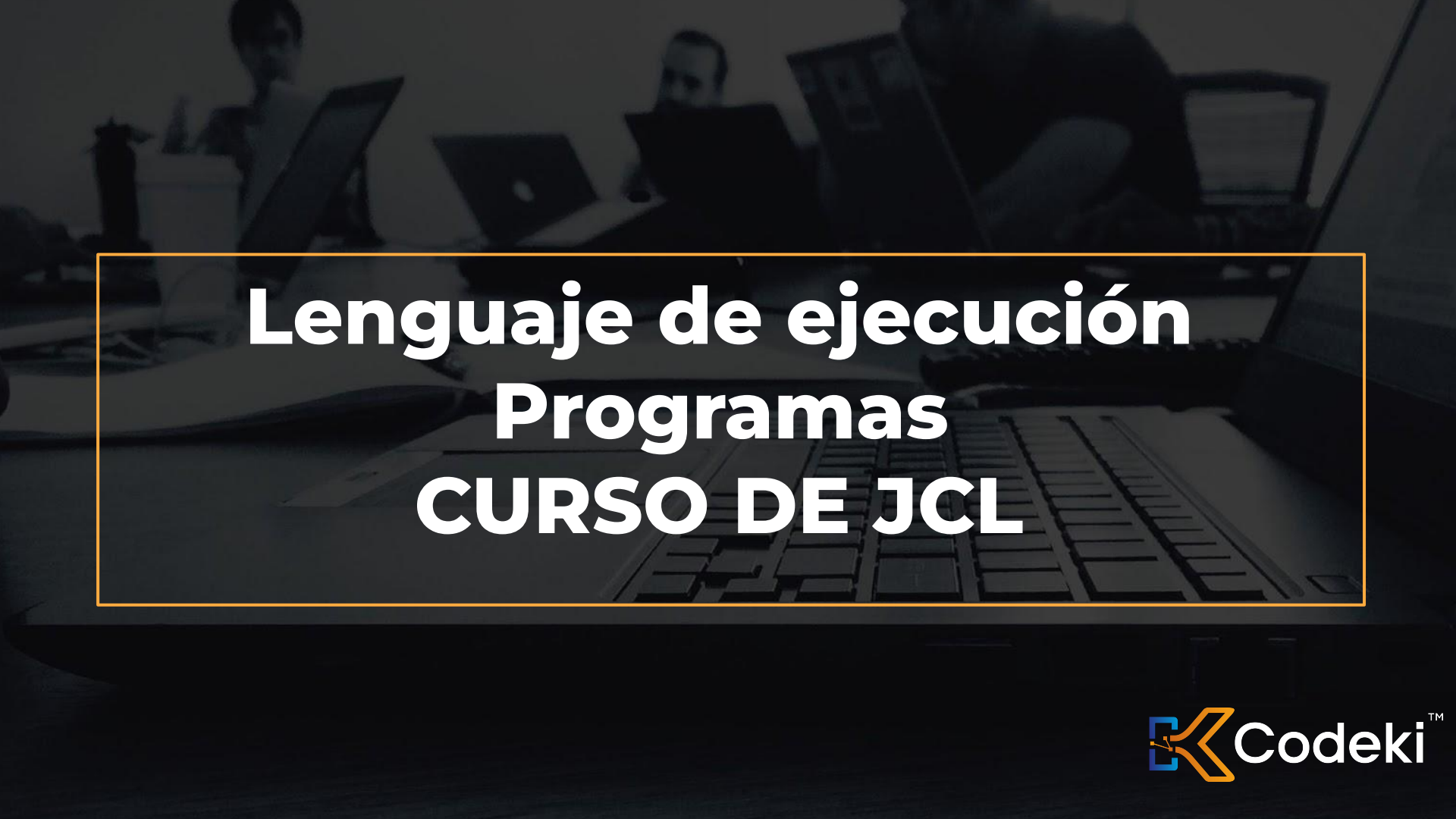
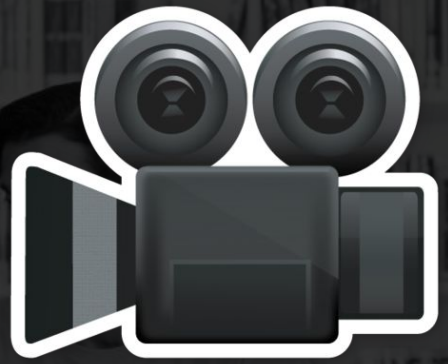
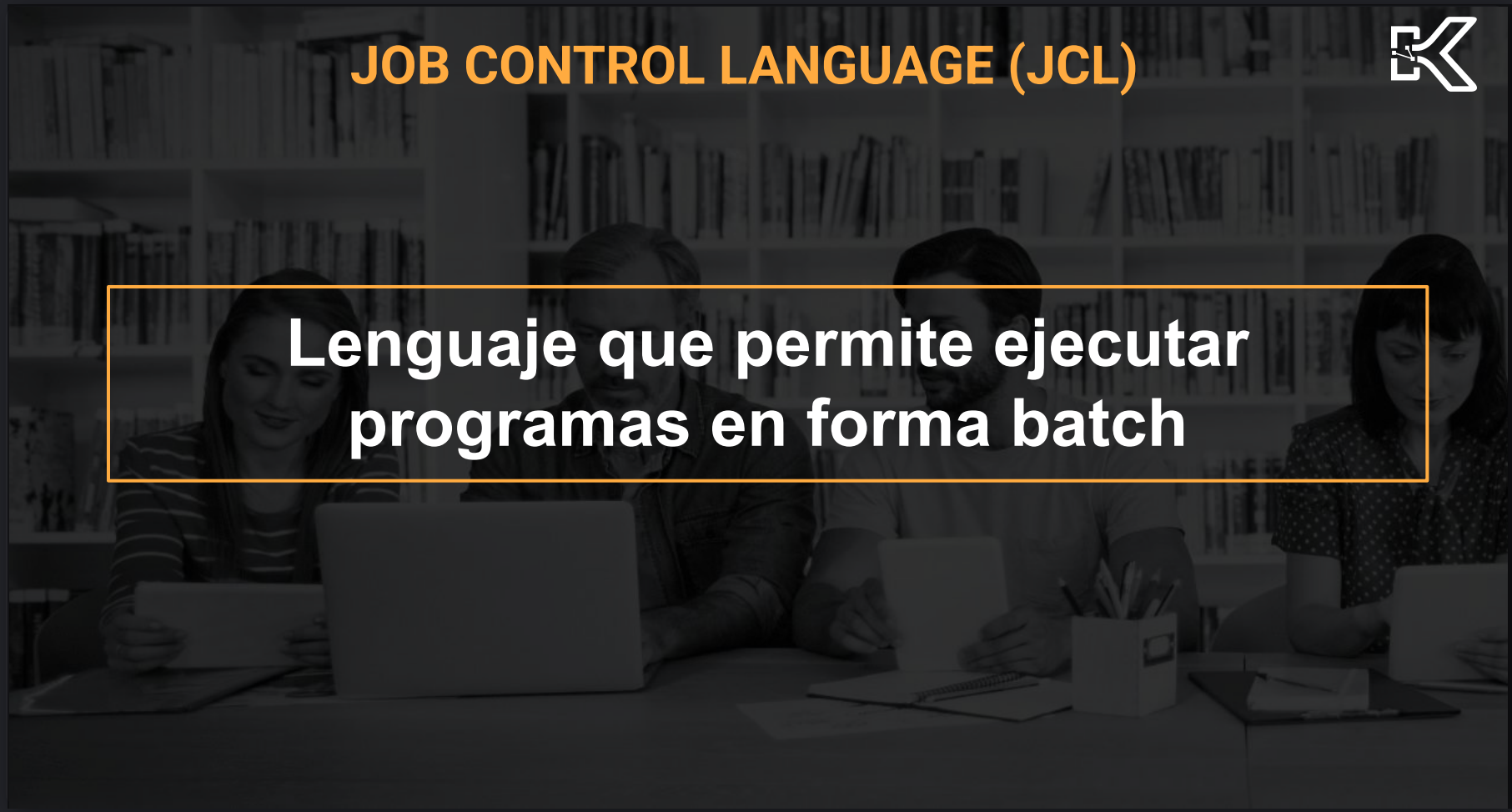

