IBM Education

WHO
PASSWORD:
SF38TEAM07

SF38 - SYSTEM/38 SYSTEMS FACILITIES

Last revision January 1984

UK 52-0054-2

This document was prepared by:

IBM United Kingdom Ltd.,
Education Department.

All enquiries regarding this document should be addressed to:

IBM United Kingdom Ltd.,
Alencon Link,
PO Box 31,
Basingstoke,
Hampshire. RG21 1NR.
Telephone: Basingstoke (0256) 56144

IMPORTANT

Your attention is drawn to the following statement in the Customer Education Terms and Conditions:-

"The customer is responsible for ensuring that:-

a. Students do not disclose, provide or make available to any person other than their employer or IBM or remove from IBM's premises any program materials or copies thereof acquired during the course without IBM's prior written consent.

b. Students must not copy in whole or in part any Program Materials acquired during the course except under the supervision of and in accordance with the instructions of IBM personnel. Program Materials consist of the basic materials and their related optional materials for IBM Program Products, IBM Special Programs and other IBM Licensed programs."

Rest Areas

These are situated at the corner of B and C blocks on the first, second and third floors. Vending machines for coffee, tea, confectionery and change are provided in each rest area. There are also cloakrooms for coats and cases.

Restaurant

This is located on the fourth floor and is open for:

Breakfast	8.15	-	9.30
Morning Coffee	9.30	-	10.45
Luncheon	12.00	-	13.45
Evening Meal	17.15	-	18.45

A choice of hot meals, cold buffet, and special Grill Bar orders are available for lunch. Meals should be eaten from the tray which should be placed on the conveyor belt when you have finished your meal. Smoking is not permitted in the Restaurant, but you are welcome to do so in the coffee area which is adjacent. Fresh ground coffee is provided via a vending machine.

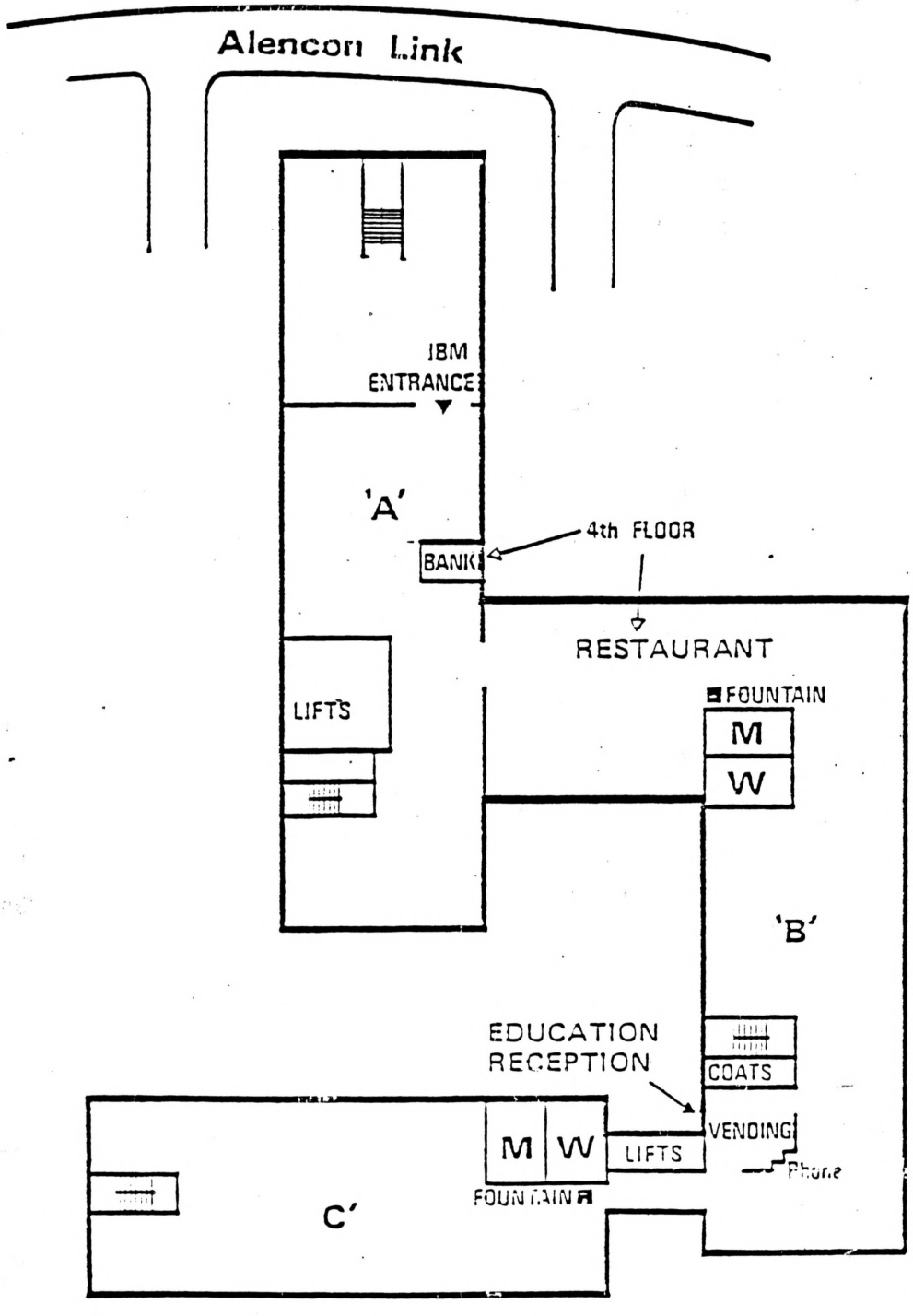
Telephones

Pay telephones are located in each rest area in addition to the booths in the reception area. These telephones are unable to accept incoming calls so if you think somebody will be ringing you up, please arrange it with your instructor. Messages that arrive for you will be accepted by the receptionist and placed on the notice board oppposite the lifts on the fourth floor.

Bank

There is a National Westminster sub-branch situated in A-block on the fourth floor outside the restaurant. It is open Monday to Friday between 11.30 am. and 2.30 pm.

IBM Basingstoke



SYSTEM/38 SYSTEMS FACILITIES

Overview: This course provides practical experience in the use of the system facilities.

Participants: DP Managers, System Designers/Analysts, and Programmers.

Prerequisites: GI38 System/38 Fundamentals.

Objectives: On completion of this course participants should be able to:

- create application libraries and modify a jobs library list
- use the Source Entry Utility to enter and modify source statements
- use the Screen Design Aid to define, modify and test a display format
- use the Programmer's Menu
- write Control Language programs to control an application
- create files from the entered source statements
- create and maintain message queues and message files
- define and use device files
- use file overrides and the File and Program reference commands
- use the Copy File function
- understand the mechanics of Save/Restore and Journalling

Contents: Controlling the application- libraries and library lists; menus; message queues and files; security.
The Source Entry Utility- insert, change, delete, move, copy, scan and substitute functions; the Programmers Menu.
CL programs- required statements; communication with the workstation; program call and parameter passing.
The coding of physical and logical files in data descriptions- the Field Reference File; the use of keywords to define edit checks and edit codes; the definition in logical files of access paths to Physical Files; DFU; overriding files; copying files.
Display files- keywords in DDS; message facilities; Screen Design Aid.
Save/Restore- saving objects, libraries, and the system; Journalling.

