir de la columna 2 y debe ir seguida de, al menos, un espacio. Si una sentencia necesita más de una línea, se escribirá un carácter cualquiera en la columna 72 y se continuará en la siguiente línea.

Los operandos deberán ir separados por comas y sin blancos intercalados (Un blanco en los operandos indicará fin de sentencia y lo que sigue se considerará un comentario).

8.1.1 Sentencia SORT

La sentencia SORT se identifica por la palabra clave SORT.

```
SORT    FIELDS =(p,m,f,s...p,m,f,s)
        FIELDS =(p,m,s...p,m,s), FORMAT =f
        FIELDS =COPY
        [,CKPT]
        [,DYNALLOC=d|(d)|(,n)|(d,n)]
        [,EQUALS] [,NOEQUALS]
        [,FILSZ=x| ,SIZE=y| ,FILSZ=En| ,SIZE=En]
        [,SKIPREC=z]
        [,STOPAFT=n]
```

Los operandos fundamentales son FIELDS, SIZE y EQUALS.

- El operando FIELDS describe los campos a ser utilizados como campos de control para la ordenación, que quedarán determinados por: su posición, su longitud, su formato y por la forma de ordenación.
  - Posición (**p**) Se indica el primer octeto del campo mediante un número entero (El primer octeto de un registro es el número 1). Los registros de longitud variable llevan cuatro octetos por delante del registro por lo que, a la hora de determinar la posición, se deben tener en cuenta estos cuatro octetos.
  - Longitud (**m**) Se indica en octetos mediante un número entero.
  - Formato (**f**) Indica el formato en que debe considerarse el campo e influye sobre el parámetro longitud. Los códigos de formato mas utilizados son:
    - **CH** Caracteres EBCDIC
    - **AC** Caracteres ISCII/ASCII
    - **ZD** Decimal con signo
    - **PD** Decimal empaquetado con signo
    - **BI** Binario

- **FI** Binario con signo

▪ Secuencia de ordenación (**s**)

- **A** Ascendente
- **D** Descendente

Ej: SORT FIELDS=(1,6,A),FORMAT=BI

La clasificación se realizará por los seis primeros octetos del registro (empieza en la posición 1 y tiene 6 de longitud) y será de forma ascendente.

❑ El parámetro SIZE sirve para dar el número o una estimación del número de registros a ordenar.

Size=número

❑ El parámetro EQUALS indica que se preserve en salida el orden de entrada para registros con campos de control idénticos.

8.1.2 Sentencia RECORD

La sentencia RECORD se identifica por la palabra clave RECORD. Los operandos fundamentales son TYPE y LENGTH.

❑ El operando TYPE indica si los registros son de longitud fija (F) o variable (V).

❑ El operando LENGTH indica la longitud de los registros.

Ej: RECORD TYPE=F,LENGTH=350

8.1.3 Sentencia END

La sentencia END se identifica por la palabra clave END.

8.2 Sentencias de control de intercalación

Para una intercalación normal solamente se necesitan dos sentencias:

❑ Sentencia Merge. Proporciona la información sobre los campos de control y el tamaño del fichero de entrada y es obligatoria en todo proceso de intercalación. Su estructura es igual a la de la sentencia Sort.

❑ Sentencia End. Señala el final de las sentencias de control.

8.3 Ejemplo

```
1   //BRB0902S JOB MSGLEVEL=(1,1),MSGCLASS=X,CLASS=H,NOTIFY=BRB0902,
    //          COND=(0,LT)
2   //PASO1    EXEC PGM=SORT
3   //SYSPRINT DD SYSOUT=*
4   //SYSOUT   DD SYSOUT=*
5   //SORTIN   DD DSN=DOMP.RECIGENE.DISKPRKS,DISP=SHR
6   //SORTIN   DD LABEL=(,NL,EXPDT=98000),UNIT=581,
    //            VOL=SER=(QUINO1,QUINO2),
    //            DCB=(RECFM=FB,LRECL=800,BLKSIZE=9600)
7   //SORTIN   DD DSN=CULE.MAESCULT.CULBC002(0),DISP=SHR,UNIT=HTAPE
8   //SORTOUT  DD DSN=DOMP.PRUEBAS.DISKPRSD,DISP=(NEW,CATLG,DELETE),
    //            UNIT=SYSCR,SPACE=(CYL,(5,1),RLSE),
    //            DCB=(LRECL=350,BLKSIZE=3500,RECFM=FB)
9   //SORTOUT  DD DSN=PREP.PERIODI1.DISKPRSD,DISP=(NEW,CATLG,DELETE),
    //            UNIT=DISK,SPACE=(CYL,(45,10),RLSE),VOL=SER=338080,
    //            DCB=(RECFM=FB,LRECL=400,BLKSIZE=22400)
10  //SORTOUT  DD DSN=AHOP.SALCLISD.DISKPRSD,DISP=(,PASS),
    //            UNIT=SYSCR,SPACE=(CYL,(10,3),RLSE),
    //            DCB=(RECFM=VB,BLKSIZE=16000)
11  //SORTWK01 DD UNIT=SYSCR,SPACE=(CYL,(15,5)),VOL=SER=338080
12  //SORTWK02 DD UNIT=SYSCR,SPACE=(CYL,(15,5)),VOL=SER=338080
13  //SORTWK03 DD UNIT=SYSCR,SPACE=(CYL,(15,5)),VOL=SER=338080
14  //SORTWK04 DD UNIT=SYSCR,SPACE=(CYL,(15,5)),VOL=SER=338080
15  //SORTWK05 DD UNIT=SYSCR,SPACE=(CYL,(15,5)),VOL=SER=338080
16  //SORTWK06 DD UNIT=SYSCR,SPACE=(CYL,(15,5)),VOL=SER=338080
17  //SORTWK01 DD UNIT=SYSCR,SPACE=(CYL,3)
18  //SORTWK02 DD UNIT=SYSCR,SPACE=(CYL,3)
19  //SORTWK03 DD UNIT=SYSCR,SPACE=(CYL,3)
20  //SORTWK04 DD UNIT=SYSCR,SPACE=(CYL,3)
21  //SORTWK05 DD UNIT=SYSCR,SPACE=(CYL,3)
22  //SORTWK06 DD UNIT=SYSCR,SPACE=(CYL,3)
23  //SYSIN    DD *
24   INCLUDE COND=(1,5,EQ,C'00000',OR,29,1,EQ,C'2'),FORMAT=CH
25   INCLUDE COND=(1,6,EQ,C'C00001',&,7,6,LE,C'000400'),FORMAT=BI
26   INCLUDE COND=(1,1,EQ,X'F1')
27   OMIT COND=(11,3,EQ,C'001'),FORMAT=BI
28   OMIT COND=(219,3,LT,X'00701C',OR,219,3,GT,X'00709C'),FORMAT=BI
29   OMIT COND=((5,2,GE,X'3000',AND,5,2,LE,X'300F'),OR,           C
                (5,2,GE,X'3020',AND,5,2,LE,X'302F'),OR,           C
                (5,2,GE,X'3040',AND,5,2,LE,X'304F'),OR,           C
                (5,2,GE,X'3600',AND,5,2,LE,X'360F')),FORMAT=BI
30   SORT FIELDS=(1,6,A),FORMAT=BI
31   SORT FIELDS=(156,3,D,42,4,A,24,9,A),FORMAT=BI,WORK=DA
32   SORT FIELDS=(2,19,CH,A,1,1,CH,A),EQUALS
33   OUTREC FIELDS=(22X,1,27,2Z,28,224)
34   INREC FIELDS=(1,27)
35   SUM FIELDS=(19,7,PD)
36   OPTION SKIPREC=10,STOPAFT=500
37   RECORD TYPE=F,LENGTH=350
38   END
39  //SYSIN    DD *
40   MERGE FIELDS=(1,19,A,110,5,A),FORMAT=BI
41   RECORD TYPE=F,LENGTH=19
42   END
```

- FICHA: 1

  Identifica el trabajo para el sistema.

