# Additional Operands
# Using Existing Datasets

## Lesson 2:

## Additional Operands



## UNIT Parameter

- The UNIT parameter instructs the system which device the dataset resides on.
- The unit is indicated in the following format:

```
UNIT=unit address
     device type
     esoteric name
```



## UNIT Parameter

- Unit Address
  - Consists of three hexadecimal characters specifying the channel, control unit, and unit number

    **Example:**

    `UNIT=180`
  - Specifying a unit address restricts the operating system and should only be used by systems programmers.



## UNIT Parameter

- Device Type
  - This option corresponds to the model number of the device.
  - The system assigns any available device of that type.

    **Example:**

    UNIT=3390 is a common disk device.

    UNIT=3420 is a common tape device.
  - Specifying the device type, in the UNIT parameter should be avoided.
    - Use the esoteric name.



## UNIT Parameter

- Esoteric Name
  - This name designates a group of devices assigned by the installation.
    - Devices in the group may not be of the same type.
  - When coding an esoteric name, the system assigns any available device from that group.

    **Example:**

    UNIT=SYSDA normally refers to all disk drives on the system.
  - Research the esoteric names using the installation's standards manual.



## VOLUME or VOL Parameter

- The VOLUME or VOL parameter specifies the serial number of the volumes containing the dataset.

  `VOL=SER=serial`

  `VOLUME=SER=serial`
  - The serial number can be up to six alphanumeric characters.

**Examples:**

```
//DD1 DD DSN=SYSDASET,DISP=SHR,UNIT=SYSDA,VOL=SER=SYSED5
//DD2 DD DSN=TAPESET,DISP=MOD,UNIT=TAPE,VOL=SER=330218
```

- These examples demonstrate the UNIT and VOLUME parameters.

# Additional Operands
# Using Existing Datasets

## Concatenating Datasets

- There will be requirements for a program to treat several input datasets as a single file.
  - Concatenation is a technique used for connecting multiple input datasets into a single dataset.
- In order to concatenate datasets, code a DD statement describing each dataset.
  - The first DD statement must specify the ddname required by the programs.
  - The ddname fields on all concatenated statements must be blank.



## Concatenating Datasets

- Separate datasets for each week of the month are merged into a single MONTH dataset.

```
//MERGE       EXEC PGM=IEBGENER
//SYSUT1 DD DSN=WEEK1,DISP=(OLD,DELETE,KEEP),
//       UNIT=SYSDA,VOL=SER=SYSED8
//      DD DSN=WEEK2,DISP=(OLD,DELETE,KEEP),
//       UNIT=SYSDA,VOL=SER=SYSED5
//      DD DSN=WEEK3,DISP=(OLD,DELETE,KEEP),
//       UNIT=SYSDA,VOL=SER=SYSED8
//      DD DSN=WEEK4,DISP=(OLD,DELETE,KEEP),
//       UNIT=SYSDA,VOL=SER=SYSED3
//SYSUT2 DD DSN=MONTH,DISP=OLD,
//       UNIT=SYSDA,VOL=SER=SYSEDO
```

- For cataloged datasets, the UNIT and the VOL are not required.



## Concatenating Datasets

- When the program begins reading the file, z/OS retrieves the first record from the first dataset, WEEK1.
- When it reaches the last record in WEEK1, z/OS does not indicate the end-of-file to the program.
  - Instead, z/OS continues without interruption using WEEK2's records.
  - At the end of the last concatenated dataset, WEEK4, z/OS notifies the program that the end-of-file has been reached.



## Tape Datasets

- The parameters taught previously, DSN, DISP, UNIT, and VOL are applicable to both disk and tape datasets.
  - They will handle the majority of tapes that are encountered.
- There are situations in which the basic parameters will insufficient:
  - Tapes which contain multiple datasets.
  - Datasets which fill multiple volumes.
  - Tapes with non standard labels.



## Tape Datasets

- These three situations can be addressed by using the:
  - LABEL parameter.
  - Subparameters of UNIT and VOLUME.