CONTENTS

SYSTEM REVIEW AND BASIC CONCEPTS

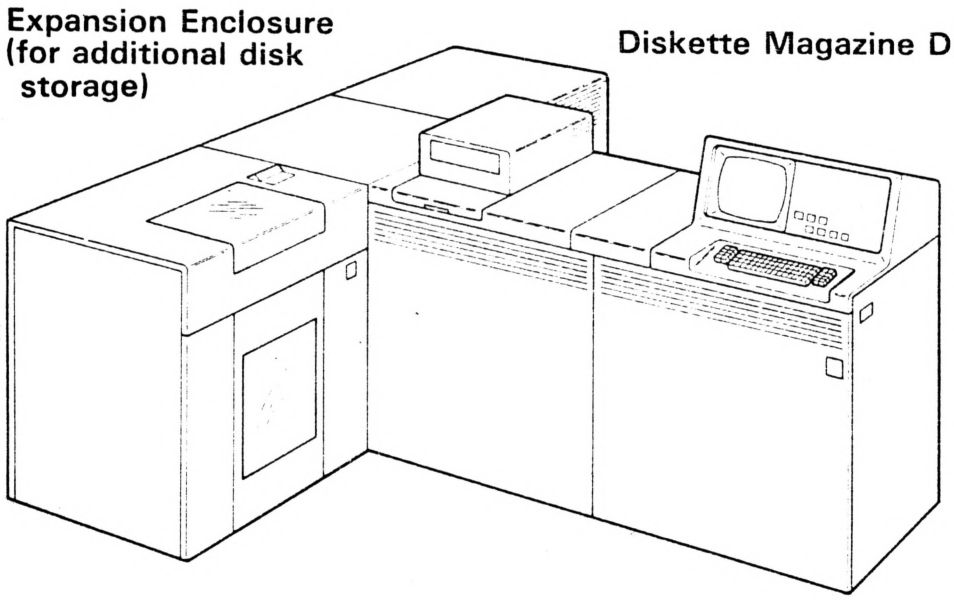
1. SYSTEM COMPONENTS

System Unit
I/O Devices

2. IMPORTANT CONCEPTS

Single level storage
Objects
Libraries

Exercise 1



SYSTEM UNIT

	Main Storage	CPU Cycle Time (nanosec)	Control Storage	C/Stg Cycle Time (nanosec)	Relative Internal Perform.
Model 4	1Mb - 2Mb	1100	8KW	200	1.0
Model 6	2Mb - 6Mb	400	8KW	133	1.6
Model 18	4Mb - 8Mb	400	12KW	133	2.6
Model 20	4Mb - 8Mb	400	12KW	67 (on 8KW)	3.3
Model 40	8Mb - 16Mb	333	12KW	67	4.3

Integrated auxiliary storage 64.5 Mb (min 1; max 6) - 387 Mb

IBM 3370 auxiliary storage 571.3 or 729.8 Mb
(max 8 on Models 18/20/40)- 5838 Mb

CE Console

I/O Devices

Diskette magazine drive (2 magazines, 3 single slots)
10 diskettes/magazine.
Workstation controllers - up to 8
Maximum local devices per controller - 32

Line Printers

4245	1200/2000 lpm	)	
3203	1200	")	
3262	650	")	maximum 2
5211	300	")	

Tape - up to 4 drives
3410/11 tape subsystem - speeds 20, 40 or 80 kbps.
3430 A01/B01 tape subsystem - 80 or 312 kbps (6250 bpi)
(with hardware Data Compression for Save/Restore)

Graphics 7371 2-pen Plotter
7372 6-pen colour Plotter

Communications
1 or 2 Communication Controllers + 3rd Comms Adapter
Up to 4 ports on each controller/adapter (12 lines)
A port can be used for either Binary Synchronous (BSC) or Synchronous Data Link Control (SDLC)
One port on each controller can be used for either:
- Local High Speed Attachment (56000 bps) to Series/1, System/34(36), 3705 or another System/38

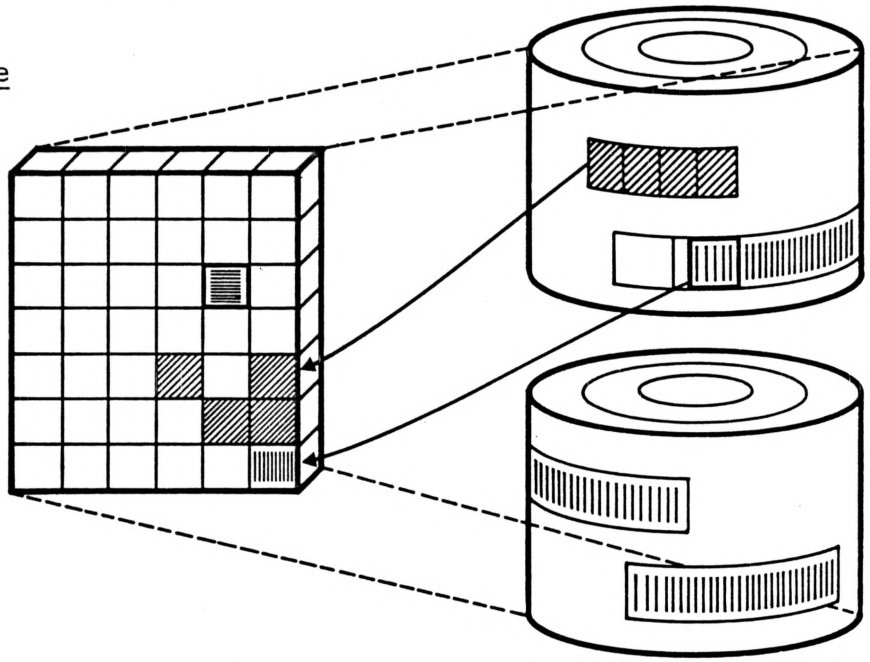
or:

- Remote High Speed Attachment (56000 bps) to 3705, 4331 Communication Adapter, or another System/38

3270 Remote Attachment/Device Emulation
EBCDIC and ASCII transmission codes
X.25 IXA Comms Attachment, giving 2 independent ports, each allowing for up to 32 'logical' connections
HCF/DHCF support.

IMPORTANT CONCEPTS

- Single level storage



System/38 uses an advanced form of virtual storage paging. This allows the system to use main storage efficiently for many processes simultaneously. Only the active (currently executing) pages of a program need to be in main storage at one time. The system pages in program or data pages as required, paging out inactive data pages before overwriting them with new pages.