Lenguaje de ejecución
Programas
CURSO DE JCL
Codeki™

Recuerda poner a grabar la clase.



JOB CONTROL LANGUAGE (JCL)



Lenguaje que permite ejecutar programas en forma batch

JCL

Job Control Language

REGLAS DE SINTAXIS PARA FORMATOS JCL

```
//JOBPRU
//PASO01
//ARCHIVO
```

//Nombre Operación Operando Comentario

- Utilizado para identificar el JOB, cada uno de sus pasos y Datasets
- Debe contener hasta 8 caracteres de longitud
- Debe comenzar a ser codificada a partir de la columna 3

REGLAS DE SINTAXIS PARA FORMATOS JCL

//JOBPRU
//JOBLIB
//PASO01
//ARCHIVO

JOB
DD
EXEC
DD

//Nombre Operación Operando Comentario

JOB CONTROL LANGUAGE (JCL)

```
//JOBPRU
//JOBLIB
//PASO01
//ARCHIVO
```

```
JOB
DD
EXEC
DD
```

```
(5556,DX),'PEDRO',CLASS=A,MSGCLASS=A
 DESARR.BIBLIO.FUENTES,DISP=SHR
PGM=PROGPRU,REGION=2M,TIME=5
DSN=SECUEN.ARCHIVO.PRUEBA,UNIT=SYSDA,
 DCB=(RECFM=FB,LRECL=1164),
DISP=(NEW,KEEP,DELETE),
 SPACE=(TRK(10,5),RLSE)
```

```
//Nombre Operación Operando Comentario
```

REGLAS DE SINTAXIS PARA FORMATOS JCL

- Son los parámetros de las sentencias. Las sentencias JOBLIB y ARCHIVO son opcionales
- El primer parámetro debe separarse de la sentencia mediante un espacio en blanco
- Entre sí, deben separarse por comas
- No debe existir ningún espacio en blanco entre los parámetros

REGLAS DE SINTAXIS PARA FORMATOS JCL

```
//*Este es un job de prueba
```

```
//JOBPRU
//JOBLIB
//PASO01
//ARCHIVO
```

```
JOB
DD
EXEC
DD
```

```
(5556,DX),'PEDRO',CLASS=A,MSGCLASS=A
 DESARR.BIBLIO.FUENTES,DISP=SHR
PGM=PROGPRU,REGION=2M,TIME=5
DSN=SECUEN.ARCHIVO.PRUEBA,UNIT=SYSDA,
 DCB=(RECFM=FB,LRECL=1164),
DISP=(NEW,KEEP,DELETE),
 SPACE=(TRK(10,5),RLSE)
```

```
//Nombre Operación Operando Comentario
```

REGLAS DE SINTAXIS PARA FORMATOS JCL

- Las columnas disponibles para el comentario van desde la 3 a la 80.
- Puede ser ubicado en cualquier parte del job, incluso entre medio de líneas que contengan parámetros de cualquier sentencia.