- FICHA: 2

  Identifica un paso del JOB. En este caso el paso hará que se ejecute el SORT.

- FICHA: 3-4

  Fichas que indican la salida de mensajes (La ficha SYSPRINT no tiene ningún valor).

- FICHA: 5 a 7

  Identifican los ficheros de entrada. Sólo puede haber un SORTIN aunque, dentro de él, puede haber más de una ficha DD.

- FICHA: 8 a 10

  Identifican el fichero de salida. Sólo puede haber un fichero de salida.

- FICHA: 11 a 22

  Identifican los ficheros de trabajo. Aquí aparecen dos formas de identificarlos (fichas 11 a 16 y fichas 17 a 22). El número de ficheros de trabajo es variable (no tiene por qué ser 6).

- FICHA: 23

  Nos indica que, a continuación, vienen los parámetros del SORT.

- FICHA: 24 a 26 – INCLUDE

  Nos indican qué condiciones deben cumplir los registros del/los fichero/s de entrada para que se incluyan en el fichero de salida. Los que no cumplan esas condiciones no se incluirán en dicho fichero. Sólo puede haber una ficha INCLUDE.

EQ=Igual	OR=o	C=Carácter
NE=No igual	&=y	X=Hexadecimal
GT=Mayor que		
GE=Mayor o igual		
LT=Menor que		
LE=Menor o igual		

  Por ejemplo, la ficha 24 nos indica que sólo se llevarán al fichero de salida aquellos registros que en la 5 primeras posiciones (empiezan en la 1 y tiene 5 de longitud) tengan 5 ceros (en formato carácter) o que

en la posición 29 (empieza en la 29 y tiene 1 de longitud) tengan un 2 (en formato carácter).

- FICHA: 27 a 29 - OMIT

  Nos indican qué condiciones deben cumplir los registros del/los fichero/s de entrada para que NO se incluyan en el fichero de salida. Los que no cumplan esas condiciones SI se incluirán en dicho fichero. Sólo puede haber una ficha OMIT.

  El formato es el mismo de la ficha INCLUDE.

  Las fichas OMIT e INCLUDE son incompatibles.

  Por ejemplo, la ficha 27 nos indica que los registros que en la posición 11 a 13 (empieza en la 11 con 3 de longitud) tengan '001' (en formato carácter) no se incluirán en el fichero de salida.

- FICHA: 30 a 32 - SORT FIELDS

  Nos indican el criterio de clasificación de los registros.

  Por ejemplo, la ficha 31 nos indica que el criterio de clasificación es el siguiente:

  - Primero se clasificará por las posiciones 156 a 158 (empieza en la 156 y tiene 3 de longitud) de forma descendente (de menor a mayor).
  - Después se clasificará por las posiciones 42 a 45 de forma ascendente (de mayor a menor).
  - Por último, se clasificará por las posiciones 24 a 32 de forma ascendente.
  - Los campos, para estas operaciones, deberán considerarse en formato binario (FORMAT=BI) y el espacio para los ficheros de trabajo se tomará de un dispositivo de acceso directo (WORK=DA). (Este último parámetro no es necesario).

  Si se utilizara el parámetro EQUALS (como en la ficha 32) nos indicaría que, para los registros en los que los campos a partir de los cuales se realiza la clasificación fueran iguales, se respetase el orden de entrada (es decir, en el fichero de salida estarán en el mismo orden que tenían en el fichero de entrada).

- FICHA: 33 a 34 - OUTREC, INREC

  Nos van a permitir "diseñar" el registro de salida. Por ejemplo, en la ficha 33 decimos que el registro de salida se configure de la siguiente manera:

- Las 22 primeras posiciones a blancos (22X).
- Las siguientes 27 posiciones con las 27 primeras del registro de entrada (1,27).
- Las siguientes 2 posiciones a ceros binarios (2Z).
- Las últimas 224 posiciones con las que van de la 28 a la 251 del fichero de entrada (28,224).

La INREC tiene el mismo formato que la OUTREC. La única diferencia entre ambas es que OUTREC modifica el registro después de realizado el sort e INREC lo hace antes.

- FICHA: 35 - SUM

  Nos permite sumar los valores de las posiciones especificadas creando un solo registro por cada clave igual (campos que cumplan los criterios dados en la ficha SORT) con la suma que se pidió.

  Por ejemplo, en esta ficha 35 se pide que se sumen los valores de las posiciones 19 a 25 (19,7) en formato decimal empaquetado con signo (PD).

  Si se produce overflow (se sobrepasa la capacidad del campo) se crea otro registro donde se continua con la suma.

- FICHA: 36 - OPTION

  Nos permite darle al sort una serie de opciones. Por ejemplo:

  - SKIPREC, indicará el número de registro del fichero de entrada en el que comenzará a realizarse el sort. (En este caso, en el registro 10).
  - STOPAFT, indicará el número máximo de registros que se aceptan para el sort. (En este caso son 500).

- FICHA: 37 - RECORD

  Nos indica el formato y la longitud de los registros que trata el programa.

  En este caso los registros son de longitud fija (F) de 350.

- FICHA: 38 - END

  Nos indica el final de las instrucciones del sort.

- FICHA: 39 a 42

  Instrucciones para la intercalación (MERGE).

  Los formatos son iguales a los del SORT.

9 UTILITY

Una UTILITY es un programa de utilidad proporcionado con el sistema operativo.

En OS/VS hay tres tipos de UTILITYS:

- Programas de utilidad del sistema (se utilizan para mantener datos de control del sistema).
- Programas de utilidad de ficheros (se utilizan para crear, modificar, listar o reorganizar ficheros).
- Programas de utilidad independientes (se utilizan para preparar dispositivos para uso del sistema).