System/38 pages all objects; data files, job queues, message files, and so on. These are brought to main storage from auxiliary storage when requested, in multiples of 512-byte pages.

When any object is created, the system gives it a unique virtual address, and stores this in a library. Then it translates this virtual address to a real location in auxiliary (disk) storage, and allocates this disk space to the object. The user doesn't specify a location when he creates the object or later when he asks for it to be retrieved. All auxiliary storage is automatically managed by the System.

Objects.

System/38 commands, high level languages and utilities operate on OBJECTS. Everything stored on the machine is an object of one sort or another, some being familiar, such as programs and files, but many are unique to S/38.

Properties of objects.

Broadly speaking, an object is made up of two components: a descriptive or attribute part and a functional (or data) part. The description contains the following information:

- Name
- Library is an index, not a predefined space. contains addresses of actual objects.
- Type
- Attribute (e.g. PF, LF)
- Owner (user profile when object is created) - can be changed
- Creation date/time
- Change date/time
- Date/time last saved offline
- Restore date/time
- Volume ID on the diskette
- Whether or not its storage was freed when saved
- Description
- Size

descriptive part
DSPOBJD
- name
- owner
- existence
- allowed operations

functional part
- PF: data records
- prog. - object code

The functional part of an object is, for example, the member in a file in which you can add, delete and modify records or, for a program, the executable code. You can often add and remove parts of the functional portion and change the attributes of an object. In such cases the appropriate CL command would be ADDxxx (eg. ADDPFM - Add a Physical File Member), RMVxxx (eg. Remove a Member) and CHGxxx for changing the attributes of an object (eg. CHGDSPF - Change a Display File).

There are currently 29 different object types presented by CPF, each of which may have the same name in the same library. There will not be any conflicts over naming because commands either imply or explicitly state the object type on which they operate.

Types of objects:

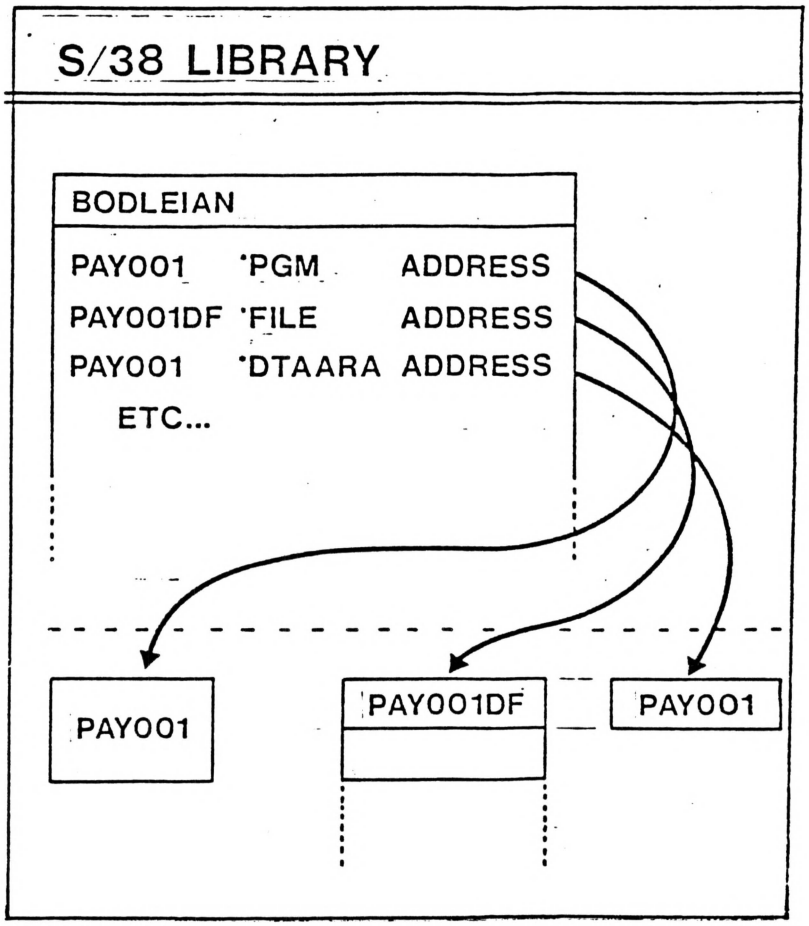
CHTFMT	Chart Format	CLS	Class
CMD	Command	CUD	Ctl Unit Description
DEVD	Device Description	DTAARA	Data Area
DTAQ	Data Queue	EDTD	Edit Description
FCT	Forms Ctl Table(RJE)	FILE	Most types of File
GSS	Graphics Symbol Set	JOBD	Job Description
JOBQ	Job Queue	JRN	Journal
JRNRCV	Journal Receiver	LIB	Library
LIND	Line Description	MSGF	Message File
MSGQ	Message Queue	OUTQ	Output Queue
PGM	Program	PRTIMG	Print Image (belt)
SAVF	Online 'Save-file'	SBSD	Subsystem Descrip'n
SPADCT	Spelling Aid Dictionary	SSND	Session Description
SYSRPYL	System Reply List	TBL	Table
USRPRF	User Profile		

Libraries.

A System/38 library, unlike libraries on other systems, is not a partitioned data set or 'container' for objects, rather it is an index to other objects. The effect, however, is that while objects can be referenced by the library in which they are 'stored', a library can never become 'full', since a library has no finite size (no object has). The reorganisation of a library is never required.

All objects are pointed to by libraries: an object cannot exist outside a library which makes backup and restore a great deal simpler: all objects pointed at by a library can be saved or restored with one command.

Similarly, the deletion of a library will delete all objects located through that library.



There are many ways of referring to an object:

Use of Libraries

REFERENCING AN OBJECT

– BY NAME ONLY
ORDERPGM

– BY NAME AND LIBRARY
ORDERPGM.ORDLIB

– BY GENERIC NAME
ORDER*

– BY GENERIC NAME AND LIBRARY NAME
ORDER*.ORDLIB

NO DUPLICATE OBJECT NAME OF THE SAME TYPE IN A GIVEN LIBRARY IS PERMITTED.

IBM supplies some libraries when the system is delivered.
First of these is QSYS, which:

contains many IBM-supplied programs and data (and libraries!);

cannot be deleted or renamed;

is owned by the System's Security Officer.

QGPL is the General Purpose Library. On delivery, it contains objects supplied for regular use, such as Source Files, Spool Queues and Subsystems.

It is by DEFAULT the library for all user-created objects;

It cannot be deleted or renamed;

It should be on every user's Library List.

interactive & batch

not accessible by other jobs

QTEMP is a library automatically created for <u>each</u> job on the system, uniquely identified with the job.

It lasts only for the duration of the job, whether interactive or batch, and only the job which caused it's creation can access it;

It is a useful place to store <u>temporary</u> objects such as scratch files and data areas required only within the job.

Other libraries provided by IBM are:

QSPL — Spooling Library - in which spool files are kept.

QGDDM — For Graphical Data Display Manager.

QSRV — Service Library for IBM Engineers.

