Unit 4

- Time Sharing Option, Interactive System Productivity Facility (ISPF) Editor, System Display and Search Facility (SDSF), TN3270, SSH, FTP
- Datasets and Data Access Methods: Dataset Names and Types, Sequential Dataset, Partitioned Datasets, VSAM - Virtual Storage Access Method, Disk Storage, Volume Labels, VTOC, Extents, (System Utilities and Tools, Dataset management utilities (IEBGENER, IDCAMS, etc.), DFSORT basic and examples.

How do we interact with z/OS

- z/OS is ideal for processing batch jobs, that is,
- workloads that run in the background with little or no human interaction.
- z/OS is just as much an interactive operating system as it is a batch processing system.
- By *interactive, we mean that users (sometimes tens of* thousands of them concurrently, in the case of z/OS) can use the system through direct interaction, such as commands and menu style user interfaces
  - Time Sharing Option/Extensions
  - ISPF
  - UNIX shell and utilities

Time Sharing Option/Extensions

- Time Sharing Option/Extensions (TSO/E) allows users to create an interactive session with the z/OS system. TSO provides a single-user logon capability and a basic command prompt interface to z/OS.
- In a z/OS system, each user is granted a user ID and a password authorized for TSO logon. Logging on to TSO requires a 3270 display device or, more commonly, a TN3270 emulator running on a PC.

Typical TSO/E logon panel

```
------------------------------ TSO/E LOGON ------------------------------

   Enter LOGON parameters below:                RACF LOGON parameters:

  Userid    ===> ZPROF

   Password  ===>                               New Password ===>

   Procedure ===> IKJACCNT                      Group Ident  ===>

   Acct Nmbr ===> ACCNT#

   Size      ===> 860000

   Perform   ===>

   Command   ===>

   Enter an 'S' before each option desired below:
           -Nomail         -Nonotice        -Reconnect        -OIDcard

PF1/PF13 ==> Help    PF3/PF15 ==> Logoff    PA1 ==> Attention    PA2 ==> Reshow
You may request specific help information by entering a '?' in any entry field
```

Figure 4-1 Typical TSO/E logon panel

Using CLISTs and REXX under TSO

- With native TSO, it is possible to place a list of commands, called a *command list* (CLIST) in a file, and execute the list as though it were one command. When you invoke a CLIST, it issues the TSO/E commands in sequence.
- CLISTs are used for performing routine tasks
- TSO users create CLISTs with the CLIST command language and Restructured Extended Executor (REXX).
- Both CLIST and REXX offer shell script-type processing.
- These are *interpretive languages, as opposed to compiled languages*
- CLIST programming is unique to z/OS, while the REXX language is used on many platforms.

ISPF overview

- Interactive System Productivity Facility
- A facility of z/OS that provides access to many of the functions most frequently needed by users.
- After logging on to TSO, users typically access the ISPF menu.
- Many users use ISPF exclusively for performing work on z/OS.
- ISPF is a full panel application navigated by keyboard. ISPF includes a text editor and browser, and functions for locating and listing files and performing other utility functions.

```
 Menu  RefList  Utilities  Help
 ------------------------------------------------------------------------------
Allocate New Data Set
 Command ===>
Data Set Name  . . . : ZCHOL.TEST.CNTL
Management class . . .                (Blank for default management class)
 Storage class  . . . .                (Blank for default storage class)
  Volume serial . . . . TEST01         (Blank for system default volume) **
  Device type . . . . .                (Generic unit or device address) **
 Data class . . . . . .                (Blank for default data class)
  Space units . . . . . TRACK          (BLKS, TRKS, CYLS, KB, MB, BYTES
                                        or RECORDS)
  Average record unit                  (M, K, or U)
  Primary quantity  . . 2              (In above units)
  Secondary quantity    1              (In above units)
  Directory blocks  . . 0              (Zero for sequential data set) *
  Record format . . . . F
  Record length . . . . 80
  Block size  . . . . .
  Data set name type  :                (LIBRARY, HFS, PDS, or blank)  *
                                       (YY/MM/DD, YYYY/MM/DD
  Expiration date . . .                 YY.DDD, YYYY.DDD in Julian form
 Enter "/" to select option             DDDD for retention period in days
    Allocate Multiple Volumes           or blank)

 ( * Specifying LIBRARY may override zero directory block)

 ( ** Only one of these fields may be specified)
  F1=Help F2=Split F3=Exit F7=Backward F8=Forward F9=Swap F10=Actions  F12=Cancel
```