9.1 IEFBR14

```
//* Borrado de un fichero
//BORRADO   EXEC PGM=IEFBR14
//DD1       DD DSN=DLIP.CUENTAS.PREREO,DISP=(OLD,DELETE)
//DD2       DD DSN=DLIP.SECOND.INDEX,DISP=(OLD,DELETE)
//DD3       DD DSN=DLIP.INDICE.UNLD,DISP=(OLD,DELETE)
//DD4       DD DSN=DLIP.TRABAJO.REL,DISP=(OLD,DELETE)
//SYSPRINT  DD SYSOUT=*
//SYSOUT    DD SYSOUT=*
//

//* Creación de un fichero (vacío)
//CREAR     EXEC  PGM=IEFBR14
//SYSPRINT  DD  SYSOUT=*
//SYSOUT    DD  SYSOUT=*
//SYSUDUMP  DD  SYSOUT=*
//SYSDBOUT  DD  SYSOUT=*
//TRANOMSD  DD  DSN=XXXX.XXXXXXX.DISKPRSD,DISP=(NEW,CATLG,CATLG),
//            DCB=(LRECL=180,BLKSIZE=18000,RECFM=FB),UNIT=SYSCR,
```

```
//              SPACE=(CYL,(1,1),RLSE)
```

9.2 IEBCOPY

```
//* Reorganizar librerías
//COPIA       EXEC PGM=IEBCOPY
//SYSPRINT    DD SYSOUT=X
//LIBRERÍA    DD DSNAME=CALP.XXXXXX.FUENTE,UNIT=SYSCR,DISP=SHR
//COPIALIB    DD DSNAME=CALP.XXXXXX.FUENTE,DISP=SHR
//SYSUT3      DD UNIT=SYSCR,SPACE=(TRK,(1))
//SYSUT4      DD UNIT=SYSCR,SPACE=(TRK,(1))
//SYSIN       DD *
COPYLIB    COPY OUTDD=COPIALIB,INDD=LIBRERIA
/*

//* Copiar miembros de una librería
//XXXXXX1    JOB  CLASS=G,MSGCLASS=X,NOTIFY=XXXXXX,MSGLEVEL=(0,0)
//PASO010    EXEC PGM=IEBCOPY
//SYSPRINT   DD     SYSOUT=*
//ENTRADA    DD     DSN=CALP.XXXXXX.FUENTE,DISP=SHR
//SALIDA     DD     DSN=CALP.XXXXXX.FUENTE,DISP=SHR
//SYSUT3     DD     UNIT=SYSCR,SPACE=(TRK,(1))
//SYSUT4     DD     UNIT=SYSCR,SPACE=(TRK,(1))
//SYSIN      DD *
     COPY OUTDD=SALIDA,INDD=ENTRADA
     SELECT MEMBER=((MEMBER,,R))
/*
```

9.3 AMBLIST

```
//* Listar módulos ejecutables
//XXXXXX1   JOB  CLASS=G,MSGCLASS=X,NOTIFY=XXXXXX,MSGLEVEL=(1,1)
//PASO020   EXEC PGM=AMBLIST
//SYSPRINT  DD SYSOUT=*
//SYSLIB    DD DSN=CALP.LINKBAT,DISP=SHR
//SYSIN     DD *
 LISTLOAD OUTPUT=XREF,TITLE=('PUNTOS DE ENTRADA',50),MEMBER=CULBC012
/*
```

9.4 IEBGENER

```
//* Imprime fuentes
//HCFUEN    EXEC PGM=IEBGENER,COND=EVEN
//SYSIN     DD *
//SYSUT1    DD DSN=CALP.XXXXXXX.FUENTE(XXXXXXX),
//           DISP=(SHR,KEEP)
//SYSUT2    DD SYSOUT=9,
//           DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSPRINT  DD DUMMY
//SYSIN     DD DUMMY
```

9.5 IEBPTPCH

Su función principal es la de listar un fichero secuencial o una parte de él.

Las especificaciones estándar son:

- Cada registro lógico empieza en una nueva línea de la página.
- Cada línea impresa consta de grupos de 8 caracteres separados por 2 blancos.

- En caso de representación carácter, los caracteres hexadecimales que no tienen representación se dejan en blanco.
- Cuando el fichero de entrada está bloqueado, en el listado se señala:
  - Con '*' el final del registro lógico.
  - Con '**' el final del bloque.

9.5.1 Ficheros de entrada y de salida:

El IEBPTPCH tiene las siguientes entradas:

- Un fichero de entrada que contiene datos a imprimir y que debe ser secuencial.
  - Los registros del fichero de entrada pueden ser de longitud fija o variable.
  - Si la longitud del registro de entrada es superior a una línea impresa, la utility divide el registro en tantas líneas como sean necesarias.
- Un fichero de control que contiene las informaciones de control para la utility. Estos datos de control son necesarios para todos los usos del IEBPYPCH y, normalmente, están incluidos en la misma corriente de control.

El IEBPTPCH tiene las siguientes salidas:

- Un fichero de salida en impresora.
- Un fichero de mensajes, normalmente es una impresora en la que se escriben mensajes informativos y/o de error generados por la utility.

9.5.2 Codificación

```
//LISTAR      EXEC PGM=IEBPTPCH
//SYSPRINT    DD SYSOUT=X
//SYSUT1      DD DSN=PREP.HISTORIC.DISKPRSD,DISP=SHR
//SYSUT2      DD SYSOUT=X
//SYSIN       DD *
    PRINT TOTCONV=XE,STOPAFT=25,MAXLINE=66
    TITLE ITEM=('LISTADO REGS.PRUEBA CONVERSION CINTA')
/*
```

9.5.3 Instrucciones de control

Las opciones del IEBPTPCH se controlan mediante las instrucciones de control PRINT, TITLE y RECORD.

- **PRINT** Indica a la utility que se debe iniciar la impresión del fichero.

  PRINT [TOTCONV=XE][,STRTAFT=N][,STOPAFT=N][,MAXFLDS=N]
  [,INITPG=N][,MAXLINE=N]

  - **TOTCONV=XE** Indica que los datos hay que listarlos en hexadecimal en vez de en caracteres.
  - **STRTAFT=N** Indica el número de registros del fichero de entrada que deben saltarse antes de comenzar la impresión. Si no se codifica se lista desde el primer registro del fichero. Si se codifica, el valor de 'N' no puede ser mayor de '32767'.
  - **STOPAFT=N** Indica el número de registros del fichero de entrada que se deben de imprimir. La impresión finaliza cuando se haya listado el número de registros indicado en 'N' o al detectar el final del fichero. Si no se codifica, se listan registros hasta detectar el final del fichero. Si se codifica, el valor de 'N' no puede ser mayor de '32767'.
  - **MAXFLDS=N** Indica el número de parámetros FIELD codificados en una instrucción RECORD posterior.
  - **INITPG=N** Indica el primer número de página del listado. A partir de este número las páginas se numeran secuencialmente de 1 en 1. Si no se codifica se asumen 1. Si se codifica, el valor de 'N' no puede ser mayor de 9999.
  - **MAXLINE=N** Indica el número máximo de líneas por página, incluidas líneas en blanco, títulos y subtítulos. Si no se codifica asume 60.

- **TITLE** Indica a la utility los títulos del listado. Permite un título y un subtítulo. Si se codifica esta sentencia debe ir a continuación de la PRINT.

  TITLE ITEM=('Título'[,Situación-salida])[,ITEM=...]

  - En el primer campo ITEM se codifica el título y en el segundo el subtítulo.
  - **'Título'**. Es el literal del título o subtítulo. Puede contener hasta 40 caracteres encerrados entre comillas.

- **'Situación-salida'**. Indica la posición de comienzo del literal en la línea de impresión. Si no se codifica asume 1.

❑ **RECORD** Indica a la utility los campos del registro que debe de imprimir.

RECORD FIELD=(Longitud[,Situación-entrada][,Conversión]
[,Situación-salida])[,FIELD=...]

- Cada parámetro FIELD indica un campo a listar.
- **'Longitud'.** Es la longitud en bytes del campo en el registro de entrada.
- **'Situación-entrada'.** Es la posición en la que comienza el campo en el registro de entrada. Si no se codifica asume 1.
- **'Conversión'.** Indica, en dos bytes, el tipo de conversión que se debe realizar con el campo de entrada. Si no se codifica se mueve el campo a la línea de impresión tal y como está. Si se codifica puede ser 'PZ' (indica que hay que desempaquetarlo) o 'XE' (indica que hay que listarlo en hexadecimal).
- **'Situación-salida'.** Es la posición a partir de la que hay que listar el campo en la línea de impresión. Si no se codifica asume 1.