SENTENCIAS PRINCIPALES

SENTENCIA NULL

- Se utiliza para marcar el fin del trabajo.
- Se la codifica con // en las columnas 1 y 2.
- Si en la columna 3 se le agrega un ‘*’, la sentencia pasa a ser un comentario.

SENTENCIAS PRINCIPALES

- **SENTENCIA JOBPRU**
  Se la utiliza para identificar un trabajo.

- **SENTENCIA PASO01**
  Es el nombre del Paso de Trabajo.

- **SENTENCIA EXEC**
  Se la utiliza para permitir ejecutar un programa

- **SENTENCIA DD (Definición de Datos)**
  Definen los archivos que se van a utilizar en los programas que se van a ejecutar. Esta es la sentencia más extensa del JCL. A través de estas sentencias se puede definir:

  - El nombre del archivo de datos a usar
  - El tipo de unidad de E/S para el archivo
  - El volumen en el que reside el archivo de datos
  - Si el archivo de datos ya existe o se está creando
  - Cantidad de espacio de DASD que se dará a un archivo de datos

La secuencia de sentencias DD en un paso no es significativa.

SENTENCIAS ESPECIALES

SENTENCIA JOBLIB

Identifica la biblioteca propia del job en la cual se aloja el programa especificado en el parámetro PGM de la sentencia EXEC.

JOBLIB DD Nombre_Biblioteca, DISP=SHR

NOTA: Parámetro DISP se detalla más adelante en los Parámetros de Sentencia DD

SENTENCIA STEPLIB

Identifica la biblioteca propia del paso en la cual se aloja el programa especificado en el parámetro PGM de la sentencia EXEC de dicho paso.

STEPLIB DD Nombre_Biblioteca, DISP=SHR

SENTENCIAS DD ESPECIALES

```
//JOBPRU   JOB  (5556,DX),'PEDRO',CLASS=A,MSGCLASS=A,
//                NOTIFY=USUARIOA
//JOBLIB   DD   PRODUCC.BIBLIO.OBJETOS,DISP=SHR
//*-------------------- Paso 1 ----------------------*
//PASO01   EXEC PGM=PROG1,REGION=2M,TIME=5
//STEPLIB  DD   DESARR.BIBLIO.OBJETOS,DISP=SHR
//*-------------------- Paso 2 ----------------------*
//PASO02   EXEC PGM=PROG2,REGION=6M
//
```

SENTENCIAS DD ESPECIALES

En el ejemplo anterior; el programa del Paso01 se encuentra en desarrollo. Por default se define una JOBLIB apuntando a los programas de producción. En el Paso02 al no haber especificada una STEPLIB, el sistema asume que el programa se encuentra en ambiente productivo.

Si se incluye una STEPLIB para un paso y una JOBLIB para todo el Job, el sistema busca primero el programa especificado para el paso en la STEPLIB. De no encontrarlo, lo busca en las definidas en la JOBLIB y de no encontrarlo lo busca en la biblioteca del sistema llamada SYS1.LINKLIB.

SENTENCIAS DD ESPECIALES

SENTENCIA SYSIN

Se la utiliza para pasar parámetros de entrada al programa especificado en el parámetro PGM de la sentencia EXEC.

SYSIN DD * Parámetros_entrada al Programa

NOTA: en el programa COBOL se deberá utilizar la sentencia:

ACCEPT Nombre_Variable FROM SYSIN

La variable deberá ser definida con el correspondiente tipo de dato referenciado en el parámetro de entrada.

SENTENCIAS DD ESPECIALES

```
//JOBPRU   JOB  (5556,DX),'PEDRO',CLASS=A,MSGCLASS=A,
//            NOTIFY=USUARIOA
//JOBLIB   DD   PRODUCC.BIBLIO.OBJETOS,DISP=SHR
//PASO01   EXEC PGM=PROGPRU,REGION=2M,TIME=5
//ARCHIN   DD   DSN=SECUEN.ARCHIVO.ENTRADA,DISP=SHR
//ARCHOUT  DD   DSN=SECUEN.ARCHIVO.PRUEBA,UNIT=SYSDA,
//            DCB=(RECFM=FB,LRECL=1164),
//            DISP=(NEW,KEEP,DELETE),
//            SPACE=(TRK,,(10,5),RLSE)
//SYSIN    DD   *
19990225DX
//
```

SENTENCIAS DD ESPECIALES

En el ejemplo anterior; el programa PROGPRU recibe como parámetro una fecha y nombre de Sistema.
Se deberá definir la variable COBOL:

```
01 WS-SYSIN.
   03 WS-SYSIN-FECHA      PIC 9(08).
   03 WS-SYSIN-SISTEMA    PIC X(02).
   03 WS-SYSIN-RESTO-LINEA PIC X(70).
```

Y en el programa, en la instancia que corresponda, se deberá codificar la sentencia:

```
ACCEPT WS-SYSIN FROM SYSIN.
```

Con lo cual las variables WS-SYSIN-FECHA y WS-SYSIN-SISTEMA contendrán los correspondientes valores detallados en el job.