QRCL — Reclaim Library.
Objects which have become detached from their library and objects which don't have an owner are placed in this library when the Reclaim Storage command (RCLSTG) is executed (in dedicated mode!). takes a couple of hours (loades of any object)
Such 'waifs' and 'orphans' are given to the Security Officer.
This library is only created if needed.

QRECOVERY — Recovery library. Intended only for System use.
following abnormal termination of system.

The following libraries exist if you have the products installed:

QRPG	RPG Library. contains compiler
QCBL	COBOL library.
QBAS	BASIC library.
QPL1	PL/1 library.
QIDU	Interactive Data Base Utilities (QUERY, DFU, SEU and SDA)
QS3E	Reformat Utility - for Conversion situations (sorting).
QTXT	Text Management.
QADM	Administrative Management.
QAPF	Advanced Printer Function.
QBGU	Business Graphics Utility.
QRJE	Remote Job Entry Facility.

User libraries. Can be created to suit your environment.

eg. for use per:
- application
- department
- user
- developer
- frequency of 'saving'

There are two types of library - Production (the default) and Test.

in debug mode to avoid updating list files.

PUBAUT
* none : no one else can access
* all : everybody can access
* normal : file can be used but not deleted or moved

Library Search Lists.

Most of the time you do not need to tell CPF about the location of an object: library search lists simplify locating the objects you refer to. A library list is an internal index (maintained for each job) that contains the names of libraries the system must search in order of occurrence to locate an object.

You can qualify an object's name by its library if you know where the object is or if you want an object from a particular library and you know an object of a given name and type exists in a library which occurs earlier in your library list. For example, a test version and a production version of a program.

Each job's library list is in two parts - the system part and the user part. The system part is available to all users of the system and the user part is unique to your job. The system part can contain up to 15 entries and the user part 25. The two parts together are known as *LIBL. Each part is initially derived from a system value - QSYSLIBL and QUSRLIBL.

QSYS is always searched any way

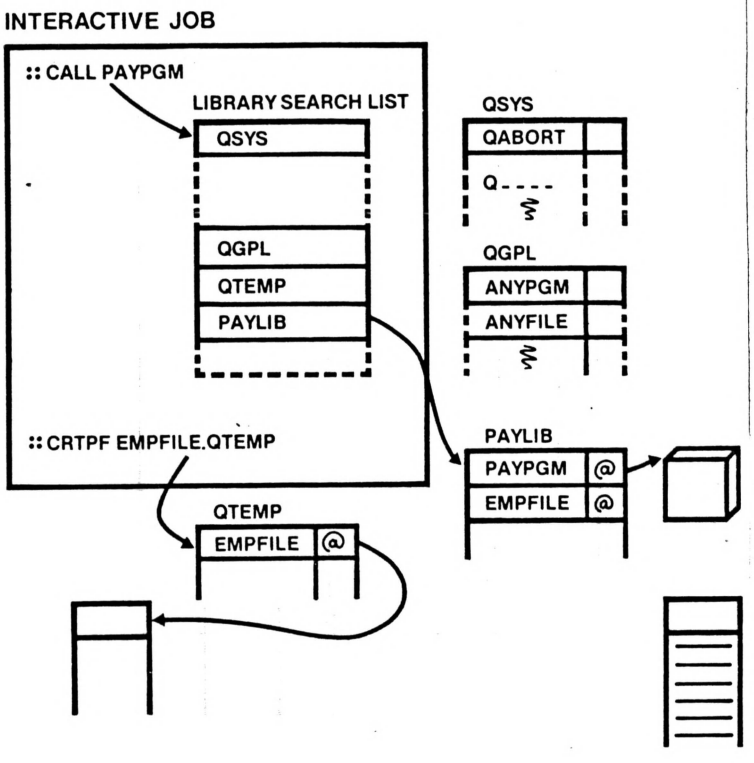
QSYSLIBL is a system value. Supplied as QSYS, but can contain up to 10 libraries.

QUSRLIBL is a sys. value supplied as QGPL QTEMP and can contain up to 15 libraries.

libraries are searched in seq. of library list.

job description contains characteristics of a job and include that jobs library list. INLLIBL

Libraries And the Library Search List



There is a command to change the System Library List in force for your current job (CHGSYSLIBL - see CL Reference Manual).

You can temporarily replace your job's User Library List with the RPLLIBL command. This command entirely replaces the list with the library/libraries you specify, until the job ends.
Alternatively, easier to use are the ADDLIBLE/RMVLIBLE commands.
You can display your job's library lists with the DSPLIBL command.

Sometimes a search may involve more than one object (generic search). On many commands, you can specify a System-recognised name in place of a library name as follows:

*LIBL - QSYSLIBL plus current user list (if in existence)

*USRLIBL - only the replaceable part of your job's library list (your job's version of QUSRLIBL)

*ALLUSR - all User libraries to which you are authorised are searched in alphabetical order. QGPL is searched but no other library begining with 'Q' is searched

*ALL - all libraries to which you are authorised are searched.

The last three values mentioned are only available on certain commands eg. DSPFD (Display File Description).

Remember that none of the normal 'CREATE' commands use a library list to determine the library in which the object is to be created.
The default library for every 'CREATE' command is QGPL (except for commands which create libraries themselves, or objects which are always stored in QSYS, such as user profiles and descriptions of lines, control units and devices).

(See also CPF Programmers Guide, pages 2-7 to 2-15 re Library Lists)

Job Descriptions and the Library List

When you initiate an interactive job by signing on at a workstation, that job's attributes are defined by a Job Description (object) associated with the workstation or user profile. These attributes include the Initial (User) Library List to be used for the job, the default value being (on Creation of a JOBD) the list in QUSRLIBL.

adds at head of list.

The RPLLIBL, ADDLIBLE or RMVLIBLE commands in a program called by the user, or keyed interactively, will change the job's user library list for the duration of the job or until another such command is executed.

When you sign on with your password on this course, an Initial Program named in your User Profile replaces your interactive job's library list so as to put the course library at the front of the list, with NO mention of your library.

Batch jobs will probably be set up to use the same library list throughout. Batch jobs are always associated with a job description, possibly named by the user when submitting the job, and not necessarily the same as the one used for the interactive job. The default JOBD used is QBATCH (one supplied) which uses the default library list (*SYSVAL).

In setting up a job description for a particular job, the Initial Library List parameter (INLLIBL) can be used to provide the correct library list, so that programs within the job will be able to run without the need to make changes to the list.

Your 'team' job description has been set up to provide Batch jobs which you submit, with the same library list that you have for your interactive job at your workstation.

Object Manipulation

The list below describes some commands associated with manipulating libraries and objects.