9.5.4 Ejemplo

El fichero de facturas pendientes es secuencial, reside en disco y tiene el siguiente formato:

Número de Factura		Número de Cliente		Nombre del Cliente		Importe de la Factura		Fecha de la Factura			
1	6	7	12	13	52	53	59	60	65	66	100

El importe de la factura tiene 13 caracteres numéricos empaquetados.

Se desean listar los siguientes campos de cada registro del fichero:

❑ Número de factura en las posiciones 20 a 25.

❑ Importe de la factura, desempaquetado, en las posiciones 30 a 42.

❑ Fecha de la factura en las posiciones 45 a 49.

El listado debe llevar el titulo 'listado de facturas' a partir de la columna 23.

```
//LISTAR      EXEC PGM=IEBPTPCH
//SYSPRINT    DD SYSOUT=X
//SYSUT1      DD DSN=PREP.facpendi.DISKPRSD,DISP=SHR
//SYSUT2      DD SYSOUT=X
//SYSIN       DD *
     PRINT  MAXFLDS=3
     TITLE  ITEM=('LISTADO DE FACTURAS',23)
     RECORD FIELD=(6,1,,20),FIELD=(7,53,PZ,30),FIELD=(6,60,,45)
/*
```

9.6 IEBCOMPR

```
//* Compara dos ficheros
//COMPARE    EXEC PGM=IEBCOMPR
//SYSUT1     DD DSN=DB2P.DB2BC008.DISKPRSD,DISP=OLD
//SYSUT2     DD DSN=DB2P.VIEJO008.DISKPRSD,DISP=OLD
//SYSPRINT   DD SYSOUT=X
//SYSOUT     DD SYSOUT=X
//SYSIN      DD *
     COMPARE TYPORG=PS
/*
```

9.7 IEBDKRDR

```
//* Lectura diskette
//PASO0      EXEC   PGM=IEBDKRDR,PARM=50
//SYSDATA    DD  UNIT=007
//SYSUT2     DD  SYSOUT=(Z,INTRDR)
//SYSUT3     DD  UNIT=SYSCR,SPACE=(128,(5000,5000,3)),
//             DCB=BLKSIZE=4096
```

```
//IEFRDER     DD  DATA,DLM='ZZ'
```

10 PARÁMETROS EN LA EXEC.

Se puede pasar un parámetro en la ficha EXEC de una JCL, de la forma:

```
//PASO EXEC PGM=PROGRAMA,PARM=parámetro        (ver JCLDOC).
```

Este parámetro se recoge en variables de la LINKAGE SECTION:

```
01  PARAMETROS.
        03 PARMLONG       PIC S9(4) COMP-3. (**)
        03 nombre-var     PIC tipo(long).
        03 …
        03 …
        03 nombre-var     PIC tipo(long).
```

La procedure será como la de cualquier programa invocado:

```
PROCEDURE DIVISON USING PARAMETROS.
```

(**) Almacena la longitud del parámetro pasado.

11 DISTINTOS EJEMPLOS DE CADENAS.

11.1 Ejemplo DB2

```
 1  //BRB0902S JOB (CONTAB),' MJDG ',CLASS=G,
    //             MSGLEVEL=(1,1),MSGCLASS=X,NOTIFY=BRB0902,
    //             USER=SYSDBP,PASSWORD=SYSDBP
 2  //JOBLIB  DD  DSN=CALP.LINKBAT,DISP=SHR
    //         DD  DSN=SYS1.DSN210.DSNLOAD,DISP=SHR
 3  //PASO01   EXEC PGM=IKJEFT01,DYNAMNBR=20
 4  //SYSOUT   DD SYSOUT=*
 5  //SYSPRINT  DD SYSOUT=*
 6  //SYSTSPRT DD SYSOUT=*
 7  //SYSUDUMP  DD SYSOUT=*
 8  //SYSDBOUT  DD SYSOUT=*
 9  //MOVCORSD DD *
    3040000662821 890125 890124 6 59 714 +00000380000
    3040090694064 890209 880706 6 66 337 +00000010000
10  //FICCON01 DD  *
    3040000XXXXXX
11  //CINREDB2  DD DSN=XXXX.XXXXXXXX.XXXXXXXX,DISP=(,CATLG,DELETE),
    //             UNIT=SYSCR,SPACE=(CYL,(5,1),RLSE),
    //             DCB=(RECFM=FB,LRECL=49,BLKSIZE=4900)
12  //IMPRES01 DD  SYSOUT=9,
    //         DCB=(RECFM=FA,LRECL=133,BLKSIZE=133)
13  //SYSTSIN  DD *
14  DSN SYSTEM(DB3) RETRY(0) TEST(0)
15  RUN PROGRAM(PROCOB9) PLAN(PROCOB9) -
        LIB('CALP.LINKBAT')
16  END
    /*
```

```
 1  //BRB0902S JOB MSGLEVEL=(1,1),MSGCLASS=X,CLASS=H,NOTIFY=BRB0902,
    //          COND=(0,LT),TYPRUN=HOLD,TIME=1440
 2  //*
 3  //JOBLIB  DD DSN=CALP.LINKBAT,DISP=SHR
 4  //PASO0    EXEC PGM=XXXXXXXX,PARM=2035
 5  //PASO1    EXEC  RDRDKT
 6  //RDRDKT.IEFRDER  DD DATA,DLM='ZZ'
 7  //XXXXXXXP JOB  CLASS=C,MSGCLASS=W,MSGLEVEL=(1,1),COND=(0,LT)
 8  //*
 9  //PASO2    EXEC   PGM=SQLEPRES,REGION=4096K
10  //STEPLIB  DD  DSN=CALE.LINKBAT,DISP=SHR
11  //SYSOUT   DD  SYSOUT=*
12  //SYSUDUMP DD  SYSOUT=*
13  //SYSDBOUT DD  SYSOUT=*
14  //FICHERO1 DD  LABEL=(,NL,EXPDT=98000),UNIT=581,
    //             VOL=SER=(QUINO1,QUINO2),
    //             DCB=(RECFM=FB,LRECL=800,BLKSIZE=9600)
15  //DBPRESTD DD  DSN=PREE.BDPRE418.AL311286,DISP=OLD,UNIT=HTAPE,
    //             VOL=SER=(800196)
16  //DBCIN3TD DD  DUMMY
17  //SALAHOKD DD  DUMMY,AMP=AMORG
18  //HISTORA1 DD  DSN=PREE.HISTOPRE.DE77AFIN(0),DISP=SHR,UNIT=HTAPE
19  //PERIODSD DD  DSN=PREP.PERIODI1.DISKPRSD,DISP=(NEW,CATLG,DELETE),
    //             UNIT=SYSCR,SPACE=(CYL,(45,10),RLSE),VOL=SER=338080,
    //             DCB=(RECFM=FB,LRECL=400,BLKSIZE=22400)
20  //FICHERO4 DD  DUMMY,
    //             UNIT=SYSCR,SPACE=(TRK,(10,5),RLSE),
    //             DCB=(RECFM=VB,LRECL=300,BLKSIZE=30000)
21  //SQLPEST2 DD  DSN=PREP.BASESQL1.SQLEPRES,DISP=(NEW,KEEP),
    //             UNIT=HTAPE,LABEL=(1,SL,EXPDT=98004),
    //             DCB=(RECFM=FB,LRECL=300,BLKSIZE=30000)
22  //CLIENTSD DD  DSN=AHOP.SALCLISD.DISKPRSD,DISP=(,PASS),
    //             UNIT=SYSCR,SPACE=(CYL,(10,3),RLSE),
    //             DCB=(RECFM=VB,BLKSIZE=16000)
23  //SYSLIN   DD  DSN=&&LOADSET,UNIT=SYSDA,DISP=(MOD,PASS),
    //             SPACE=(TRK,(3,3)),DCB=BLKSIZE=800
24  //IMPRES01 DD  SYSOUT=(X,,STD),FCB=3262,COPIES=2,
    //             DCB=(LRECL=00133,BLKSIZE=00133,RECFM=FA)
25  //FICCON01 DD *
26  13M 24-04-89
27  070587        C00003      MAYO
28  /*
29  //* ULTIMO PASO
30  //PASO3    EXEC   PGM=XXXXXXXX
31  //SYSOUT   DD SYSOUT=*
32  //SYSUDUMP DD SYSOUT=*
33  //SYSDBOUT DD SYSOUT=*
34  //PREPERSD DD  DSN=PREP.PERIODI1.DISKPRSD,DISP=(SHR,DELETE,KEEP)
35  //CTACLISD DD  DSN=CLIP.CTASCLTE.DISKPRSD,DISP=SHR
36  //HISTORN2 DD  DSN=PREE.HISTOPRE.DE77AFIN(+1),DISP=(,CATLG),
    //             UNIT=HTAPE
37  //SYSLIN   DD  DSN=&&LOADSET,DISP=(OLD,DELETE)
38  //IMPRES01 DD SYSOUT=9,
    //            DCB=(LRECL=00133,BLKSIZE=00133,RECFM=FA)
39  //IMPRES02 DD SYSOUT=(X,,STD),FCB=3262,
    //            DCB=(LRECL=133,BLKSIZE=0133,RECFM=FA),FREE=CLOSE
40  ZZ
41  //
```