SENTENCIAS DD ESPECIALES

CONCATENACIÓN DE BIBLIOTECAS (Para JOBLIB y para STEPLIB)

```
//JOBPRU   JOB  (5556,DX),'PEDRO',CLASS=A,MSGCLASS=A,
//           NOTIFY=USUARIOA
//JOBLIB        DD   DSN=PRODUCC.BIBLIO.OBJETOS,DISP=SHR
//                DD   DSN=MONITOR.BIBLIO.OBJETOS,DISP=SHR
//                DD   DSN=DESARR.BIBLIO.OBJETOS,DISP=SHR
//PASO01   EXEC PGM=PROG1,REGION=2M,TIME=5
//ARCHOUT  DD   DSN=SECUEN.ARCHIVO.PRUEBA,UNIT=SYSDA,
//           DCB=(RECFM=FB,LRECL=1164),
//           DISP=(NEW,KEEP,DELETE),
//           SPACE=(TRK,(10,5),RLSE)
//
```

SENTENCIAS DD ESPECIALES

En el ejemplo anterior; cuando se ejecuta el programa PROG1, el sistema busca el miembro primero en la biblioteca PRODUCC.BIBLIO.FUENTES. De no encontrarlo, lo busca en MONITOR.BIBLIO.FUENTES. De no encontrarlo, lo busca en DESARR.BIBLIO.FUENTES. De no encontrarlo, lo busca en SYS1.LINKLIB (biblioteca del sistema). De no encontrarlo, cancela el job.

SENTENCIA JOB

Las Sentencias

JOB

EXEC

DD

Parámetros Posicionales:

Son aquellos que se identifican por su posición dentro de la sentencia.

De codificarse, deben ser escritos antes que todos los demás parámetros.

Parámetros Opcionales o no Posicionales:

Pueden codificarse en cualquier orden y se los identifican por el formato

PARAMETRO=Información_variable

SENTENCIA JOB - PARÁMETROS POSICIONALES

INFORMACIÓN DE LA CUENTA

Depende de cada instalación. Se puede definir un **número de cuenta** y sub parámetros, como por ejemplo, grupo de trabajo, ubicación física, etc. Si se codifican subparámetros el formato debe ser el siguiente:
(Nro Cuenta, Grupo de Trabajo, Ubicación)

```
//JOBPRU   JOB  (5556,DX,FLORIDA),'PEDRO',CLASS=A
```

NOMBRE DEL PROGRAMADOR

Generalmente especifica el nombre de usuario o, como en algunas instalaciones, una breve descripción del proceso.

```
//JOBPRU   JOB  ,'PEDRO',CLASS=A
```

SENTENCIA JOB - PARÁMETROS POSICIONALES

CLASS

Se le asigna a un trabajo una Clase en la Cola de Entrada y Ejecución en JES2.

CLASS=Clase

Clase: A..Z o 0..9

```
//JOBPRU   JOB  ,'PEDRO',CLASS=A
```

El ejemplo indica que el job será alojado en la clase A durante su espera para ser procesado y será ejecutado en la misma clase una vez seleccionado por el Task Dispatcher del JES2.

SENTENCIA JOB - PARÁMETROS POSICIONALES

MSGLEVEL

Se utiliza para controlar la cantidad y el tipo de información de salida generada por el JES y el JCL

MSGLEVEL=(Nivel Sentencias,Mensajes)

Nivel Sentencias	Significado
0	Imprime solo información de control de la sentencia JOB
1	Imprime información de control de todas las sentencias del JCL (incluyendo los procedimientos de llamados)
2	Idem anterior pero sin incluir procedimientos llamados

Mensajes	Significado
0	Sólo mensajes referentes a la actividad del JCL
1	Mensajes referentes a la actividad del JES2 y el JCL

SENTENCIA JOB - PARÁMETROS POSICIONALES

```
//JOBPRU   JOB  ,'PEDRO',CLASS=A,MSGLEVEL=(1,1)
```

El ejemplo indica que se deben imprimir los mensajes JCL y JES2 generados durante la ejecución de todas las sentencias del job.

- Sentencias JCL numeradas
- Mensajes de error
- Mensajes de asignación de recursos
- Mensajes de terminación

Existen mensajes de asignación y terminación para cada uno de los pasos de trabajo. Son el resultado de las operaciones efectuadas por el initiator que ejecuta cada paso del job.

SENTENCIA JOB - PARÁMETROS POSICIONALES

MSGCLASS

Determina la clase de salida para el JES2 (con o sin impresora asociada) para los mensajes de asignación y terminación de cada paso del job.

MGSCLASS=Clase de Salida

Clase de Salida: A..Z o 0..9

```
//JOBPRU   JOB  ,'PEDRO',CLASS=A,MSGLEVEL=(1,1),MSGCLASS=X
```

El ejemplo indica que se deben imprimir los mensajes JCL y JES2 generados durante la ejecución de todas las sentencias del job y los mismos serán alojados en la Clase de Output X.

SENTENCIA JOB - PARÁMETROS POSICIONALES

COND

Especifica las condiciones bajo las cuales se continúa o detiene la ejecución del job. Cuando un programa finaliza su ejecución deja como legado un código de retorno el cual se compara con el código definido en el parámetro COND.

COND=(Código,operador)

GT	Mayor
GE	Mayor o Igual
EQ	Igual
NE	Distinto
LE	Menor o Igual
LT	Menor

Código: 0 (ok) .. 4095

El código de retorno del programa se compara con el valor especificado en COND.
Mientras sea verdadera se ejecuta el JOB

SENTENCIA JOB - PARÁMETROS POSICIONALES

```
//JOBPRU   JOB  ,'PEDRO',CLASS=A,MSGLEVEL=(1,1),
//              ,MSGCLASS=X,COND=(8,LT)
```

El código especificado en el parámetro COND significa:
Mientras el Código de Retorno del Paso que se ejecutó es menor a 8, continúa la ejecución del paso siguiente.

```
//JOBPRU   JOB  ,'PEDRO',CLASS=A,MSGLEVEL=(1,1),
//              ,MSGCLASS=X,COND=(4,GE)
```

El código especificado en el parámetro COND significa:
Mientras el Código de Retorno del Paso que se ejecutó es mayor o igual a 4, continúa la ejecución del paso siguiente.