Create a Library	CRTLIB
Display a Library	DSPLIB
Delete a Library	DLTLIB
Clear a Library	CLRLIB
Replace Library List	RPLLIBL
Display Library List (in QTEMP)	DSPLIBL
Add Library List Entry	ADDLIBLE
Remove Library List Entry	RMVLIBLE
Create Object	CRTXXXXXX
Delete Object	DLTXXX
Check Object	CHKOBJ
Move Object (not generic)	MOVOBJ
Rename Object	RNMOBJ
Create Duplicate Object	CRTDUPOBJ

(This can make the setting up of a testing environment very easy - but see CPF Programmers Guide, pages 2-31/32).

Exercise 1

1. Create a library called SF38TEAMxx for the objects you will be creating during the rest of this course.

2. Move the job description SF38TEAMxx from SF38LIB library to your library.

3. Display your current library list and make a note of it.

4. Change the user part of your library list so that your new library is properly included.

5. Change the supplied job description SF38TEAMxx to the new library list. CHGJOBD SF38TEAMxx INLLIBL(.....). (This is for BATCH jobs)

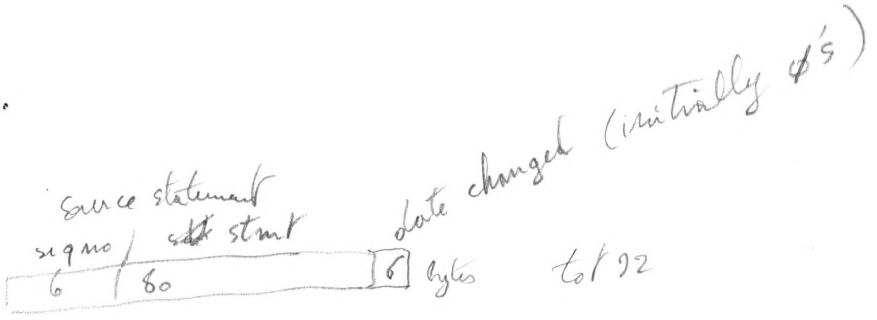
SF38TEAM07
~~QSYS~~
SF38LIB
QRPG
QIDU
QGPL
QTEMP

QSYS must not be included in user libl.

THE SOURCE ENTRY UTILITY

1. SOURCE ENTRY UTILITY
   - Source Files
   - Source Members
   - SEU Functions

   Exercise 2



system checking according to source file (RPG, CL, etc

Source Files

System/38 makes use of files to store source code. A physical file stores data records in members of files, a member being a logical grouping of records. Whilst the members are not physically partitioned in the file, it appears to the user of the file that the file consists only of records contained in the member being accessed at the time.

This concept is used widely on other systems, (S/3, S/32, S/34) where a library is a file containing many members, each of which is a named collection of records whether they be source program records or object program records. The S/38 however, only stores source records in a source file.

Source files on other systems (S/370, 4300, 8100) behave in the same way but are called Partitioned Data Sets.

It is preferable to use the standard names for your source files because the create commands use them as default parameter values. While it is perfectly possible to store all source code for all objects in one source file, you are strongly advised against it.
IBM supplies the following source files for your use:

same format but ≠ syntax checking

QCLSRC	To store Control Language program and job stream source
QDDSSRC	To store Data Description Specifications (Display Files, Physical and Logical Files etc.)
QRPGSRC	To store RPG program source
QCBLSRC	To store COBOL program source
QBASSRC	To store BASIC program source
QPL1SRC	To store PL1 program source

SEU Functions

SEU provides the following basic functions:

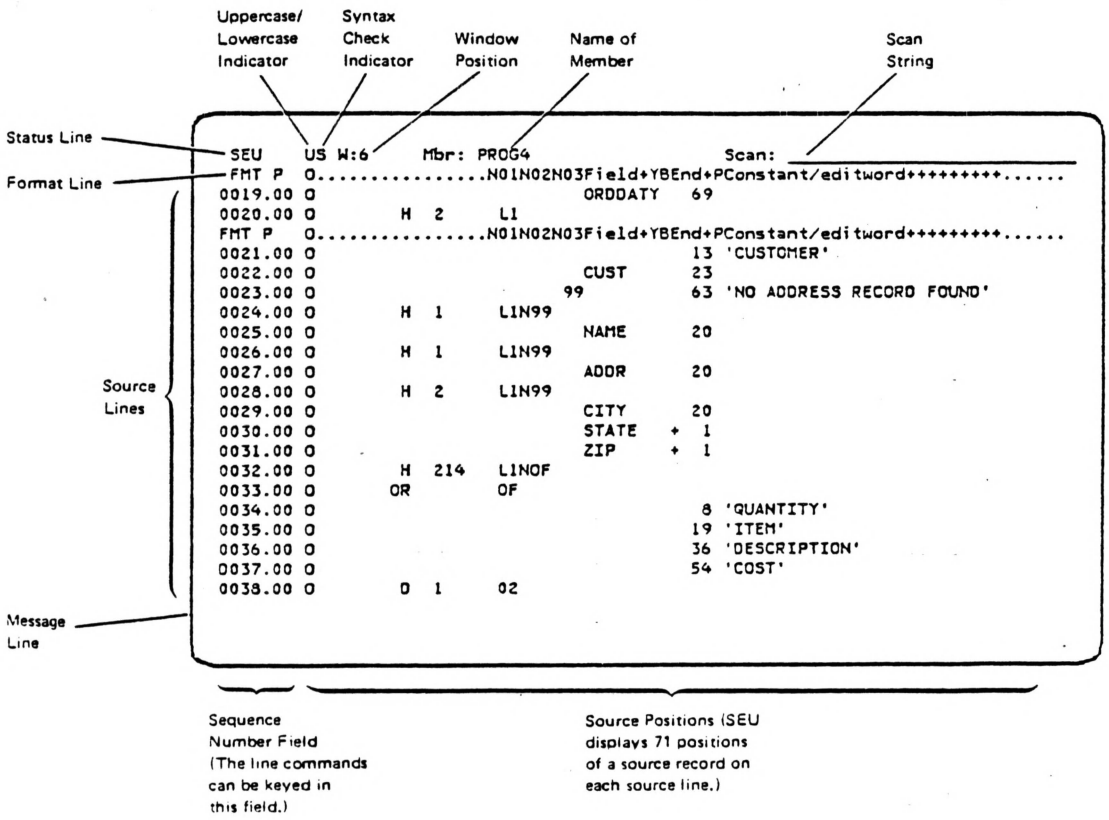
- INSERT, CHANGE, DELETE records. (=source statements)
- MOVE or COPY one or more records to a different location within the Source Member.
- BROWSE/COPY source from another Source Member.
- BROWSE a Spool File. (eg. a compilation listing)
- SCAN records for a specified character string and SUBSTITUTE a new value.
- SYNTAX checking as each line of code is entered.

In addition SEU will . . .

- Supply display formats for source language specification types.
- Provide a date of last update.
- Provide prompting support for entering control language commands.
- Provide an extensive HELP facility.

Edit Display

The most frequently used operations are performed by keying in Line Commands in the sequence number field of the edit display.



Line Commands

The line commands and their functions are shown below.