Figure 4-5 Allocating a data set using ISPF panels

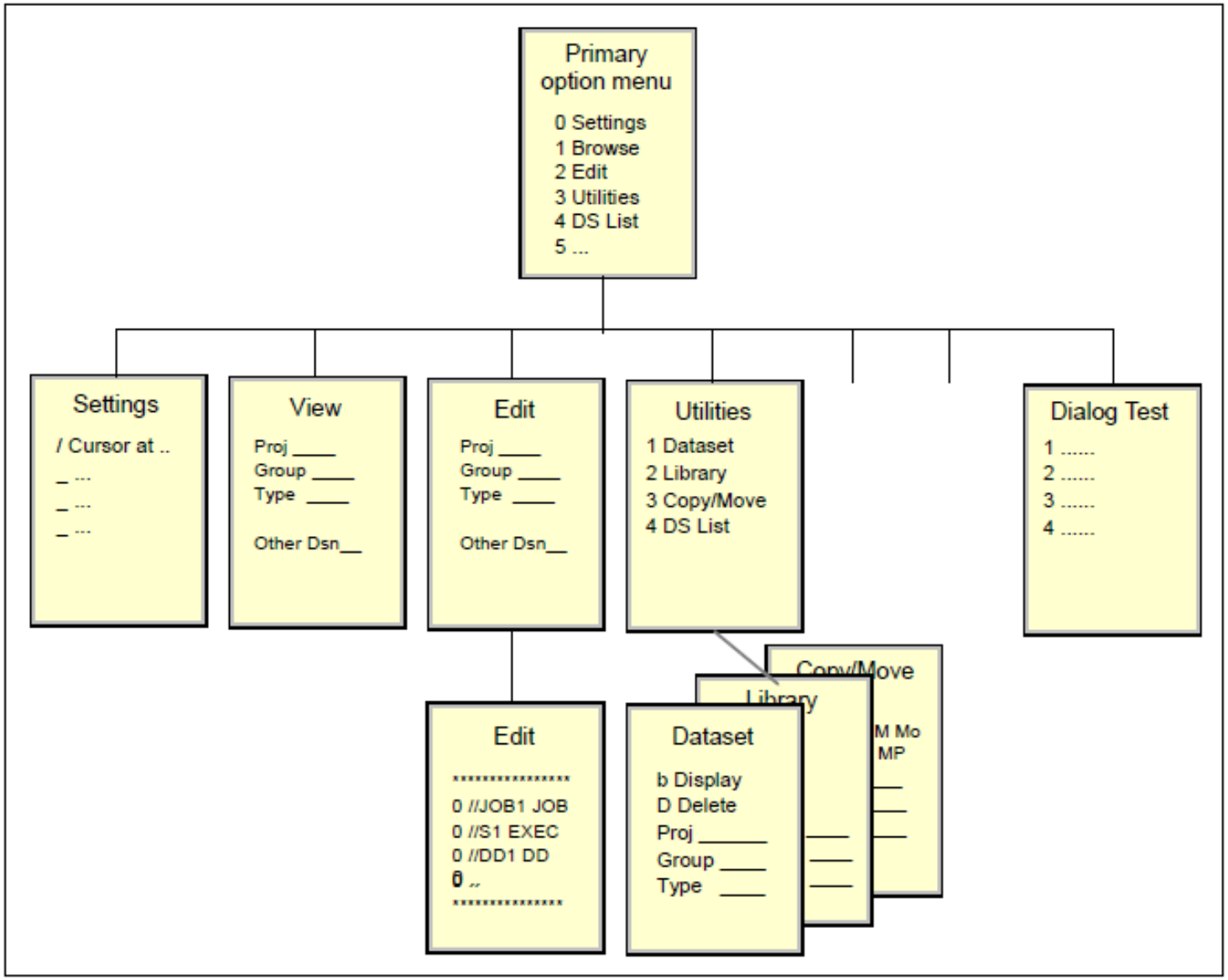

Figure 4-7 ISPF menu structure

Table 4-1 Keyboard mapping

Function	Key
Enter	Ctrl (right side)
Exit, end, or return	PF3
Help	PF1
PA1 or Attention	Alt-Ins or Esc
PA2	Alt-Home
Cursor movement	Tab or Enter
Clear	Pause
Page up	PF7
Page down	PF8
Scroll left	PF10
Scroll right	PF11
Reset locked keyboard	Ctrl (left side)

ISPF Tutorial main menu

```
Tutorial ---------------------- Table of Contents -------------------- Tutorial

                  ISPF Program Development Facility Tutorial

The following topics are presented in sequence, or may be selected by entering
a selection code in the option field:
     G  General      - General information about ISPF
     0  Settings     - Specify terminal and user parameters
     1  View         - Display source data or output listings
     2  Edit         - Create or change source data
     3  Utilities    - Perform utility functions
     4  Foreground   - Invoke language processors in foreground
     5  Batch        - Submit job for language processing
     6  Command      - Enter TSO command, CLIST, or REXX exec
     7  Dialog Test  - Perform dialog testing
     9  IBM Products - Use additional IBM program development products
     10 SCLM         - Software Configuration and Library Manager
     11 Workplace    - ISPF Object/Action Workplace
     X  Exit         - Terminate ISPF using log and list defaults
     The following topics will be presented only if selected by number:
     A  Appendices   - Dynamic allocation errors and ISPF listing formats
     I  Index        - Alphabetical index of tutorial topics

F1=Help      F2=Split     F3=Exit      F4=Resize    F5=Exhelp    F6=Keyshelp
  F7=PrvTopic  F8=NxtTopic  F9=Swap     F10=PrvPage  F11=NxtPage  F12=Cancel
```

Figure 4-9 ISPF Tutorial main menu

ISPF Editor

- The ISPF Editor is a text-editing tool used within IBM's z/OS mainframe operating system.
- It provides a user-friendly interface for editing source code, data, and configuration files.
- Designed to streamline editing tasks for mainframe users. Supports various editing functions and integrates with other ISPF components.

- You can perform the following tasks:
- To view a data set’s contents, enter a v (view) as a line command in the column.
- To edit a data set’s contents, enter an e (edit) as a line command in the column.
- To edit the contents of a data set, move the cursor to the area of the record to be changed and type over the existing text.
- To find and change text, you can enter commands on the editor command line.
- To insert, copy, delete, or move text, place these commands directly on the line numbers where the action should occur.
- To commit your changes, use PF3 or save. To exit the data set without saving your changes, enter Cancel on the edit command line.

ISPF Editor Interface Overview

- **Key Components:**
  - **Menu Bar:** Provides access to different editor functions and options.
  - **Command Line:** Where users input commands to execute various tasks (e.g., saving, searching).
  - **Data Entry Area:** Main area where the text or data is displayed and edited.
  - **Status Line:** Displays information about the current state of the editor and file.