- FICHA: 1 y 7

  Identifican el trabajo para el sistema.

  - BRB0902S Nombre del JOB.
  - JOB Nombre de la operación.
  - MSGLEVEL=(1,1) Indica que se impriman todas las sentencias de control y todos los mensajes de asignación/terminación.
  - MSGCLASS=X Especifica que la clase de salida donde queremos que vayan todos los mensajes del sistema y las sentencias de control es la "X".
  - CLASS=H Indica que la cola de entrada donde se va a colocar el JOB en espera de su ejecución es la clase "H".
  - NOTIFY=BRB0902 Le indica al sistema que envíe un aviso al usuario de TSO 'BRB0902' cuando termine la ejecución del trabajo.
  - COND=(0,LT) Especifica que el JOB se debe de dejar de ejecutar si algún paso devuelve un código de retorno mayor de '0'.
  - TYPRUN=HOLD Indica que el JOB se coloca en cola de entrada pero que no se llamará a ejecución hasta que sea liberado por un operador.
  - TIME=1440 Indica que el JOB no tiene limitación de tiempo.

- FICHA: 2-8-29

  Nos indican líneas de comentarios.

- FICHA: 3 - JOBLIB

  Indica que los programas especificados en las fichas EXEC deben buscarse en la librería 'CALP.LINKBAT' que ya existe y se comparte (DISP=SHR).

- FICHA: 4-5-9-30

  Identifican pasos de trabajo dentro del JOB.

  - PASOX. Nombre del paso.
  - PGM=XXXXXXXX. Indica el nombre del programa que se va a ejecutar.
  - PARM=2035. Indica que se debe pasar la constante '2035' al programa que se está ejecutando.
  - REGION=4096K. Se limita a 4096 k's la memoria virtual que se puede tomar.

La sentencia DD nos va a permitir referenciar los ficheros con los que se va a trabajar.

Su codificación es la siguiente:

//NOMBRE DD PARAMETROS COMENTARIOS

- FICHA: 6 - DATA

  Nos indica que entre los datos que componen el fichero figuran sentencias de control. Para indicar cual es el delimitador que actúa como fin de fichero se utilizara el parámetro DLM.

  DLM='ZZ'. Nos define un delimitador distinto del estándar (en este caso 'ZZ') que nos indicará el fin del fichero. (En este caso el fin de fichero se especifica en la ficha 40).

- FICHA: 10 - STEPLIB

  Indica que el programa especificado en la ficha EXEC de ese paso debe buscarse en la librería 'CALE.LINKBAT' que ya existe y que se comparte (DISP=SHR).

- FICHA: 11-31 - SYSOUT

  Indica que el fichero queda asignado a la misma clase de salida que la especificada en el parámetro MSGCLASS.

- FICHA: 12-13-32-33 - SYSUDUMP, SYSDBOUT

  Nos definen ficheros donde se va a efectuar un vuelco de memoria en caso de que el paso termine anormalmente.

- FICHA: 14

  - FICHERO1. DDname del fichero a utilizar.
  - LABEL. Especifica el tipo de etiquetas de la cinta.
    - **,** Lugar para la secuencia.
    - **NL** Sin etiquetas.
    - **EXPDT** Expiración.
  - UNIT=581. Nos indica que la cinta debe de ser montada en la unidad 581.
  - VOL=SER. Nos indica el volumen donde reside y el número de serie del fichero que deseamos (QUINO1 y QUINO2).

- DCB. Nos describe internamente el fichero.
  - **RECFM=FB** Los registros son de longitud fija y bloqueados.
  - **LRECL=800** Los registros son de 800 octetos.
  - **BLKSIZE=9600** El bloque es de 9600 octetos (debe ser un múltiplo de la longitud de registro).

❑ FICHA: 15

Nos indica que el fichero a utilizar tienen por DDNAME 'DBPRESTD' y como DSNAME 'PREE.BDPRE418.AL311286', que es de utilización compartida (DISP=SHR), que está en un dispositivo HTAPE y que el volumen y el número de serie en donde se encuentra es el 800196.

❑ FICHA: 16-17

Los ficheros 'DBCIN3TD' y 'SALAHOKD' se definen como ficticios (el parámetro AMP=AMORG se pondrá en los VSAM al definirlos como DUMMY).

❑ FICHA: 18

Nos indica que el fichero HISTORA1 que se encuentra en el dispositivo HTAPE y que será compartido (SHR), es el primero del grupo de generación 'PREE.HISTOPRE.DE77AFIN'.

❑ FICHA: 19

Nos define un nuevo fichero, el PERIODSD, cuya DSNAME será 'PREP.PERIODI1.DISKPRSD'.

- DISP. Decimos que el fichero se crea en ese paso (NEW) y que si el paso termina bien debe ser catalogado (CATLG) y si termina mal debe ser borrado (DELETE).
- UNIT=SYSCR. El fichero se creará en la unidad SYSCR.
- SPACE=(CYL,(45,10),RLSE). Indicamos el espacio que asignamos a ese nuevo fichero. El espacio se medirá en cilindros (CYL) y será, inicialmente, de 45, cogiéndose, en caso necesario, extensiones de 10 cilindros. Con RLSE indicamos que el espacio que no se haya utilizado en la creación del fichero deberá liberarse cuando se cierre el fichero.
- VOL=SER=338080. Le indicamos el volumen y el número de serie en donde queremos que se cree el fichero (338080).
- DCB=(RECFM=FB,LRECL=400,BLKSIZE=2400). Los registros serán de 400 octetos de longitud, fija y bloqueada a 22400.