Line Command	Description
Insert	
I	Insert a blank line on which data can be keyed in. If you key in data on the blank line, SEU will insert another line to key in data.
In␣	Insert n blank lines on which data can be keyed in.
Positioning	
n␣	Position the record that has sequence number n as the first record on the display.
+n␣	Roll up n records.
-n␣	Roll down n records.
Prompting	
P	Place this line in a prompting section that has the same format as the line on which you keyed in the P.
Pff	Place this line in a prompting section that has specified format ff.
IP	Insert with prompting using the format of the line on which you keyed in the IP.
IPff	Insert with prompting using specified format ff.
Shift	
Ln␣	Shift data left n positions.
LLn␣	Shift data in a block of records (another LL defines the other boundary of the block) left n positions.
Rn␣	Shift data right n positions.
RRn␣	Shift data in a block of records (another RR defines the other boundary of the block) right n positions.
Skeleton Lines	
S	Define the contents of this line to be the skeleton line and remember the cursor position for later inserting of this skeleton line.
IS	Insert the current skeleton line and position the cursor to the remembered position.
ISn␣	Insert n skeleton lines and position the cursor to the remembered position.
Window	
Wn␣	Display data position n of the records in the first available position of the display.

Notes:

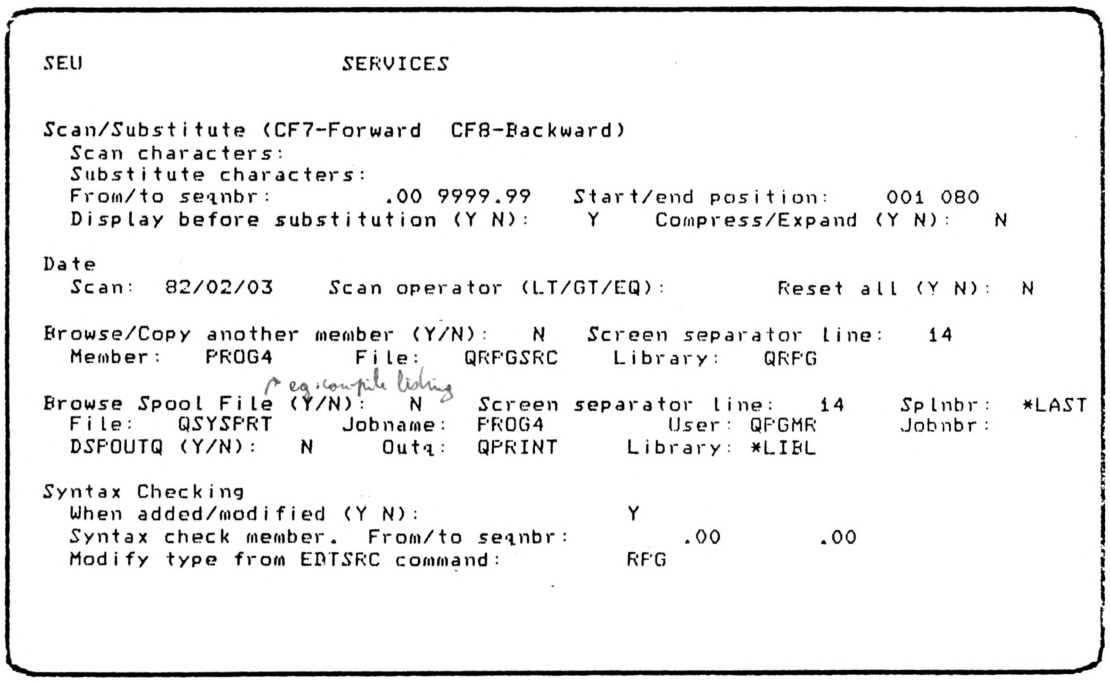
1. n is a number, which must be followed by a blank (␣).
2. ff is the two-position format identifier of a format.
3. The insert operation is a repeating operation, unless you specify n inserts.
4. For the shift line commands, if n is not specified, the default is to shift the data one position.

Line Command	Description
Copy and Move	
C	Copy a record to the target specified by A or B.
CC	Copy a block of records (another CC defines the other boundary of the block) to the target specified by A or B.
Cn␣	Copy n records, starting with this record, to the target specified by A or B.
M	Move a record to the target specified by A or B.
MM	Move a block of records (another MM defines the other boundary of the block) to the target specified by A or B.
Mn␣	Move n records, starting with this record, to the target specified by A or B.
A	Copy or move after (A) the line on which the A is keyed in.
B	Copy or move before (B) the line on which the B is keyed in.
An␣	Make n copies of the records copied or moved after (A) the line on which the An␣ is keyed in.
Bn␣	Make n copies of the records copied or moved before (B) the line on which the Bn␣ is keyed in.
Delete	
D	Delete a record.
DD	Delete a block of records (another DD defines the other boundary of the block).
Dn␣	Delete n records, starting with this record.
Format Lines	
F	Display a format line above the line on which you keyed in the F. The format line will have the same format as the line on which you keyed in the F.
Fff	Display a format line having specified format ff above the line on which you keyed in the Fff.
IF	Insert a blank line under a format line that has the format of the line on which you keyed in the IF. If you key in data on the blank line and press the Enter/Rec Adv key, SEU will insert another blank line to key in data.
IFn␣	Insert n blank lines under a format line that has the format of the line on which you keyed in the IFn␣.
IFff	Insert a blank line under a format line that has the specified format ff. If you key in data on the blank line and press the Enter/Rec Adv key, SEU will insert another blank line to key in data.
IFffn␣	Insert n blank lines under a format line that has specified format ff.
Increment	
Nn␣	Use n as the increment when assigning sequence numbers to inserted, moved, or copied records.

Services Display

The functions of Scan/Substitute, Browse/Copy another member, and the Syntax Checking control are accessed via the Services Display.

This is enabled by pressing CMD 5.