Basic Navigation

- **Commands and Function Keys:**
- **Common Commands:**
  - EDIT: Enter edit mode.
  - VIEW: View a dataset in read-only mode.
  - BROWSE: Browse a dataset without editing.
  - CREATE: Create a new dataset.
- **Function Keys:**
  - F1: Help
  - F3: Exit
  - F2: Save
  - F4: Commands Menu

Adding a GUI to ISPF

- You can download and install a variety of ISPF graphical user interface(GUI) clients to include with a z/OS system

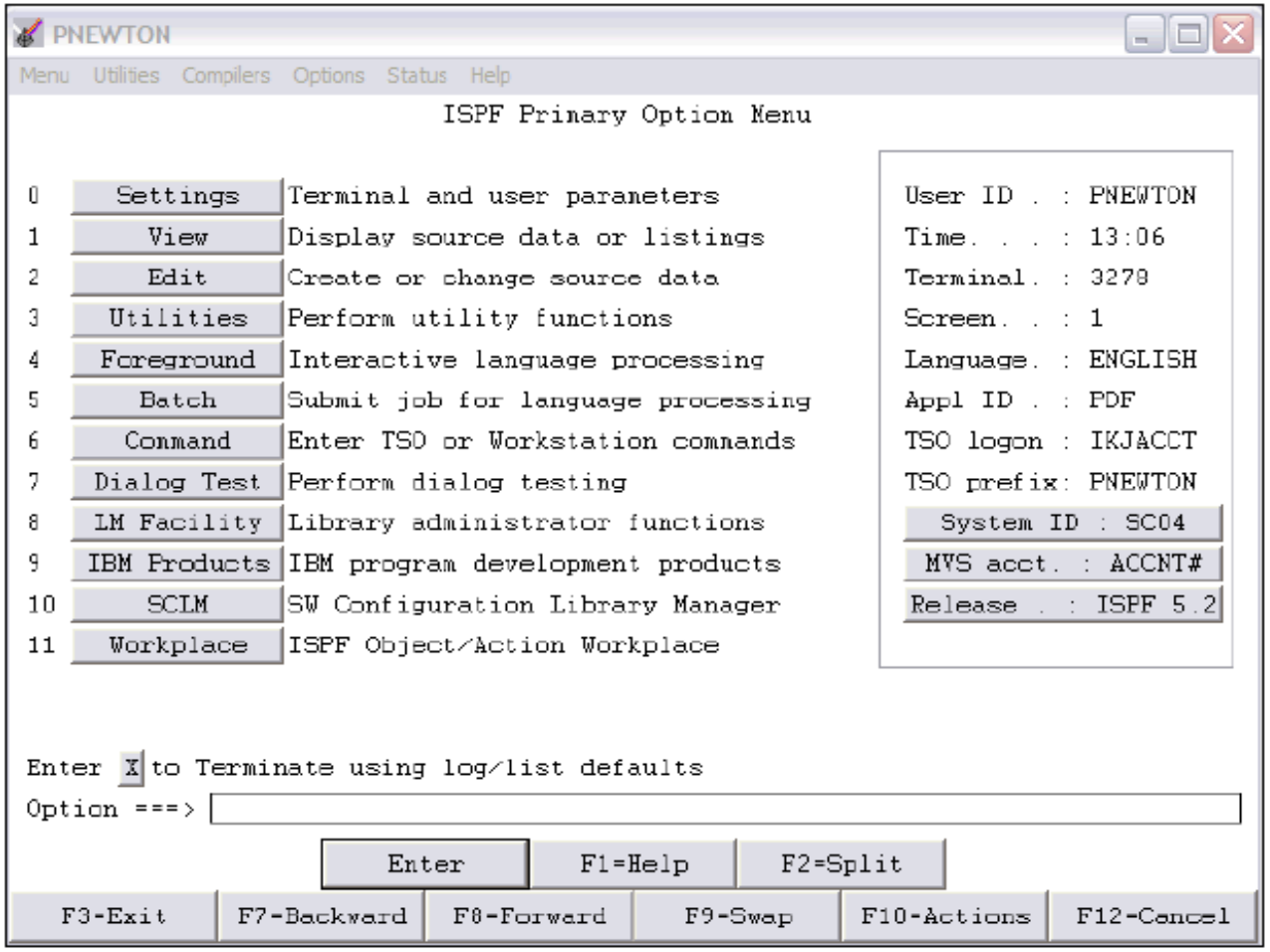

Figure 4-17 ISPF GUI

z/OS® UNIX® shell and utilities

- The z/OS UNIX shell and utilities provide an interactive interface to z/OS.
- To perform some command requests, the shell calls other programs, known as **utilities**.
- Invoke shell scripts and utilities.
- Write shell scripts (a named list of shell commands, using the shell programming language).
- Run shell scripts and C language programs interactively, in the TSO background or in batch.

z/OS® UNIX® shell and utilities

- A user can invoke the z/OS UNIX shell in the following ways:
  - From a 3270 display or a workstation running a 3270 emulator
  - From a TCP/IP-attached terminal, using the **rlogin and telnet commands**
  - From a TSO session, using the OMVS(Open MVS) command.

z/OS® UNIX® shell and utilities

- There are some differences between the asynchronous terminal support (direct shell login) and the 3270 terminal support (OMVS command):
- You cannot switch to TSO/E. However, you can use the TSO SHELL command to run a TSO/E command from your shell session.
- You cannot use the ISPF editor
- You can use the UNIX vi editor, and other interactive utilities that depend on receiving each key stroke, without hitting the Enter key.
- You can use UNIX-style command-line editing.

SDSF

- **System Display and Search Facility**
- Utility that allows you to monitor, control, and view the output of jobs in the system.
- After submitting a job, it is common to use the *System Display and Search Facility (SDSF) to review the output for successful completion or review and* correct JCL errors.