- FICHA: 20

  El fichero se crea como ficticio (DUMMY).

  El espacio se mide en pistas (TRK).

  Los registros son de longitud variable y bloqueados (VB).

  (En el parámetro LRECL, para el caso de longitud variable, se da la mayor longitud de los registros).

- FICHA: 21

  El fichero se crea en ese paso y se le dice que, si el paso termina bien, lo guarde para pasos posteriores.

  El fichero se creará en el dispositivo HTAPE (cinta).

  LABEL=(1,SL,EXPDT=9800). Aquí le explicamos que el fichero será el primero dentro de la cinta, que tiene etiqueta estándar IBM y la expiración del mismo.

- FICHA: 22

  El fichero se crea y, si el paso termina normalmente, se guarda para pasos posteriores (PASS) conservando toda la información sobre el mismo.

- FICHA: 23

  Aquí se define un fichero temporal (la DSN comienza por '&&') que será utilizado en el PASO3 (ficha 37) de forma exclusiva, borrándose al final de dicho paso.

- FICHA: 24-38-39

  Nos definen ficheros de impresora.

  - SYSOUT=(X,,STD). Nos indica que la clase a la que queda asignada esa impresora es la 'X' y que el tipo de formulario es el estándar (STD).
  - FCB=3262. Especifica que el programa de control de carro es el 3262.
  - COPIES=2. Indica el número de copias que deseamos (por defecto asume 1).
  - DCB=(LRECL=133,BLKSIZE=133,RECFM=FA). Se indica que es un fichero de impresora con una longitud de registro de 133 y un blocaje también de 133.

- FREE=CLOSE. El fichero se libera cuando se cierra (por defecto es END, que libera al terminar el paso).

❑ FICHA: 25

Definimos un fichero (que ya existe) y le decimos que los datos relativos al mismo van en las fichas siguientes.

Los datos van en las fichas 26 y 27 y la ficha 28 indica el final de los mismos.

❑ FICHA: 34

Definimos un fichero que ya existe y que será tratado de forma compartida. Si el paso termina normalmente el fichero se borrará y si no se guardará.

❑ FICHA: 41

Indica la finalización de un JOB. Cualquier ficha posterior a esta no sería tratada.

11.2 Ejemplo DL/I

```
 1   //XXXXXXXX JOB MSGLEVEL=(1,1),MSGCLASS=W,CLASS=A,NOTIFY=XXXXXXX
 2   //JOBLIB  DD DSN=CALP.LINKBAT,DISP=SHR
 3   //PASOX    EXEC   PGM=DFHDRP,PARM=('PGM=PPRG530,PSB=PSBCPGG,',
     //         'SSA=4096,CICS=DBDCCICS,LANG=C,CMPAT=Y')
 4   //PASOX    EXEC DLIBATCH,MBR=PREBC032,PSB=PSBGPRE,
     //         DBRC=N,IRLM=N,REGION=4096K,TIME=1440
 5   //STEPLIB  DD DSN=CALP.LINKBAT,DISP=SHR
     //         DD DSN=IMS130.RESLIB,DISP=SHR
 6   //DFHLIB   DD DSN=CICS17.LOADLIB1,DISP=SHR
 7   //DFSVSAMP DD DSN=CALE.VSAM.BUFFER(BREOCON),DISP=SHR
     //         DD DSN=CALE.VSAM.BUFFER(BREOPER),DISP=SHR
 8   //SYSPRINT DD SYSOUT=*
 9   //SYSOUT   DD SYSOUT=*
10   //SYSUDUMP DD SYSOUT=*
11   //SYSDBOUT DD SYSOUT=*
12   //FICCON01 DD *
     301188198830 DE NOVIEMBRE DE 1.988
13   //CACMORSD DD  DSN=PREE.MOROSOS.MORBC240(0),DISP=SHR,UNIT=HTAPE
14   //DBPRESTD DD DSN=PREE.PRESTAMO.PREBC032,DISP=(NEW,PASS),UNIT=HTAPE,
     //            LABEL=(,SL,EXPDT=98100),
     //            DCB=(RECFM=FB,LRECL=418,BLKSIZE=29260)
15   //ADJUCASD DD DSN=PREE.ADJUCASD.DISK00SD,DISP=(NEW,CATLG,DELETE),
     //            DCB=(RECFM=FB,LRECL=30,BLKSIZE=3000),UNIT=SYSCR,
     //            SPACE=(CYL,(20,10),RLSE,,ROUND)
16   //IMPRES01 DD SYSOUT=(A,,1003),FCB=3262,
     //            DCB=(LRECL=00133,BLKSIZE=00133,RECFM=FA)
17   //
```

- FICHA: 3-4

  Identificación del trabajo a ejecutar.

  La ficha 3 nos indica que la ejecución se realizará bajo CICS y la 4 que esta será vía BATCH.

  - **DFHDRP** Programa que ejecutara el trabajo bajo CICS.
  - **PARM** Parámetros que identifican ese trabajo:
    - **PGM** Programa a ejecutar.
    - **PSB** Proporciona la estructura de datos de la aplicación (nos indica a qué base/s se accede y si el acceso es para lectura sólo o lectura y modificación).
    - **SSA** Es el argumento de búsqueda.
    - **CICS** Nombre del CICS bajo el que se ejecutará el proceso.
    - **LANG** Lenguaje del programa especificado en PGM (C=Cobol, A=Assembler, etc.).
    - **CMPAT** Hace referencia a las PCB (definición de las estructuras lógicas) en función de la PSB que se utilice (que sea de lectura o de modificación), reservando o no la primera para la realización de checkpoint (con independencia de que estos se realicen o no). Así, por ejemplo, con CMPAT=Y se reserva la primera PCB (PCB(1)) para la realización de checkpoint, con lo que los accesos a la base se deberán realizar con otra PCB (PCB(2)). Se deberá poner siempre y cuando la PSB sea de actualización de la base.
  - **DLIBATCH** Programa que ejecutará el trabajo en BATCH.
  - **MBR** Programa a ejecutar.
  - **PSB** Nombre de la PSB a utilizar.
  - **DBRC** Indicará si se va a utilizar o no la base de datos de control del recovery (data base recovery control) durante la ejecución del trabajo. (N=no se va a utilizar; Y=sí se va a utilizar).
  - **IRLM** Igual que el anterior pero relativo al IRLM (resource lock manager).
  - **REGION** Tamaño de memoria a utilizar.
  - **TIME** Tiempo máximo que el JOB puede utilizar la CPU.