SENTENCIA JOB - PARÁMETROS POSICIONALES

COND=(EVEN)

Permite que se ejecute el paso aún cuando un paso previo haya terminado en forma anormal.

COND=(ONLY)

Hace que el paso se ejecute solamente si algún paso previo no terminó anormalmente.

Cada paso que sigue a uno que termina en forma anormal será salteado a menos que en el EXEC del paso siguiente se especifique el parámetro COND=(EVEN).

SENTENCIA JOB - PARÁMETROS POSICIONALES

```
//JOBPRU   JOB  ,'PEDRO',CLASS=A,MSGLEVEL=(1,1),
//              ,MSGCLASS=X,COND=((8,LT),(4,EQ))
```

Es posible especificar pares ordenados de condiciones. La ‘,’ (coma) entre pares ordenados equivale a un OR.

Se pueden establecer hasta **8** condiciones de finalización en un Job

SENTENCIA JOB - PARÁMETROS POSICIONALES

NOTIFY

Indica el Usuario que recibirá el mensaje de fin de corrida (exitosa o no).

NOTIFY=Nombre_Usuario

Nombre_Usuario: nombre de usuario existente en la instalación

```
//JOBPRU   JOB  'PEDRO',CLASS=A,MSGLEVEL=(1,1),
//                MSGCLASS=X,COND=(8,LT),
//                NOTIFY=USUARIOA
```

Si en el momento de la finalización del job el usuario se encuentra logoneado, será notificado al instante.

De no estarlo, el sistema guardará dicho mensaje el cual le será presentado al usuario al terminar su proceso de logon.

SENTENCIA JOB - PARÁMETROS POSICIONALES

TIME

Especifica la cantidad máxima de tiempo que un trabajo puede utilizar la CPU. Este parámetro es preventivo de aquellos programas que entran en loop indefinidamente.

TIME=(minutos,segundos) **ejemplo TIME=(,3)**

Minutos : 1..1440 (NOLIMIT)

```
//JOBPRU   JOB  ,'PEDRO',CLASS=A,MSGLEVEL=(1,1),
//            MSGCLASS=X,COND=(8,LT),
//            NOTIFY=USUARIOA,TIME=(,35)
```

En el ejemplo el job terminará su ejecución si el tiempo insumido de CPU excede los 35 segundos. **No especificar los 0 minutos pues los resultados pueden ser impredecibles**

SENTENCIA JOB - PARÁMETROS POSICIONALES

```
//JOBPRU   JOB  ,'PEDRO',CLASS=A,MSGLEVEL=(1,1),
//              MSGCLASS=X,COND=(8,LT),
//              NOTIFY=USUARIOA,TIME=(,5)
```

En el ejemplo el job terminará su ejecución si el tiempo insumido de CPU excede los 5 segundos

```
//JOBPRU   JOB  ,'PEDRO',CLASS=A,MSGLEVEL=(1,1),
//              MSGCLASS=X,COND=(8,LT),
//              NOTIFY=USUARIOA
```

SENTENCIA JOB - PARÁMETROS POSICIONALES

REGIÓN

Especifica la cantidad de espacio de memoria que requerirá el trabajo.

REGION=Valor en Megabytes

Valor en Megabytes: 1..2047 (Max. 2GB direccionables, capacidad máxima de un address space, hablando de direcciones virtuales)

```
//JOBPRU   JOB  ,'PEDRO',CLASS=A,MSGLEVEL=(1,1),
//            MSGCLASS=X,COND=(8,LT),
//            NOTIFY=USUARIOA,REGION=0M
```

SENTENCIA JOB - PARÁMETROS POSICIONALES

TYPRUN

Utilizado en Jobs que tienen requerimientos especiales de procesamiento.

TYPRUN=HOLD
TYPRUN=SCAN

HOLD: el trabajo se retiene antes de su ejecución hasta que el operador lo libere
SCAN: chequea si hay errores de sintaxis en JCL en el job, sin ejecutarlo.

```
//JOBPRU   JOB  ,'PEDRO',CLASS=A,MSGLEVEL=(1,1),
//            MSGCLASS=X,COND=(8,LT),
//            NOTIFY=USUARIOA,REGION=6M,TYPRUN=SCAN
```

En el ejemplo el job NO se ejecutará. Al terminar el chequeo se podrá visualizar el resultado en la Cola de Salidas Holdeadas del JES2, en donde indicará los errores a invel sentencia y parámetro.

SENTENCIA EXEC - PARÁMETROS POSICIONALES

El primer parámetro debe ser alguno de los siguientes:

NOMBRE DE PROGRAMA

Especifica el programa que deberá ser ejecutado en el paso.

PGM=Nombre_Programa

```
//JOBLIB   DD   DESARR.BIBLIO.PROG,DISP=SHR
//DEFPROG  EXEC PGM=PROGRAMA
```

Nombre de programa: debe estar almacenado como miembro de una biblioteca, la cual (si es privada, no del sistema denominada SYS1.LINKLIB), deberá especificarse en la sentencia JOBLIB o STEPLIB (si es privada sólo para ese paso).

SENTENCIA EXEC - PARÁMETROS POSICIONALES

NOMBRE DE PROCEDIMIENTO

Invoca a un procedimiento alojado en una biblioteca de procedimientos

PROC=Nombre_Procedimiento

```
//JOBPRU   JOB  ,'PROC EN BIBLIO',CLASS=A,MSGCLASS=A,
//            NOTIFY=USUARIOA
//JOBLIB   DD   DESARR.BIBLIO.PROCS,DISP=SHR
//DEFPROC  EXEC PROC=PROCED
//ARCHOUT  DD   DSN=SECUEN.ARCHIVO.PRUEBA,UNIT=SYSDA,
//            DCB=(RECFM=FB,LRECL=1164),
//            DISP=(NEW,KEEP,DELETE),
//            SPACE=(TRK,(10,5),RLSE)
//
```

SENTENCIA EXEC - PARÁMETROS POSICIONALES

PARM

Su función es la de pasar parámetros (hasta un máximo de 100 caracteres) al programa en cuestión.