- SDSF provides a number of additional functions, including:
  - Viewing the system log and searching for any literal string
  - Entering system commands (in earlier versions of the operating system, only the operator could enter commands)
  - Controlling job processing (hold, release, cancel, and purge jobs)
  - Monitoring jobs while they are being processed
  - Displaying job output before deciding to print it
  - Controlling the order in which jobs are processed
  - Controlling the order in which output is printed
  - Controlling printers and initiators

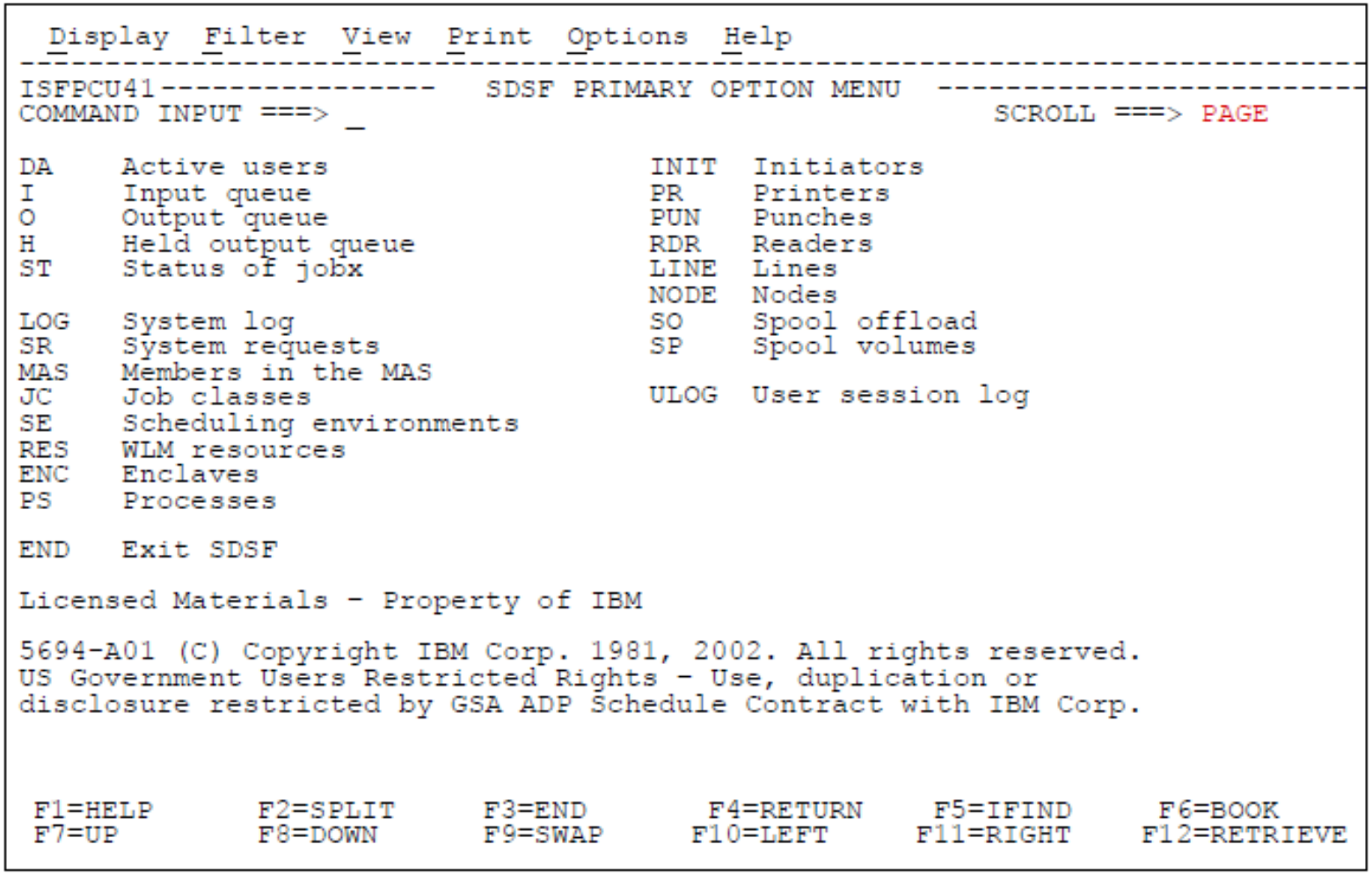

Figure 6-5 SDSF primary option menu

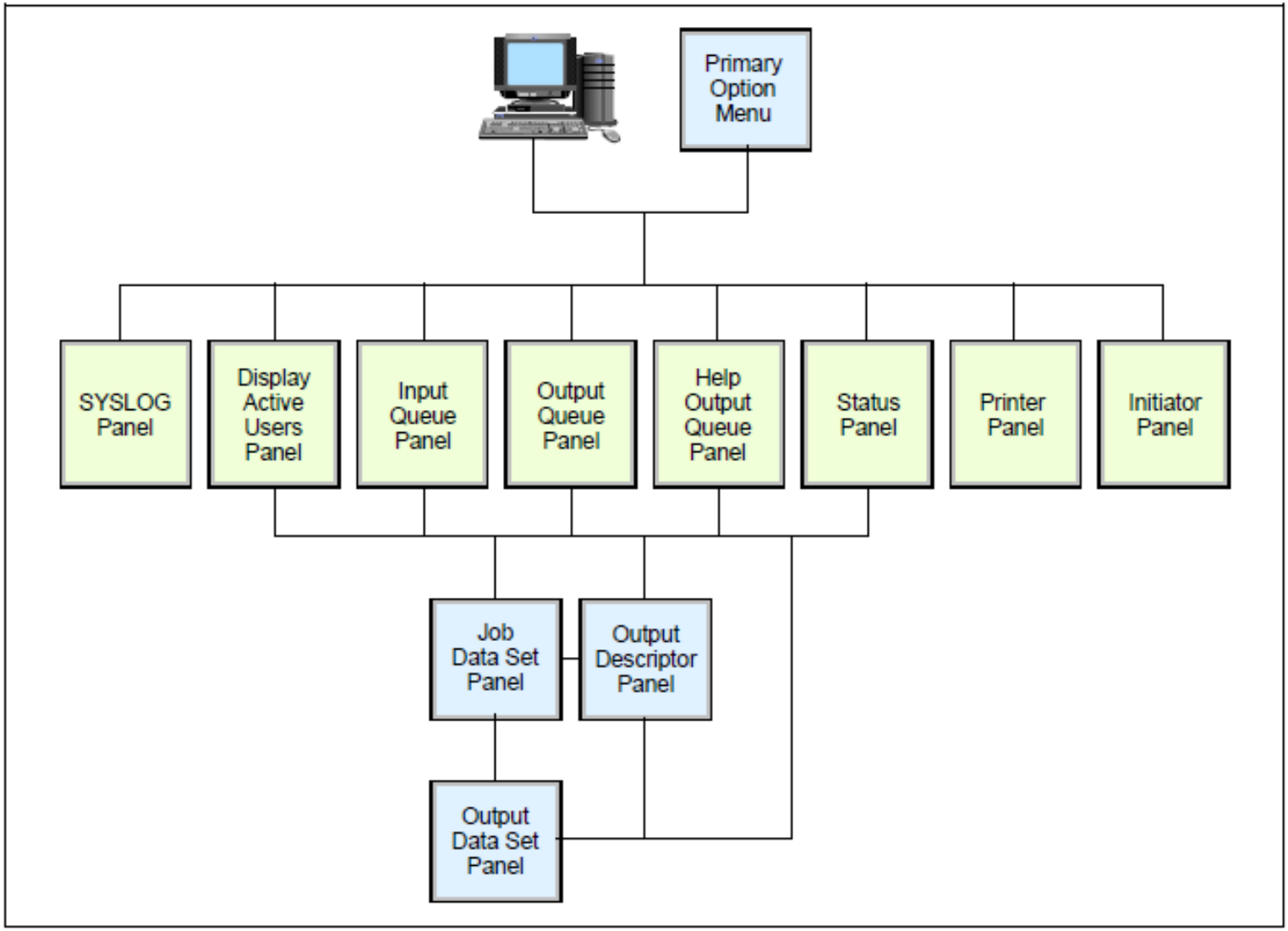

Figure 6-6 SDSF panel hierarchy

- You can see the JES output data sets created during the execution of your batch job. They are saved on the JES spool data set.
- You can see the JES data sets in any of the following queues:
- **I Input**
- **DA Execution queue**
- **O Output queue**
- **H Held queue**
- **ST Status queue**

Introduction to the 3270 terminal

- before the Internet , large organizations established their own SNA (Systems **Network** Architecture) networks.
- SNA networks were used to communicate between remote end-users and the centralized mainframe.
- The display management protocol used to facilitate this communication within an SNA environment was called the 3270 data stream.
- TN3270 is a protocol used to connect to IBM mainframe computers from a terminal emulator. It’s part of the Telnet protocol suite and is specifically designed to facilitate communication with IBM’s 3270 terminals, which were widely used with mainframes.
- At the end user's location in an SNA network was a device referred to as a 3270 terminal.
- A 3270 terminal was a non-programmable (sometimes called "dumb") workstation with display and keyboard

IBM 3270 Display Terminal

3270 data stream

- The 3270 data stream operations are designed primarily for transmitting data between an application program and a 3270 display with keyboard so end users can interact with mainframe-based applications.
- Today, a single instance of the TN3270E server can support up to 128 000 emulated 3270 display terminals.

- Datasets and Data Access Methods: Dataset Names and Types,
- Sequential Dataset,
- Partitioned Datasets
- VSAM - Virtual Storage Access Method,
- Disk Storage, Volume Labels, VTOC, Extents,

What is a data set

- a dataset is a collection of data that is stored together as a single unit. This is analogous to files in other computing environments
- A collection of logically related data records, such as a library of macros or a source program.
- z/OS manages data by using *data sets. The term data set refers to a file that* contains one or more records. Any named group of records is called a data set.
- Data sets can hold information such as medical records or insurance records that are used by a program running on the system.
- Data sets are also used to store information needed by applications or the operating system itself, such as source programs, macro libraries, or system variables or parameters