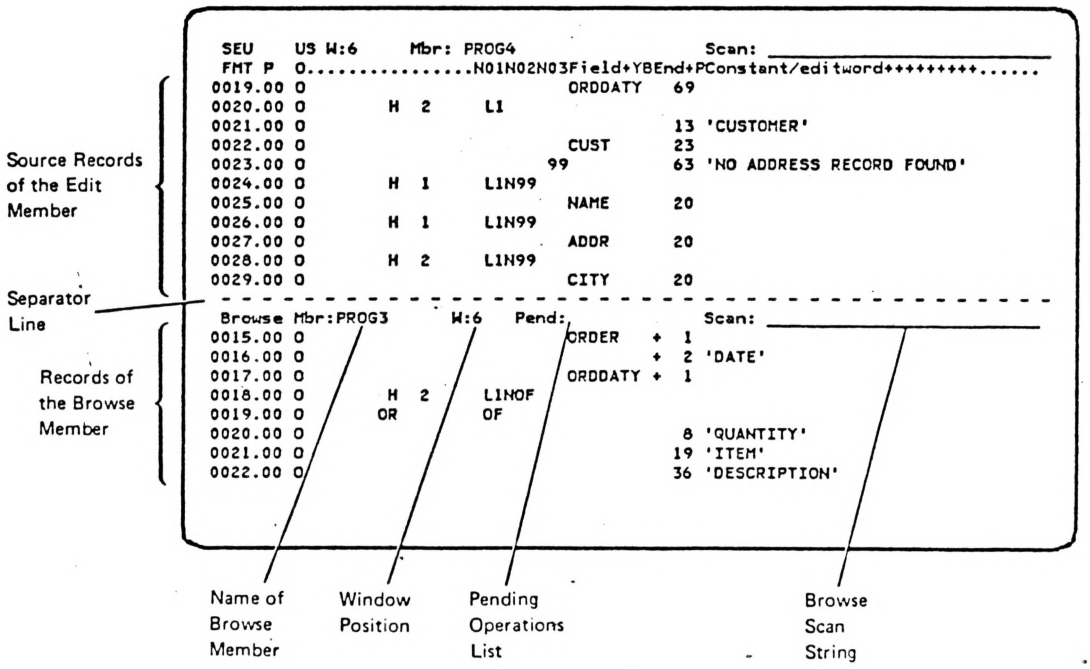


Split-Edit Display

The Browse/Copy Another Member function results in a Split-Edit screen as shown in the example below.

It is possible at this stage to Roll-up/Roll-down each individual member displayed.

One of more statements may be copied from the Browse Member to the Source Member being edited.



The Browse Member may be other than a Source Member (ie, A data base file).
In this case, the COPY function is not available.

HELP Functions

Throughout the SEU editing, the HELP key is available.

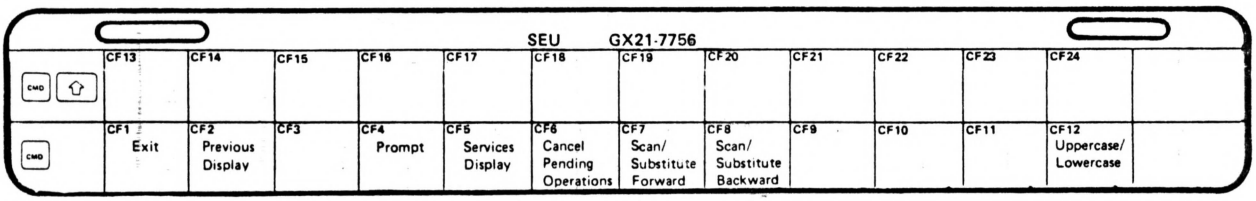
Pressing the HELP key causes the SEU HELP MENU to be displayed.

```
                              SEU HELP MENU                    Next Option: __
 1.  SEU Introduction                                          CF1 To Exit Help
 2.  Edit Display Introduction
 3.    D, Dn, DD                              - Delete
 4.    C, Cn, CC, M, Mn, MM, A, An, B, Bn     - Copy and Move
 5.    I, In                                  - Insert
 6.    F, Fff, IF, IFff, IFffn                - Format Lines and Inserts
 7.    S, IS, ISn                             - Skeleton Lines and Inserts
 8.    P, Pff, IP, IPff                       - Prompt Lines and Inserts
 9.    +n, -n, nnnn.nn                        - Record Positioning
10.    R, Rn, RR, RRn, L, Ln, LL, LLn         - Shifting Data Right or Left
11.    Wn                                     - Windowing (Sideways Viewing)
12.  Browse or Copy from Another Member
13.  Special Keys:  HOME, ROLL, HELP, DELETE, INSERT, PRINT, DUP
14.  EDTSRC Command Options and Member List Display
15.  Work Space Concepts and Recovery Display
16.  List of Prompt Formats and Source Types
17.  CF1  - Exit Display                  18.  CF5  - Services Display
19.  CF6  - Cancel Pending Operations     20.  CF7  - Scan or Substitute Forward
21.  CF8  - Scan or Substitute Backward   22.  CF12 - Switch Upper/Lower Case
```

Note: Option 16 from the HELP menu displays the available formats which may be used in the Line Commands.

COMMAND KEYS.

The SEU template shows the available command key functions.



Exit Display

The Exit Display allows several options and is accessed by pressing CMD 1.

```
SEU                    EXIT

Select one of the following:
  1. Exit without update
  2. Exit and update member
  3. Exit and create a new member
  4. Update member, no exit
  5. Create member, no exit
  6. Return to editing

Option: 2
                                      MEMBER      FILE        LIBRARY
For options 2 to 5:                   PROG4       QRPGSRC     QRPG
     Resequence member (Y N):         Y  Start:       1.00  Increment:  1.00

For options 1 to 3:
     Return to member list  (Y N):    N

For options 1 to 6:
     Print Source Listing  (Y N):     N

TOTAL RECORDS       ADDED      CHANGED       DELETED       SYNTAX ERRORS LEFT
      35              3           2             7
```

Initiating SEU

From the command Entry Display, enter EDTSRC and press CMD4. The following prompt will then be displayed.

```
                          Edit Source (EDTSRC) Prompt
Enter the following:
  File name:                              SRCFILE     __________
    Library name:                                       *LIBL
  Edit member:                            SRCMBR      *SELECT
  Source type:                            TYPE        *SAME
  Text:                                   TEXT        *BLANK
```

The SEU "Exit" display allows the printing of a Source Listing (the default is NO !).

In the Education Centre, output is not as a general rule printed out automatically.

To display all "Spool Files" waiting in an output queue, use the command: DSPOUTQ QPRINT
(or other queue name as directed).
This shows all outputs waiting at a point in time, not just source listings.
By keying a 6 in the field to the left of an entry, you can "release" for printing that particular output. Keying a 9 will make the item unavailable. Keying a 1 will enable you to display the CONTENTS of that spool file. Keying a 2 will enable you to display the ATTRIBUTES of that spool file, which can then be changed by the use of CMD key 3 in the subsequent display - for example, to change the number of copies you require, or to switch that spool file to a different output queue.

It can be aggravating to find many spool files in an output queue, with the need to roll-up (or down) many times (- a good reason to clear your entries by cancelling or printing frequently !).
There are, therefore, two alternative ways in which to display (in certain circumstances) just your own spool files:

Use the command DSPJOB, then take option 6 from the resulting menu. (You will see only those spool files generated since you LAST signed on.)

Use the command DSPSBMJOB, then take option 2 from the resulting display, for the particular batch job you previously submitted within the current period of sign-on.

DSPSBMJOB *WRKSTN (or DSPSBMJOB *USER) would give a display of your other batch jobs submitted prior to the current period of sign on.

Exercise 2

1. In your own team library create a Source File named QDDSSRC.

2. Create in this a new source member called NEWFRF, type *PF.
   Now use the BROWSE/COPY function to copy all of the member FLDREFFILE from the Source File QDDSSRC.SF38LIB to your member NEWFRF.