11.3 DLITEST - PRINT BASE PREST – TEST

```
//BRB0902P JOB MSGLEVEL=(1,1),MSGCLASS=X,CLASS=G,NOTIFY=BRB0902
//*
//* ************************************************************
//* *       DLITEST - PRINT BASE PREST - TEST                  *
//* *       ================================                   *
//* ************************************************************
//*
//PASO1     EXEC DLIBATPR,MBR=DFSDDLT0,PSB=PSBGPRE,
//          DBRC=N,IRLM=N,REGION=4096K
//STEPLIB  DD DSN=CALP.LINKBAT,DISP=SHR
//         DD DSN=IMS130.RESLIB,DISP=SHR
//DFHLIB   DD DSN=CICS17.LOADLIB1,DISP=SHR
//DFSVSAMP DD DSN=CALP.VSAM.BUFFER(BREOPRE),DISP=SHR
//BASEPR1  DD DSN=DLIP.ESDSPRES.V337508,DISP=SHR
//IDEXPR1  DD DSN=DLIP.KSDSIPPR.V337508,DISP=SHR
//SIDXPMO  DD DSN=DLIP.KSDSISPR.V337508,DISP=SHR
//SYSDBOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//IEFRDER  DD DUMMY
//PRINTDD  DD SYSOUT=*
//SYSIN    DD *
S 1 1 1 1 1
U ************************************
U *   DL/I TEST  -  BD. DE PRESTAMOS   *
U ************************************
L        GU    PRESTAMO (CLAPRES  => 5214100000340230060)
EH8    OK
L   0050 GNP   HISTOPRE
/*
//*
//*    F I N     D E L     P R O C E S O
```

11.4 Prerreorganización bd de préstamos

```
//XXXXXX1  JOB  CLASS=G,MSGCLASS=X,NOTIFY=XXXXXX,MSGLEVEL=(1,1)
//*
//************************************
//* PRERREORGANIZACION BD DE PRESTAMOS *
//************************************
//*
//PASO010  EXEC PGM=DFSRRC00,PARM='ULU,DFSURPR0'
//STEPLIB  DD DSN=IMS130.RESLIB,DISP=SHR
//         DD DSN=CALE.LINKBAT,DISP=SHR
//DFSRESLB DD DSN=IMS130.RESLIB,DISP=SHR
//IMS      DD DSN=CALP.IMS130.PSBLIB.POP,DISP=SHR
//         DD DSN=CALP.IMS130.DBDLIB.POP,DISP=SHR
//SYSPRINT DD SYSOUT=*
//DFSURCDS DD DSN=PREP.PRESTAMO.PREREO,DISP=(NEW,CATLG,DELETE),
//            DCB=(BLKSIZE=1600),SPACE=(CYL,1),UNIT=SYSCR
//SYSIN    DD *
DBR=BASPRES
```

```
/*
```

11.5 Volcado a disco de la base de préstamos secuencial

```
//*
//*    ****************************
//***  VUELCA A DISCO LA BASE DE PRESTAMOS SECUENCIAL DE PRUEBAS *****
//*    ****************************
//*
//PASO040  EXEC DLIBAPOP,MBR=PREBC033,PSB=PSBLPRE,
//         DBRC=N,IRLM=N,REGION=4096K,TIME=1440
//SYSPRINT DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//STEPLIB  DD DSN=CALP.LINKBAT,DISP=SHR
//         DD DSN=IMS130.RESLIB,DISP=SHR
//DBPRESTD DD DSN=PREP.BASEPRES.DISKPRSD,DISP=SHR
//IMPRES01 DD SYSOUT=*
//DFSVSAMP DD DSN=CALP.VSAM.BUFFER(BREOPRE),DISP=SHR
//DFSURWF1 DD DSN=PREP.TRABAJO.REL1,UNIT=SYSCR,
//         DISP=(,CATLG,DELETE),SPACE=(CYL,90),
//         DCB=(RECFM=VB,LRECL=900,BLKSIZE=1800)
//SYSDBOUT DD SYSOUT=*
//BASEPR1  DD DSN=DLI.ESDSPRES.V338013.PRO,DISP=SHR
//IDEXPR1  DD DSN=DLI.KSDSIPPR.V338013.PRO,DISP=SHR
//*
```

11.6 Prefix resolution bd préstamos

```
//*
//**********************************
//* PREFIX RESOLUTION BD PRESTAMOS  *
//**********************************
//*
//PASO050  EXEC PGM=DFSURG10,REGION=100K
//STEPLIB  DD DSN=IMS130.RESLIB,DISP=SHR
//         DD DSN=CALE.LINKBAT,DISP=SHR
//SYSOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSDBOUT DD SYSOUT=*
//SORTWK01 DD UNIT=SYSCR,SPACE=(CYL,30)
//SORTWK02 DD UNIT=SYSCR,SPACE=(CYL,30)
//SORTWK03 DD UNIT=SYSCR,SPACE=(CYL,30)
//SORTIN   DD DSN=PREP.TRABAJO.REL1,DISP=SHR
//DFSURWF2 DD DSN=INTERM.WORK.FILE1,UNIT=SYSCR,
//            SPACE=(CYL,30),DISP=(NEW,DELETE),
//            DCB=(LRECL=900,BLKSIZE=1800,RECFM=VB)
//DFSURWF3 DD DSN=OUT.WORK.DSET1,UNIT=SYSCR,
//            SPACE=(CYL,30),DISP=(NEW,DELETE),
//            DCB=(LRECL=900,BLKSIZE=1800,RECFM=VB)
//DFSURCDS DD DSN=PREP.PRESTAMO.PREREO,DISP=SHR
//DFSURIDX DD DSN=PREP.SECOND.INDEX1,UNIT=SYSCR,
```

```
//             SPACE=(CYL,30),DISP=(NEW,CATLG,DELETE),
//             DCB=(LRECL=900,BLKSIZE=1800,RECFM=VB)
//*
```

11.7 Unload índice secundario (hisam unload)

```
//*
//*****************************************
//* UNLOAD INDICE SECUNDARIO (HISAM UNLOAD) *
//*****************************************
//*
//PASO060  EXEC PGM=DFSRRC00,REGION=4096K,
//         PARM='ULU,DFSURUL0'
//STEPLIB  DD DSN=IMS130.RESLIB,DISP=SHR
//         DD DSN=CALE.LINKBAT,DISP=SHR
//DFSRESLB DD DSN=IMS130.RESLIB,DISP=SHR
//IMS      DD DSN=CALP.IMS130.PSBLIB.POP,DISP=SHR
//         DD DSN=CALP.IMS130.DBDLIB.POP,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SIDXPMO  DD DSN=DLI.KSDSISPR.V338013.PRO,DISP=SHR
//INDICEW  DD DSN=PREP.SECOND.INDEX1,DISP=SHR
//SALIDA   DD DSN=PREP.INDICE.UNLD1,UNIT=SYSCR,
//            SPACE=(CYL,30),DISP=(NEW,CATLG,DELETE),
//            DCB=(LRECL=900,BLKSIZE=1800,RECFM=VB)
//DFSVSAMP DD DSN=CALE.VSAM.BUFFER(BREOCON),DISP=SHR
//SYSIN    DD *
X1MSIDXPMO  SIDXPMO  SALIDA            INDICEW
/*
```

11.8 Reload índice secundario (hisam reload)

```
//*****************************************
//* RELOAD INDICE SECUNDARIO (HISAM RELOAD)   *
//*****************************************
//*
//PASO070  EXEC PGM=DFSRRC00,REGION=4096K,
//         PARM='ULU,DFSURRL0'
//STEPLIB  DD DSN=IMS130.RESLIB,DISP=SHR
//         DD DSN=CALE.LINKBAT,DISP=SHR
//DFSRESLB DD DSN=IMS130.RESLIB,DISP=SHR
//IMS      DD DSN=CALP.IMS130.PSBLIB.POP,DISP=SHR
//         DD DSN=CALP.IMS130.DBDLIB.POP,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SIDXPMO  DD DSN=DLI.KSDSISPR.V338013.PRO,DISP=SHR
//DFSUIN01 DD DSN=PREP.INDICE.UNLD1,DISP=SHR
//DFSVSAMP DD DSN=CALE.VSAM.BUFFER(BREOCON),DISP=SHR
//SYSIN    DD DUMMY
/*
```