- a *record is a fixed number of bytes containing data.*

data set

- There are many different types of data sets in z/OS, and different methods for accessing them.
- Data sets can be permanent or temporary:
- three types of data sets:

1. sequential,
2. partitioned,
3. and VSAM data sets.

sequential data set

- simplest form of datasets
- In a *sequential data set, records are data items that are stored consecutively.*
- New records are appended to the end of the data set.
- *To* retrieve the tenth item in the data set, for example, the system must first pass the preceding nine items.
- Data items that must all be used in sequence, such as the alphabetical list of names in a classroom roster, are best stored in a sequential data set.
- often used for logs or batch processing.

PDS(partitioned data set)

- PDS consists of a *directory and members.*
- A PDS is a collection of sequential data sets, called *members. Each member is like a sequential data set and has a simple name,* which can be up to eight characters long.
- *The directory* holds the address of each member and thus makes it possible for programs or the operating system to access each member directly.
- A PDS also contains a directory. The directory contains an entry for each member in the PDS with a reference (or pointer) to the member. Member names are listed alphabetically in the directory, but members themselves can appear in any order in the library.
- Partitioned data sets are often called *libraries.*
- *Programs are stored as members* of partitioned data sets.
- Generally, the operating system loads the members of a
- PDS into storage sequentially, but it can access members directly when selecting a program for execution.
- commonly used for storing programs, scripts, or configuration files.

Advantages of a partitioned data set

- Grouping of related data sets under a single name saves you disk space
- Members of a PDS can be used as sequential data sets, and they can be appended (or *concatenated) to sequential data sets.*
- Multiple PDS data sets can be concatenated to form large libraries.
- PDS data sets are easy to create with JCL or ISPF; they are easy to manipulate with ISPF utilities or TSO commands.

Disadvantages of a partitioned data set

- When a member is deleted, its pointer is deleted too, so there is no mechanism to reuse its space. This wasted space is often called *gas and must be periodically removed by* reorganizing the PDS, for example, by using the IEBCOPY utility to compress it.
- Limited directory size: The size of a PDS directory is set at allocation time.
- Lengthy directory searches: an entry in a PDS directory consists of a name and a pointer to the location of the member. Entries are stored in alphabetical order of the member names. Inserting an entry near the front of a large directory can cause a large amount of I/O activity, as all the entries behind the new one are moved along to make room for it.