## Tapes with Multiple Datasets

- A single tape can hold a significant amount of data.
  - Normally, tape is used for large datasets which do not require the rapid access of disk.
- Each tape usually contains one dataset.
  - However, tape is sometimes used for small datasets.
    - For example, one tape can contain Jan fund dataset, Feb fund dataset, and March fund dataset.
  - This would occur in a situation in which it is necessary to back up important datasets or copy datasets to another installation.
  - In these situations, a single tape can hold many small datasets.

# Additional Operands
# Using Existing Datasets

## Tapes with Multiple Datasets

- **LABEL Parameter**
  - On a tape containing multiple datasets, the LABEL parameter designates the sequence number of the specified dataset.

```
LABEL=sequence
```

**Example:**

```
//INPUT DD DSN=SYSED.TAPEDS,
//      DISP=OLD
//      UNIT=TAPE,
//      VOL=SER=552800,
//      LABEL=3
```



## Tapes with Multiple Datasets

- The code in the example has the dataset named TAPEDS as the third dataset on tape 522800.
  - There is a limitation with a tape when using multiple datasets on a volume.
    - During a step, the system can only access one dataset on the tape.
      - This means that a program can't read from two different datasets on one tape.
    - Nor can it read one dataset and write another on the same tape.



## Multiple-volume Datasets

- Some very large datasets are not able to fit on one tape.
  - Instead they span over two or more volumes.
  - JCL provides several options for processing multiple-volume datasets.



## Multiple-volume Datasets

- **Multiple Serial Numbers**
  - On the VOL parameter, specify the serial numbers of all volumes containing the dataset.

```
VOL=SER=(serial,serial,serial...)
```



## Multiple-volume Datasets

- When more than one volume is needed, enclose the serial numbers in parentheses.
  - They must be listed in sequence within the dataset.

**Example:**

  - The dataset MANYTAPE fills all of two and part of a third tape.

```
//DATAFILE DD DSN=MANYTAPE,
//         DISP=OLD,
//         UNIT=TAPE,
//         VOL=SER=(330218,230836,532661)
```



## Multiple-volume Datasets

- **Starting Sequence Number**
  - There may be a requirement not to start at the beginning of the dataset.
    - This can be done by providing the z/OS operating system with the sequence number of the volume from which to begin processing.
  - The starting sequence number is another VOL subparameter.

```
VOL=(,,start,SER=(serial,serial,serial...))
```

# Additional Operands
# Using Existing Datasets

## Multiple-volume Datasets

- The starting sequence number is the third positional subparameter of VOL.
  - If the keyword SER is the only subparameter that is needed, the positional subparameters can be ignored.
  - However, if these other subparameters are required, all the subparameters must be enclosed in parentheses.
- As with other positional parameters, absent subparameters must be indicated with a comma.



## Multiple-volume Datasets

**Example:**

- This code starts processing with the second volume of the dataset MANYTAPE:

```
//DATAFILE  DD  DSN=MANYTAPE,
//          DISP=OLD,
//          UNIT=TAPE,
//          VOL=(,,2,SER=(330218,230836,532661))
```



## Label Formats - Processing

- Tape labels contain information about the dataset such as the name, creation date, number of blocks, sequence number, blocksize, and whether the dataset spans multiple volumes.
  - z/OS checks the label during allocation and termination in order to ensure that the dataset has been fully processed.
- This label information may be written in one of several formats.
  - The LABEL parameter designates which format is to be used.

```
LABEL=(sequence,format)
```



## Label Formats - Processing

- Label formats may be designated by one of the following options:
  - SL: Standard Labels are the default for tapes produced by IBM OS type systems.
  - AL: ANSI Labels are the format used by most non-IBM systems.
  - NL: No Labels indicates that no label information is present.
    - Typically, NL is used when a company, such as a software vendor, distributes tapes to many different systems.
  - NSL: Nonstandard Labels are labels which do not conform to any standard format.
    - The installation must provide a special program to process these labels.