11.9 Prerreorganización bd de cuentas

```
//XXXXXX1  JOB  CLASS=G,MSGCLASS=X,NOTIFY=XXXXXX,MSGLEVEL=(1,1)
//*
//*=================================
//* PRERREORGANIZACION BD DE CUENTAS
//*=================================
//*
//PREREOR  EXEC PGM=DFSRRC00,PARM='ULU,DFSURPR0'
//STEPLIB  DD DSN=IMS13.RESLIB,DISP=SHR
//         DD DSN=CALP.LINKBAT,DISP=SHR
//DFSRESLB DD DSN=IMS13.RESLIB,DISP=SHR
//IMS      DD DSN=CALP.IMS13.PSBLIB,DISP=SHR
//         DD DSN=CALP.IMS13.DBDLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//DFSURCDS DD DSN=DLIP.CUENTAS.PREREO,UNIT=SYSCR,DISP=(,CATLG,DELETE),
//            DCB=(BLKSIZE=1600),SPACE=(CYL,(3,1),RLSE)
//SYSIN    DD *
DBR=BASCONE
/*
```

11.10 Prefix resolution base cuentas

```
//*
//*=========================================
//* PREFIX RESOLUTION BASE CUENTAS DE PRUEBAS
//*=========================================
//*
//PREFIXR  EXEC PGM=DFSURG10,REGION=100K
//STEPLIB  DD DSN=IMS13.RESLIB,DISP=SHR
//SYSOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SORTWK01 DD UNIT=SYSCR,SPACE=(CYL,10)
//SORTWK02 DD UNIT=SYSCR,SPACE=(CYL,10)
//SORTWK03 DD UNIT=SYSCR,SPACE=(CYL,10)
//SORTIN   DD DSN=DLIP.TRABAJO.REL,DISP=SHR
//DFSURWF2 DD DSN=DLIP.INTERM.WORK.FILE,UNIT=SYSCR,
//            SPACE=(CYL,10),DISP=(NEW,DELETE),
//            DCB=(LRECL=900,BLKSIZE=1800,RECFM=VB)
//DFSURWF3 DD DSN=DLIP.OUT.WORK.DSET,UNIT=SYSCR,
//            SPACE=(CYL,10),DISP=(NEW,DELETE),
//            DCB=(LRECL=900,BLKSIZE=1800,RECFM=VB)
//DFSURCDS DD DSN=DLIP.CUENTAS.PREREO,DISP=SHR
//DFSURIDX DD DSN=DLIP.SECOND.INDEX,UNIT=SYSCR,DISP=(,CATLG,DELETE),
//            SPACE=(CYL,10),
//            DCB=(LRECL=900,BLKSIZE=1800,RECFM=VB)
//*
```

11.11 Unload índice secundario (hisam unload)

```
//*
//*=======================================
//* UNLOAD INDICE SECUNDARIO (HISAM UNLOAD)
//*=======================================
//*
//UNLDIND  EXEC PGM=DFSRRC00,REGION=4096K,
//         PARM='ULU,DFSURUL0'
//STEPLIB  DD DSN=IMS13.RESLIB,DISP=SHR
//DFSRESLB DD DSN=IMS13.RESLIB,DISP=SHR
//IMS      DD DSN=CALP.IMS13.PSBLIB,DISP=SHR
//         DD DSN=CALP.IMS13.DBDLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SIDXTAR  DD DSN=DLIP.KSDSISCO.V337508,DISP=SHR
//INDICEW  DD DSN=DLIP.SECOND.INDEX,DISP=SHR
//SALIDA   DD DSN=DLIP.INDICE.UNLD,DISP=(,CATLG,DELETE),
//            UNIT=SYSCR,SPACE=(CYL,10),
//            DCB=(LRECL=900,BLKSIZE=1800,RECFM=VB)
//DFSVSAMP DD DSN=CALP.VSAM.BUFFER(BREOCON),DISP=SHR
//SYSIN    DD *
X1MSIDXTAR  SIDXTAR  SALIDA           INDICEW
/*
```

11.12 Reload índice secundario (hisam reload)

```
//*
//*******************************************
//* RELOAD INDICE SECUNDARIO (HISAM RELOAD)   *
//*******************************************
//*
//PASO070  EXEC PGM=DFSRRC00,REGION=4096K,
//         PARM='ULU,DFSURRL0'
//STEPLIB  DD DSN=IMS130.RESLIB,DISP=SHR
//         DD DSN=CALE.LINKBAT,DISP=SHR
//DFSRESLB DD DSN=IMS130.RESLIB,DISP=SHR
//IMS      DD DSN=CALP.IMS130.PSBLIB.POP,DISP=SHR
//         DD DSN=CALP.IMS130.DBDLIB.POP,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SIDXPMO  DD DSN=DLI.KSDSISPR.V338013.PRO,DISP=SHR
//DFSUIN01 DD DSN=PREP.INDICE.UNLD1,DISP=SHR
//DFSVSAMP DD DSN=CALE.VSAM.BUFFER(BREOCON),DISP=SHR
//SYSIN    DD DUMMY
 /*
```

11.13 Otros ejemplos

Ejemplos de JCL´S con diferentes formatos de selección:

- Siempre se especifica el número de caracteres que realmente se está ocupando en el fichero, es decir, tanto si ponemos BI como si ponemos PD tendremos que escribir la longitud empaquetada.

  Ej1: INCLUDE COND = (15,3,PD,NE,85)

  Ej2: INCLUDE COND = (15,3,BI,NE,X'00085C')

  En ambos casos se están seleccionando los registros que contienen '85' en la posición 15 del fichero.

  En el primer caso sólo hay que poner la cadena que queremos buscar sin comillas y sin formato hexadecimal, además, sólo hay que poner lo que queremos buscar directamente, sin necesidad de rellenar con ceros por la izquierda.

  En el segundo caso, es necesario especificarlo en hexadecimal con el contenido exacto.

- El parámetro BI también se puede poner para buscar una cadena alfanumérica.

  Ej: (45,3,BI,EQ,C'190').

  En este caso, se busca el valor 190 en la posición 45, y aunque se pudiera poner CH es más eficiente poner BI.

- SORT OUTREC

```
*************************************************************
****  El fichero de entrada tiene una longitud de 70. Le añadimos un campo
****  empaquetado de 3 posiciones con el valor '1', delante del registro.
*************************************************************
//PASO010  EXEC PGM=SORT
//SORTIN   DD  DSN=AHOP.CZRLARLI.REPRO,DISP=SHR
//SORTOUT  DD  DSN=AHOP.CZRLARLI.OUTREC,DISP=(,CATLG,DELETE),
//         UNIT=SYSDA,SPACE=(CYL,(3,2),RLSE),LRECL=73,
//         RECFM=FB
//SYSOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SORTWK01 DD UNIT=SYSDA,SPACE=(CYL,(10,10))
//SORTWK02 DD UNIT=SYSDA,SPACE=(CYL,(10,10))
//SYSIN DD *
  SORT FIELDS=COPY
  OUTREC FIELDS=(X'00001C',1,70)
  END
/*
```