Virtual Storage Access Method (VSAM)

- The term Virtual Storage Access Method (VSAM) applies to both a data set type and the access method used to manage various user data types.
- VSAM is used primarily for applications. It is not used for source programs, JCL, or executable modules
- VSAM to organize records into four types of data sets:
- key-sequenced, entry-sequenced, linear, or relative record.

1) KSDS (Key-Sequenced Data Set): Data is organized by a key, allowing for efficient retrieval. Keys are used to index records, enabling fast access and updates.

- **Usage**: Suitable for applications needing quick lookups, such as customer records or transaction databases.
- **Access Method**: Indexed access, where records are retrieved using their key. This provides random access to data. Records are of variable length.

Virtual Storage Access Method (VSAM)

- **ESDS (Entry-Sequenced Data Set)**: Data is stored in the order in which it is entered, without indexing. Records are accessed in the sequence they were added or by relative record number (RRN).
- **Usage**: Useful for logs or data where order is important but indexing is not required.
- **Access Method**: Sequential or direct access by RRN.
- **RRDS (Relative Record Data Set)**: Records are accessed based on their relative position in the dataset rather than by key.
- **Usage**: Used when records need to be accessed by a relative position or index.
- **Access Method**: Direct access by relative record number, This provides random access

Virtual Storage Access Method (VSAM)

- Linear Data Set (LDS):This is, in effect, a byte-stream data set and is the only form of a byte-stream data set in traditional z/OS files (as opposed to z/OS UNIX files). A number of z/OS system functions use this format heavily, but it is rarely used by application programs.

Virtual Storage Access Method (VSAM)

- VSAM works with a logical data area known as a control interval (CI), which is shown in Figure 5-2. The default CI size is 4 KB, but it can be up to 32 KB. The CI contains data records, unused space, record descriptor fields (RDFs), and a CI descriptor field.
- VSAM data is always variable-length and records are automatically blocked in control intervals.

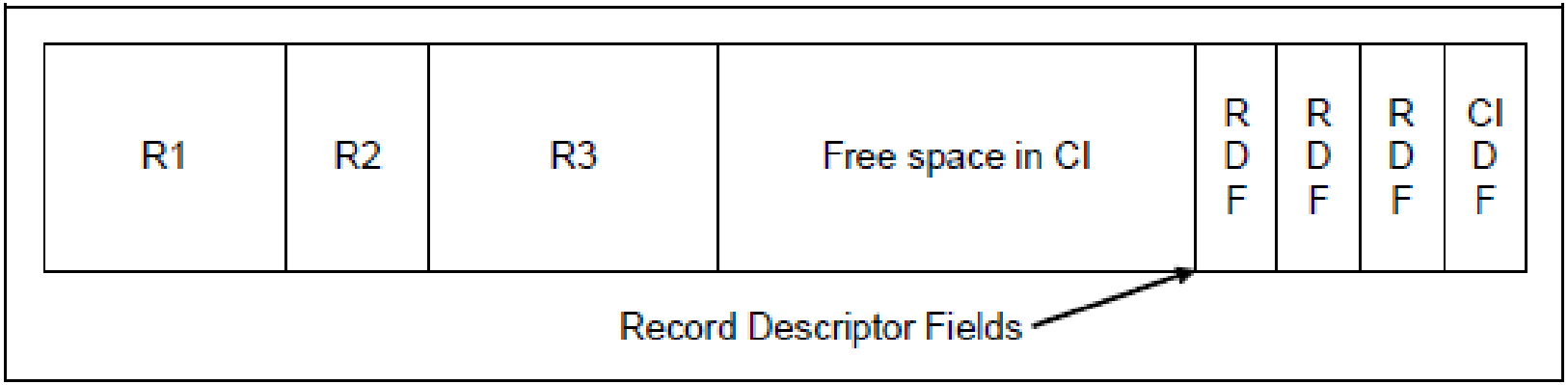

Figure 5-2 Simple VSAM control interval

Dataset structure

- Data set- Data sets are stored on tape or disks.
- Direct Access Storage Device (DASD)*DASD* is another name for a disk drive.
- Space- Disk space is allocated in units called cylinders, tracks, or blocks.
- Cylinder-A disk drive contains *cylinders*. A cylinder is a unit of storage with a fixed number of tracks.
- Track-Cylinders contain *tracks*, which are circular paths on the surface of a disk or diskette on which information is magnetically recorded and are in Count Key Data (CKD) format
- Record-Tracks contain records. A *record* is some number of bytes containing data. Records are either fixed length or variable length
- Blocks-Records can be grouped into data *blocks*, which are the units of recording on disk.
- Volume-The term *volume* is often used to refer to a disk.
- Volume serial-The six-character name of a disk or tape volume, such as TEST01.
- Device type-A model or type of disk device, such as 3390.OrganizationThe method of processing a data set, such as sequential.

What are access methods

- An access method defines the technique that is used to store and retrieve data.
- Access methods have their own data set structures to organize data, system-provided programs (or **macros**) to define data sets, and utility programs to process data sets.
- Access methods are identified primarily by the data set organization
- Commonly used access methods include the following:
  - QSAM Queued Sequential Access Method (heavily used)
  - BSAM Basic Sequential Access Method (for special cases)
  - BDAM Basic Direct Access Method (becoming obsolete)
  - BPAM Basic Partitioned Access Method (for libraries)
  - VSAM Virtual Storage Access Method (used for more complex
  - applications)