## Label Formats - Processing

- LTM: LTM stands for leading tape mark and precedes the label information.
  - This format is standard for IBM DOS type systems.
- BLP: BLP stands for bypass label processing and will ignore existing labels.
  - For most applications, standard labels should be used for all tapes.
  - Tapes that come from another location should either use BLP or AL.
  - When using something other than SL, consult the installation standards in order to determine the label format.



## Label Formats - Processing

**Example:**

```
//TAPE1    DD DSN=ANSITAPE,
//         DISP=OLD,
//         UNIT=TAPE,
//         VOL=SER=000222,
//         LABEL=(,AL)

//TAPE2    DD DSN=NONLABEL,
//         DISP=OLD,
//         UNIT=TAPE,
//         VOL=SER=000000,
//         LABEL=(3,NL)
```

- Label format is the second positional subparameter of the LABEL parameter.

# Additional Operands
# Using Existing Datasets

## Label Formats - Processing

- The first example has the LABEL parameter processing the first dataset on a tape with an ANSI label.
- The second example designates the third dataset on a tape that does not have labels.
- The vast majority of tapes in an installation will have the default standard labels.
  - The other options will be important when processing tapes created at another installation.



## Tape Operations - Improving

- Every time a job needs a tape dataset, the job must wait on the computer operator or the automatic tape system.
- The operator must:
  1. Find the requested tape in the library.
  2. Go to the tape drive and remove the tape used by a previous job.
  3. Double check the serial number and mount the tape.



## Tape Operations - Improving

- z/OS provides two UNIT parameter options which reduce the time a job must wait for tape mounts.

  `UNIT=(device,[count],[DEFER])`

  - Device indicates the unit address, device type, or esoteric name requested for the dataset.
  - Count specifies the number of units to be allocated for this dataset.
  - DEFER instructs z/OS not to mount the tape until it is opened.



## Tape Operations - Improving

- **Requesting Multiple Units**
  - The count subparameter allows multiple units to be dedicated to a dataset.
  - This permits a program to process a multiple-volume dataset without delays for tape mounts.

**Example:**

```
//DATAFILE   DD DSN=MANYTAPE,
//           DISP=OLD,
//           UNIT=(TAPE,2),
//           VOL=SER(330218,230836,532661)
```



## Tape Operations - Improving

- This subparameter instructs z/OS to allocate two devices from the TAPE group.
  - During allocation, z/OS mounts the first two volumes, 330218 and 230836.
  - When the program finishes the first tape, it continues processing the second tape without interruption.
  - While the second tape is being processed, the operator replaces the first volume with the third volume, 532661.
- Accordingly, the program can continue without waiting for tape mounts.



## Requesting Multiple Units

- The count subparameter allows multiple units to be dedicated to a dataset.
  - This permits a program to process a multiple-volume dataset without delays for tape mounts.

**Example:**

```
//    DATAFILE   DD DSN=MANYTAPE,
//    DISP=OLD,
//    UNIT=(TAPE,2),
//    VOL=SER(330218,230836,532661)
```

  - This subparameter instructs the z/OS operating system to allocate two devices from the TAPE group.

## Deferred Mounting

**Deferred Mounting**

- The final UNIT subparameter instructs z/OS not to mount the tape during allocation.
- Instead, the tape will be mounted when the program opens the file.

```
//LOG  DD      DSN=ERRORLOG,
//            DISP=MOD,
//            UNIT=(TAPE,,DEFER),
//            VOL=SER=717484
```

CETI/COMPUTER EDUCATION TECHNIQUES, INC. 2-2:31

## Deferred Mounting

- For the deferred mounts to be effective, a program must OPEN a file only when it actually is used.
- The common technique of opening all files at the beginning of the program, defeats any benefit gained by deferring the mount.

CETI/COMPUTER EDUCATION TECHNIQUES, INC. 2-2:32