PARM='Subparámetro1,Subparámetro2,...'

Es posible pasar uno o más sub parámetros (separados por comas).

```
//JOBPRU   JOB  ,'JOB CON PARM',CLASS=A,MSGCLASS=A,
//            NOTIFY=USUARIOA
//JOBLIB   DD   DESARR.BIBLIO.PROCS,DISP=SHR
//PASO01   EXEC PGM=PROG1,REGION=6M,PARM='P1,1999,DX'
//ARCHOUT  DD   DSN=SECUEN.ARCHIVO.PRUEBA,UNIT=SYSDA,
//            DCB=(RECFM=FB,LRECL=1164),
//            DISP=(NEW,DELETE,DELETE),
//            SPACE=(TRK,(10,5),RLSE)
//
```

SENTENCIA EXEC - PARÁMETROS POSICIONALES

DUMMY

De especificarse, debe ser el primero de la lista de parámetros de la sentencia DD.

DD DUMMY

DUMMY representa a un archivo fantasma. La salida puede haber sido definida en programa pero en el job se lo asocia a un archivo que no tiene información ni es resultado de ningún proceso. Puede ser utilizado para archivos de entrada o salida.

```
//JOBPRU   JOB  ,'JOB CON DUMMY',CLASS=A,MSGCLASS=A,
//            NOTIFY=USUARIOA
//JOBLIB   DD   DSN=DESARR.BIBLIO.FUENTES,DISP=SHR
//PASO01   EXEC PGM=PROG1,REGION=2M
//ARCHIN   DD   DSN=SECUEN.ARCHIVO.PRUEBA,DISP=OLD
//ARCHOUT  DD   DUMMY
//
```

SENTENCIA EXEC - PARÁMETROS POSICIONALES

*

De especificarse, debe ser el único parámetro de la sentencia DD.

DD *

Para el ZOS el DD * equivale a entrada de datos, la cual finaliza al encontrar la próxima sentencia JCL.

```
//JOBPRU   JOB  (5556,DX),'PEDRO',CLASS=A,MSGCLASS=A,
//           NOTIFY=USUARIOA
//JOBLIB   DD   PRODUCC.BIBLIO.FUENTES,DISP=SHR
//PASO01   EXEC PGM=PROGPRU,REGION=2M,TIME=5
//ARCHIN   DD   DSN=SECUEN.ARCHIVO.ENTRADA,DISP=SHR
//ARCHOUT  DD   DSN=SECUEN.ARCHIVO.PRUEBA,DISP=SHR
//SYSIN    DD   *
REGISTRO1
REGISTRO2
//
```

SENTENCIA EXEC - PARÁMETROS POSICIONALES

En el ejemplo ANTERIOR, el programa COBOL deberá efectuar la ejecución de DOS sentencias ACCEPT ws-nombre-variable FROM SYSIN para obtener como parámetro REGISTRO1 y REGISTRO2.

SENTENCIA DSN - PARÁMETROS POSICIONALES

DSNAME (DSN)

Este es un parámetro de la sentencia DD que permite identificar el archivo de datos a utilizar (ya creado) o nominar (nuevo archivo) de salida que se creará.

DSN=Nombre_Archivo
&&Nombre_Archivo

Nombre_Archivo : nombre de archivo Secuencial, VSAM o Particionado.
&&Nombre_Archivo : nombre de un archivo temporario.

Archivos temporarios

Se los puede incluir en algún paso del trabajo. Son aquellos que nacen y mueren durante la ejecución del Job.

SENTENCIA DD - PARÁMETROS POSICIONALES

```
//JOBPRU   JOB  (5556,DX),'ARCH TEMP',CLASS=A,MSGCLASS=A,
//           NOTIFY=USUARIOA
//JOBLIB   DD   PRODUCC.BIBLIO.FUENTES,DISP=SHR
//*--------------------- Paso 1 -------------------------
//PASO01   EXEC PGM=PROG10,REGION=2M,TIME=5
//ARCHIN   DD   DSN=SECUEN.ARCHIVO.ENTRADA,DISP=SHR
//ARCHTEMP DD   DSN=&&TEMPORARIO,UNIT=SYSDA,
//           DCB=(RECFM=FB,LRECL=1164),
//           DISP=(NEW,KEEP,DELETE),
//           SPACE=(TRK,(10,5),RLSE)
//*--------------------- Paso 1 -------------------------
//PASO01   EXEC PGM=PROG20,REGION=6M
//ARCHIN   DD   DSN=&&TEMPORARIO,DISP=SHR
//ARCHOUT  DD   DSN=SECUEN.ARCHIVO.SALIDA,UNIT=3390,
//           DCB=(RECFM=FB,LRECL=1164),
//           DISP=(NEW,KEEP,DELETE),
//           SPACE=(TRK,(10,5),RLSE),VOL=SER=DIS729
//
```

SENTENCIA DISP - PARÁMETROS POSICIONALES

DISP

Este parámetro sirve para indicarle al ZOS si un archivo existe o será creado en el Job. Esta indicación se realiza antes de comenzar la ejecución del programa.