Disk Labels, VTOC, Extents

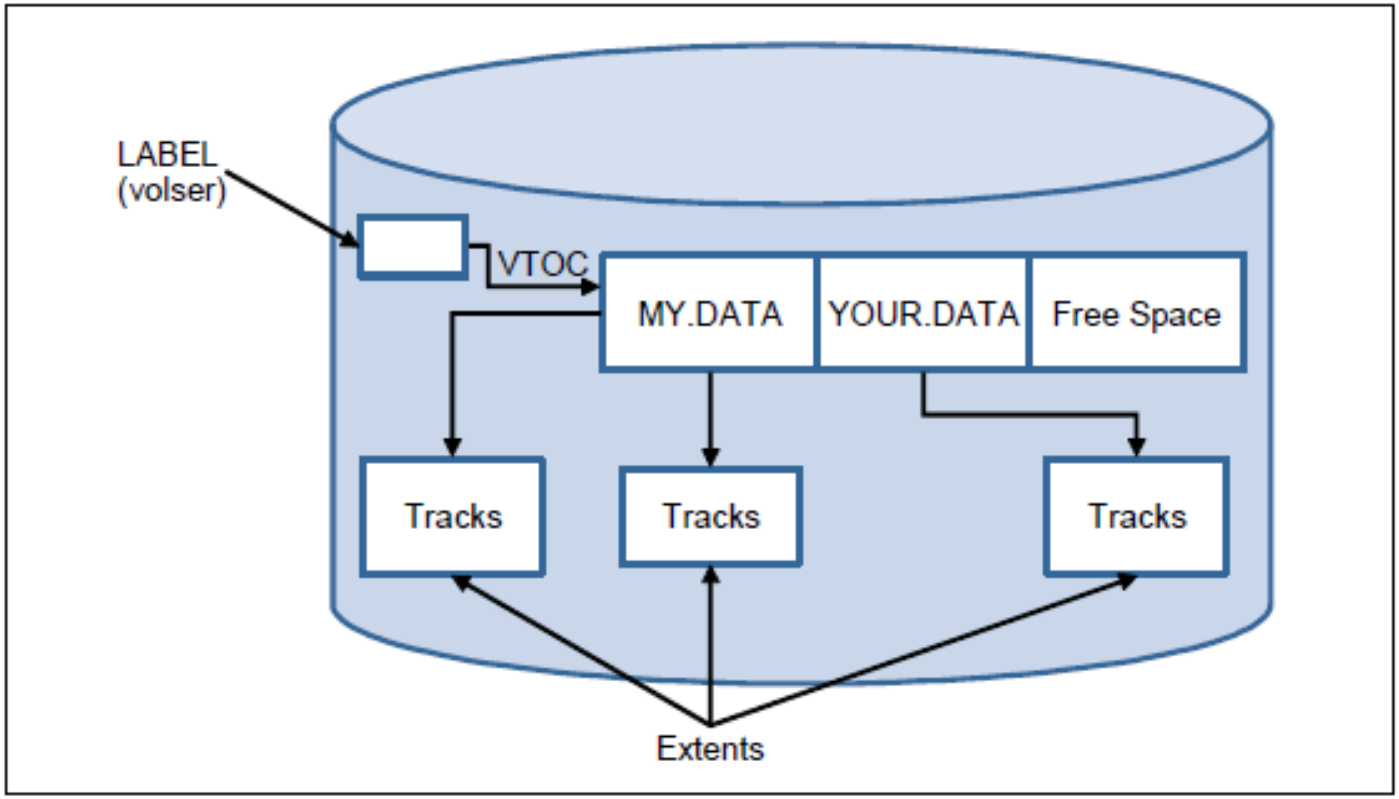

Figure 5-3 Disk label, VTOC, and extents

Disk Label

- A disk label is a record of metadata that resides at the beginning of a disk volume.
- It provides essential information about the volume, including its identity and some characteristics.
- **Purpose**: The disk label helps in identifying and managing disk volumes. It ensures that the system and administrators can correctly refer to and handle different volumes.

VTOC

- The VTOC is a table located on a disk volume (a physical storage device) that keeps track of the datasets stored on that volume.
- It essentially acts as an index or directory for the volume, enabling the system to efficiently locate and manage datasets.
- **Structure of VTOC:**
- **Table Format**: The VTOC is organized in a structured format that includes entries for each dataset on the volume. Each entry typically contains information such as the dataset name, size, location on the volume, and other attributes.
- **Physical Location**: The VTOC itself is stored at a specific location on the disk volume, often at the beginning or in a designated area of the volume.

VTOC

- Record 1 on the first track of the first cylinder provides the label for the disk. It contains the 6-character volume serial number (volser) and a pointer to the *volume table of contents (VTOC), which can be located anywhere on the disk.*
- The VTOC lists the data sets that reside on its volume, along with information about the location and size of each data set, and other data set attributes. A standard z/OS utility program, ICKDSF, is used to create the label and VTOC.
- The VTOC also has entries for all the free space on the volume.

VTOC

- **Key Functions of VTOC:**
- **Dataset Management**:
  - **Tracking**: The VTOC records information about the datasets stored on the volume, including their locations, sizes, and attributes.
  - **Organization**: It helps in organizing and managing datasets by maintaining metadata that allows the system to efficiently locate and access data.
- **Space Management**:
  - **Allocation**: The VTOC assists in tracking free space on the volume and managing dataset allocations. It keeps a record of which areas of the disk are used and which are free.
  - **Reclaiming**: When datasets are deleted or moved, the VTOC is updated to reflect the newly available free space.
- **Recovery**:
  - **Backup and Restore**: The VTOC plays a role in backup and recovery procedures by providing information needed to restore datasets to their correct locations on the volume.
- **Integrity**:
  - **Consistency Checking**: The VTOC helps in maintaining data integrity by ensuring that the information about datasets is consistent and up-to-date.

Extents

- An extent is a contiguous block of storage on a disk volume allocated to a dataset. Datasets are not necessarily stored in a single, continuous area; instead, they can span multiple extents.

Interaction Between These Components

- **Disk Label and VTOC**:
  - The disk label provides basic metadata about the volume, while the VTOC provides detailed information about the datasets stored on that volume. Both are essential for managing disk volumes but serve different purposes.
- **VTOC and Extents**:
  - The VTOC records information about the extents allocated to each dataset. It tracks how the dataset is spread across the volume and updates when extents are allocated or released.