DISP=(estado archivo antes ejecutar, estado archivo después de ejecutar (terminación normal), estado archivo después de la ejecución (terminación anormal))

Estos subparámetros son posicionales. De no especificarse alguno, se debe respetar su lugar dentro del parámetro DISP.

Por ejemplo:

DISP=(,suparámetro2,subparámetro3)

SENTENCIA DISP - PARÁMETROS POSICIONALES

SUBPARAMETROS ANTES DE LA EJECUCIÓN

- **NEW:** se creará un archivo.
- **OLD:** archivo existente. Ningún otro programa lo usará hasta que nuestro job lo libere.
- **MOD:** archivo existente. Se agregarán registros al final del mismo. Ningún otro programa lo puede utilizar.
- **SHR:** archivo existente. Puede ser compartido por otros programas.

Si no se especifica el primer subparámetro del DISP en la sentencia DD, el MVS asume que el archivo al que se hace referencia en el DSN es NEW.

SENTENCIA DISP - PARÁMETROS POSICIONALES

SUBPARAMETROS ANTE TERMINACIÓN NORMAL DEL JOB

- **DELETE:** el archivo será eliminado.
- **KEEP:** el archivo es guardado para ser utilizado luego
- **CATLG:** cataloga el archivo

SUBPARAMETROS ANTE TERMINACIÓN ANORMAL DEL JOB

- **DELETE:** el archivo será eliminado.
- **KEEP:** el archivo es guardado para ser utilizado luego

SENTENCIA DISP - PARÁMETROS POSICIONALES

CONSIDERACIÓN PARA ARCHIVOS VSAM

- **Deben ser DISP = SHR u OLD**
- Para crearlos o eliminarlos se utiliza la función DEFINE o DELETE del programa Access Method Services (AMS)
- **No deben contener información UNIT o VOL**

SENTENCIA IDCAMS - PARÁMETROS POSICIONALES

```
//JOBPRU   JOB  ,'ALOCAR UN ARCHIVO VSAM',CLASS=A,MSGCLASS=A,
//            NOTIFY=USUARIOA
//*------------------- Delelte del CLUSTER ---------------------
//DEFCLUS  EXEC IDCAMS,COND=(4,LT)
            DELETE ARC.PRUEBA CLUSTER PURGE
//*------------------ Definición del CLUSTER --------------------
            DEFINE -
            CLUSTER(NAME(ARC.PRUEBA)) -
//*------------------- Definición del DATA ---------------------
            DATA -
            (NAME(ARC.PRUEBA.DATA) -
            CISZ(4096) -
            VOLUMES(DIS729) -
            CYL(30 10) -
            KEYS(22 0) -
            RECORDSIZE(250 250) -
            FREESPACE(26 10) -
            SHR(2 3)) -
```

SENTENCIA IDCAMS - PARÁMETROS POSICIONALES

```
//*------------------- Definición del INDEX ---------------------
                    INDEX -
          (NAME(ARC.PRUEBA.INDEX) -
          VOLUMES(DIS729) -
          CYL(5 1) -
          SHR(2 3)) -

//
```

SENTENCIA DD - PARÁMETROS POSICIONALES

CONCATENACIÓN DE ARCHIVOS

```
//JOBPRU   JOB  (5556,DX),'CONCAT',CLASS=A,MSGCLASS=A,
//           NOTIFY=USUARIOA
//JOBLIB   DD   PRODUCC.BIBLIO.FUENTES,DISP=SHR
//           MONITOR.BIBLIO.FUENTES,DISP=SHR
//           DESARR.BIBLIO.FUENTES,DISP=SHR
//PASO01   EXEC PGM=PROG1,REGION=2M,TIME=5
//ARCHIN   DD   DSN=SECUEN.ARCHIVO.ENTRADA1,DISP=SHR
/        DD   DSN=SECUEN.ARCHIVO.ENTRADA2,DISP=SHR
//ARCHSAL  DD   DSN=SECUEN.ARCHIVO.SALIDA,UNIT=SYDA
//           DCB=(RECFM=FB,LRECL=1164),
//           DISP=(NEW,CATLG,DELETE),
//           SPACE=(TRK,(10,5),RLSE)
//
```

En el ejemplo los dos archivos de entrada (tienen el mismo formato y longitud de registro) se concatenan como input único del programa PROG1. **No se pueden concatenar archivos VSAM**

SENTENCIA DD - PARÁMETROS POSICIONALES

```
//JOBPRU   JOB  ,'JOB CON PARM',CLASS=A,MSGCLASS=A,
//            NOTIFY=USUARIOA
//JOBLIB   DD   DESARR.BIBLIO.PROCS,DISP=SHR
//PASO01   EXEC PGM=PROG1,REGION=6M
//ARCHOUT  DD   DSN=SECUEN.ARCHIVO.PRUEBA,UNIT=SYSDA,
//            DCB=(RECFM=FB,LRECL=1164),
//            DISP=(NEW,KEEP,DELETE),
//            SPACE=(TRK,(10,5),RLSE)
//
```

En el ejemplo se asigna una unidad SYSDA, es decir no se especifica el volumen.

SENTENCIA VOL - PARÁMETROS POSICIONALES

- **VOL:** Se lo utiliza para identificar el volumen en el cual está almacenado el archivo de entrada o será almacenado el archivo de salida.
- **VOL=SER=volumen**
- (volumen1,volumen2,.,volumen n)
- **VOL=SER=volumen**
- Se especifica el número de serie del volumen, teniendo la certeza de que el archivo, cuando es de salida, no superará la capacidad del DASD.