JCL - Utility Programs

- IEFBR14- The utility program IEFBR14 performs no action other than return a completion code of 0; Dummy program often used for dataset operations.
- IDCAMS- Used for dataset management, such as cataloging, deleting, and renaming.
- IEBCOPY- IEBCOPY is a data set utility that is used to copy or merge members between one or more partitioned data sets, or partitioned data sets extended (PDSEs), in full or in part.
- IEBDG- Create a test data set consisting of patterned data
- IEBGENER- It is used to copy or print sequential data sets. Also used to copy any TAPE file to DISK or DISK to TAPE.
- IEBUPDTE- The IEBUPDTE utility creates multiple members in a partitioned data set, or updates records within a member. While it can be used for other types of records, its main use is to create or maintain JCL procedure libraries or assembler macro libraries.
- IEBCOMPR – It is used to compare the contents of two PS or PDS datasets.

- The IEBGENER utility is a copy program that has been part of the operating system since the first release of OS/360. One of its many uses is to copy a sequential data set, a member of a partitioned data set (PDS) or PDSE, or a z/OS® UNIX® System Services (z/OS UNIX) file such as a HFS file.
- IEBGENER also can filter data; change a data set's logical record length (LRECL) and block size (BLKSIZE); and generate records.

- //SMITH2 JOB 1,GEOFF,MSGCLASS=X
- // EXEC PGM=IEBGENER
- //SYSIN DD DUMMY //SYSPRINT DD SYSOUT=X
- //SYSUT1 DD DISP=SHR,DSN=SMITH.SEQ.DATA
  //SYSUT2 DD
  DISP=(NEW,CATLG),DSN=SMITH.COPY.DATA,UNIT=33
  90,
- // VOL=SER=WORK02,SPACE=(TRK,3,3))

- IEBGENER requires four data definition (DD) statements with the DD names shown in the example:
- The SYSIN DD statement is used to read control parameters; for simple uses, no control parameters are needed and a DD DUMMY can be used.
- The SYSPRINT statement is for messages from IEBGENER.
- The SYSUT1 statement is for input and the SYSUT2 statement is for output.
- This example reads an existing data set and copies it to a new data set.

IDCAMS

- IDCAMS stands for Integrated Data Cluster Access Method Services. IDCAMS utility is used to create, modify and delete the VSAM datasets.
- IDCAMS Utility is very useful utility to manipulate VSAM datasets

Example 1: Allocating and Loading Data Into VSAM

```
//JOBIBMKS JOB
     (123),'IBMMAINFRAMER',CLASS=C,MSGCLASS=S,MSGLEVEL=(1,1),
 // NOTIFY=&SYSUID
 //*
 //STEP001 EXEC PGM=IDCAMS
 //SYSPRINT DD *
//DATAIN DD DISP=OLD,DSN=userid.TEST.INPUT
//SYSIN DD *
DEFINE CLUSTER ( NAME (userid.TEST.VSAM) –
                        VOLUMES(WORK02) –
                       CYLINDERS(1 1) –
                        RECORDSIZE(72 100) –
                       KEYS(9 8) –
             INDEXED)
 REPRO INFILE(DATAIN) –
 OUTDATASET(userid.DATA.VSAM) –
ELIMIT(200) /*
```

Example 2: Deleting a VSAM dataset

```
//JOBIBMKS JOB
   (123),'IBMMAINFRAMER',CLASS=C,MSGCLASS=S,
   MSGLEVEL=(1,1),
 // NOTIFY=&SYSUID
//*
 //STEP001 EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
 DELETE useird.DATA.VSAM CLUSTER
/*
```

DFSORT basic and examples

- DFSORT is a program you use to sort, merge, and copy information.
- When you sort records, you arrange them in a particular sequence, choosing an order more useful to you than the original one.
- When you merge records, you combine the contents of two or more previously sorted data sets into one.
- When you copy records, you make an exact duplicate of each record in your data set.
- Merging records first requires that the input data sets are identically sorted for the information you will use to merge them and that they are in the same order required for output. You can merge up to 100 different data sets at a time.

DFSORT basic and examples

In addition to the three basic functions, you can perform other processing simultaneously:

1. You can control which records to keep in the final output data set.
2. You can parse, edit, and reformat your records before or after other processing.
3. You can sum numeric information from many records into one record.
4. You can create one or more output data sets for a sort, copy, or merge application.
5. You can perform various "join" operations on two files by one or more keys.

JCL Sort structue is as follows--

//STEP001 EXEC PGM=SORT,REGION=1024K,PARM=parameters

//SYSOUT DD SYSOUT=* Output messages from SORT

//SORTIN DD DSN=...,DISP=SHR Input if SORT request

//SORTOUT DD DSN=... Output for SORT request //SORTOFxx DD DSN=... OUTFILE output data sets //SORTXSUM DD DSN=... Output eliminated by the SUM stm //SORTWKnn DD UNIT=SYSDA, Work files if SORT request

//SYSIN DD * Control statement input data set sort control statements

/*

DFSORT basic and examples

- Listed below the processing order of the sort control statements:
  - SORTIN
  - SKIPREC
  - INCLUDE/OMIT
  - STOPAFT
  - INREC
  - SORT/SUM or COPY
  - OUTREC
  - OUTFIL
  - SORTOUT

DFSORT basic and examples

- **SORTIN** - Input record
- **SKIPREC=n** causes sort to skip over 'n' records in the input file before starting a sorting or copying operation.
- **INCLUDE** - Include to the record for further process; OMIT - Exclude the record for further processing
- **STOPAFT=n** causes sort to stop after 'n' records in the input file have been sorted or copied.
- **INREC** - It allows you to reformat the input records before they are sorted, merged, or copied.
- **SORT** - Sort the record; SUM - Remove duplicate or sum the fields; COPY - copy the record.
- **OUTREC** - It allows you to reformat the input records after they are sorted, merged, or copied.
- **OUTFIL** - It allows you to create one or more output data sets.
- **SORTOUT** - Output record