SENTENCIA VOL - PARÁMETROS POSICIONALES

```
//JOBPRU      JOB     ,'JOB CON PARM',CLASS=A,MSGCLASS=A,
//                    NOTIFY=USUARIOA
//JOBLIB      DD      DESARR.BIBLIO.PROCS,DISP=SHR
//PASO01      EXEC    PGM=PROG1,REGION=6M
//ARCHOUT     DD      DSN=SECUEN.ARCHIVO.PRUEBA,UNIT=SYSDA,
//                    DCB=(RECFM=FB,LRECL=1164),
//                    DISP=(NEW,KEEP,DELETE),
//                    SPACE=(TRK,(10,5),RLSE),
//                    VOL=SER=DIS729
//
```

SENTENCIA VOL - PARÁMETROS POSICIONALES

- **VOL=SER=(volumen1,volumen2,.,volumen n)**
- Hay veces en las que los archivos de datos superan el espacio disponible en un DASD. A estos archivos se los denomina multivolumen.
- El ZOS respeta la secuencia de volúmenes especificados. Este método de creación de archivo multivolumen se lo denomina de pedido específico.

SENTENCIA VOL - PARÁMETROS POSICIONALES

```
//JOBPRU     JOB     ,'JOB CON PARM',CLASS=A,MSGCLASS=A,
//                   NOTIFY=USUARIOA
//JOBLIB     DD      DESARR.BIBLIO.PROCS,DISP=SHR
//PASO01     EXEC    PGM=PROG1,REGION=6M
//ARCHOUT    DD      DSN=SECUEN.ARCHIVO.PRUEBA,UNIT=3390,
//                   DCB=(RECFM=FB,LRECL=1164),
//                   DISP=(NEW,KEEP,DELETE),
//                   SPACE=(CYL(500,100),RLSE),
//                   VOL=SER=(DIS729,DIS731)
//
```

SENTENCIA DD - PARÁMETROS POSICIONALES

- **DCB:** En general cuando se utilizan archivos de datos, es necesario especificarle al ZOS:
  - La medida del bloque físico (blocksize)
  - La medida del registro (logical record length)
  - El formato del registro (record format)

- Si esta información no ha sido codificada en el programa, se la puede especificar a través del parámetro DCB de la sentencia DD.

- **DCB=(LRECL=longitud de registro,BLKSIZE=tamaño del bloque, RECFM=formato del registro)**

SENTENCIA RECFM - PARÁMETROS POSICIONALES

- **RECFM=V** longitud variable, desbloqueado. En este caso LRECL se lo debe definir como el valor del registro más largo del usuario más 4 bytes. El BLKSIZE se debe especificar como LRECL+4. Estos 4 bytes adicionales son utilizado por el MVS para llevar un control de la longitud variable registro.

- **RECFM=VB** longitud variable, bloqueado. Idem caso anterior.

- **RECFM=VA o VBA** longitud variable, de salida. El primer byte del registro se utiliza para el control de la impresora.

SENTENCIA RECFM - PARÁMETROS POSICIONALES

```
//JOBPRU        JOB     ,'SALIDA',CLASS=O,MSGCLASS=A,
//                      NOTIFY=USUARIOA
//JOBLIB        DD      DESARR.BIBLIO.PROCS,DISP=SHR
//PASO01        EXEC    PGM=PROG1,REGION=6M
//ARCHOUT       DD      DSN=SECUEN.ARCHIVO.PRUEBA,UNIT=3390,
//                      DCB=(RECFM=VB,LRECL=204,BLKSIZE=204),
//                      DISP=(NEW,KEEP),
//                      SPACE=(TRK(10,5),RLSE),
//                      VOL=SER=DIS729
//
```

En el ejemplo a la longitud máxima de registro de 200 bytes se le agregan 4 bytes por ser de longitud variable

SENTENCIA SPACE - PARÁMETROS POSICIONALES

- **SPACE:** Indica la cantidad de espacio primario y secundario que ocupará el nuevo archivo

- SPACE=(medida del bloque,(primario,secundario),RLSE)
  TRK (pista)
  CYL (cilindro)

- **Asignación primaria:** es lo que se estima que el archivo ocupará. El ZOS tratará de asignar la cantidad primaria en cinco o menos ubicaciones distintas dentro del mismo volumen DASD. A estas ubicaciones se las llama dominios.

- **Asignación secundaria:** de no alcanzar la asignación primaria, el ZOS agrega la cantidad de espacio especificada en la asignación secundaria hasta en 16 dominios.

SENTENCIA RLSE - PARÁMETROS POSICIONALES

- **Sub parámetro RLSE:**
- Este sub parámetro del parámetro SPACE de la sentencia DD se lo utiliza para liberar el espacio sobrante en la asignación primaria.
- **Cuando se usen archivos VSAM las sentencias DD:**
- No se debe especificar el parámetro SPACE. El espacio para los archivos VSAM se provee cuando se crea el archivo usando la función DEFINE.

SENTENCIA SYSOUT - PARÁMETROS POSICIONALES

- **SYSOUT:**
- Para operar sobre salidas impresas de los resultados del Job, se utiliza el operando SYSOUT en la sentencia DD.
- **SYSOUT= ***
- *: la clase de salida será la especificada en el parámetro MSGCLASS de la sentencia JOB.

SENTENCIA SYSDUMP - PARÁMETROS POSICIONALES

- **SYSUDUMP**
- El vuelco de memoria es una herramienta tradicional de depuración. Cuando un programa cancela, en la memoria quedó registrada la clave para detectar el error.
- Para solicitarle al MVS un vuelco de memoria se deben usar la siguiente DDNAME reservada:
- Se debe incluir en el paso de trabajo en el que se quiere hacer un vuelco de memoria si el mismo termina anormalmente.

JCL

CONSULTAS?

Codeki™

¡Muchas gracias!