3. Using NEWFRF, use the various functions of SEU as follows:

   Position yourself at statement 84
   Change the field name from SINSTR to SHPINS
   Re-position back to the top of the member
   Insert a new 'comment' line between statements 1 and 2:
   FOR PRACTICE PURPOSES ON SF38
   Define statement 89 as a 'Skeleton'
   Insert this skeleton before statement 11 then change the field name to COMSTS
   Check the statement's date
   Scan for occurrences of LINE in the member, noting the numbers of the affected statements
   Delete these statements with the minimum of effort
   Use the 'Prompt' facility to insert after statement 8:
   BALANT 8 2 EDTCDE(K) COLHDG('BALANCES TOTAL')
   Duplicate statements 24 and 25 after statement 6, then re-name the field to ALTDES, add ALT before DESCRIPTION in the COLHDG brackets and change ITEM to ALTITEM in the TEXT brackets
   Delete statement 94
   Re-position statements 49 and 50 so that the field name can be changed to CUSNAM (retaining alphabetic sequence)
   Use the 'Format' facility to insert for the CUSNAM field an additional function CHECK(AB)
   Statement 7's COLHDG('BALANCE') should have only one blank after the previous entry
   Correct any outstanding syntax errors.

WORKSTATION DISPLAY FUNCTIONS

1. WORKSTATIONS
   - Display screens, Keyboards and Printers
   - Display Formats
   - Describing display formats - DDS

2. DATA VALIDATION
   - Validity checking
   - Error handling

3. DISPLAY DEVICE FILES
   - Creating the display device file

Exercise 3

WORKSTATIONS

System/38 supports local and remote workstation displays and printers for doing work on the system, and in addition, Personal Computers.

Displays

PC, PC/XT and PC/AT

5251 model 11 - 24 lines x 80 characters (1920 characters)
model 12 - remote cluster controller (also 24 x 80)

5291 - 24 lines x 80 characters
- selectable block cursor

5292 - seven-colour display station
- 24 lines x 80 characters

3180 model 2 - 24 lines x 80 characters
- 192 displayable char. set, Operator selectable

Indications of: Message waiting/System available/Keyboard shift/ Insert mode/Input inhibited (on line 25 of 529X displays)

Keyboards: - Typewriter or (5251 display only) Data Entry
- Slimline - 529X displays
- New slimline 122-key modifiable (3180 model 2)

Matrix and line printers

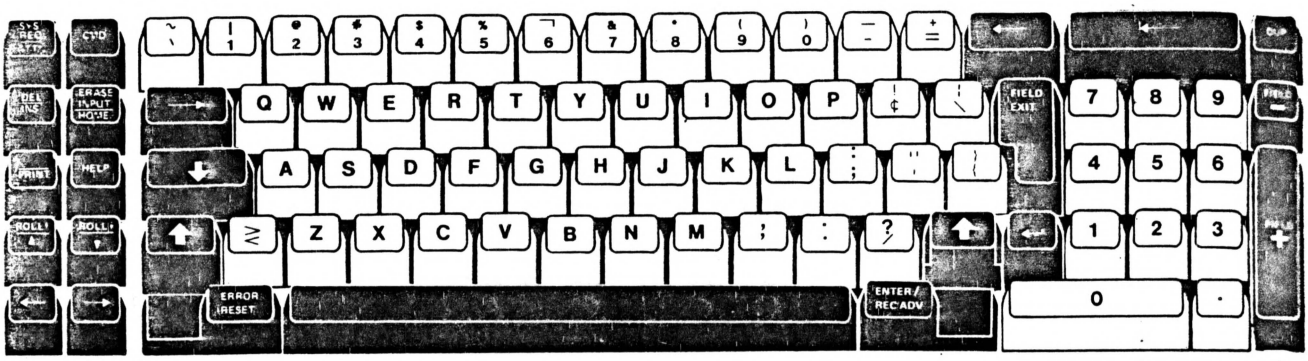
5256 - matrix printer (models 1, 2 or 3)

5224 - desk-top 175 lpm printer

5225 - floor-standing 560 lpm printer

5219 - printwheel printer for Text quality print
- models D01 or D02; take A4 size paper

4214 - good for graphics applications



display file contains up to 1024 formats, each requiring a unique name
to a single member

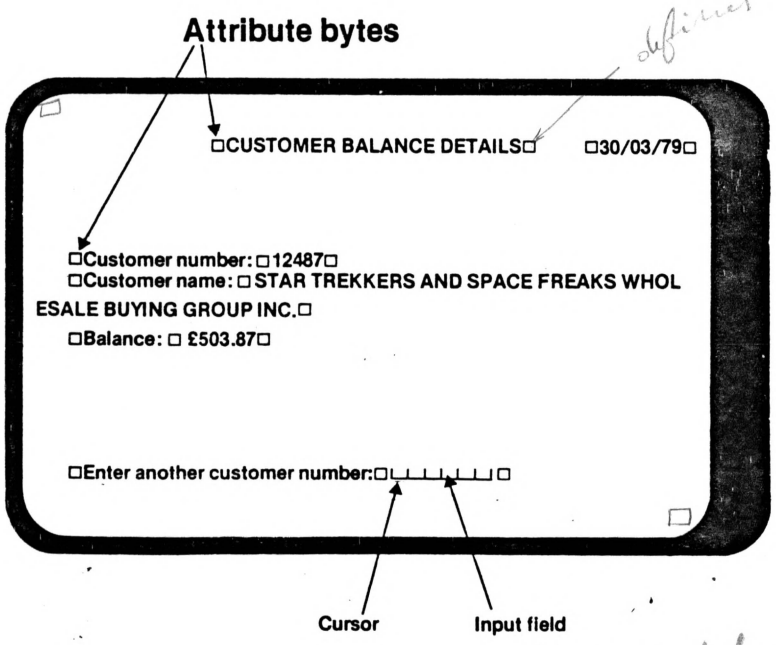
Data is entered from the keyboard into a buffer in the display and simultaneously shown on the screen. When a complete record has been entered, the operator presses the Enter or a Command key. This signals to the System/38 that the display is ready to transmit the entered data. It also prevents the operator keying any more at the keyboard until the system responds.

Command Keys:

Twelve of the topmost keys in conjunction with the 'CMD' key can behave as command keys to signal special functions to the program. A programmer-defined indicator is sent back to the program when a command key is pressed. This indicator is followed by any entered input field data if the key was defined as a command function (CF) key. If the key was described as a command attention (CA) key, no data is passed back, even if entered. The command keys are usually associated (by the programmer when he describes a screen record format) with special functions (like 'cancel' or 'Go to next program").

o Display formats

Data is displayed and entered on the 5250 screen as fields. A field is defined on the screen with an attribute byte which describes how the field is to be displayed. This attribute byte takes up one character position on the screen immediately before the field. The field can wrap round from one line on the display to the next below - but not from the bottom of the display to the top. The field continues to the next attribute byte.



defines next field of blanks

message field default at line 24

enable command key 8 as command attention (CF08)

CA 08

CA 08(08)

↑ indicator: use same as key number (01→24)

when CF8 is pressed, indicator 08 comes on.

CF 09(09) : command function key

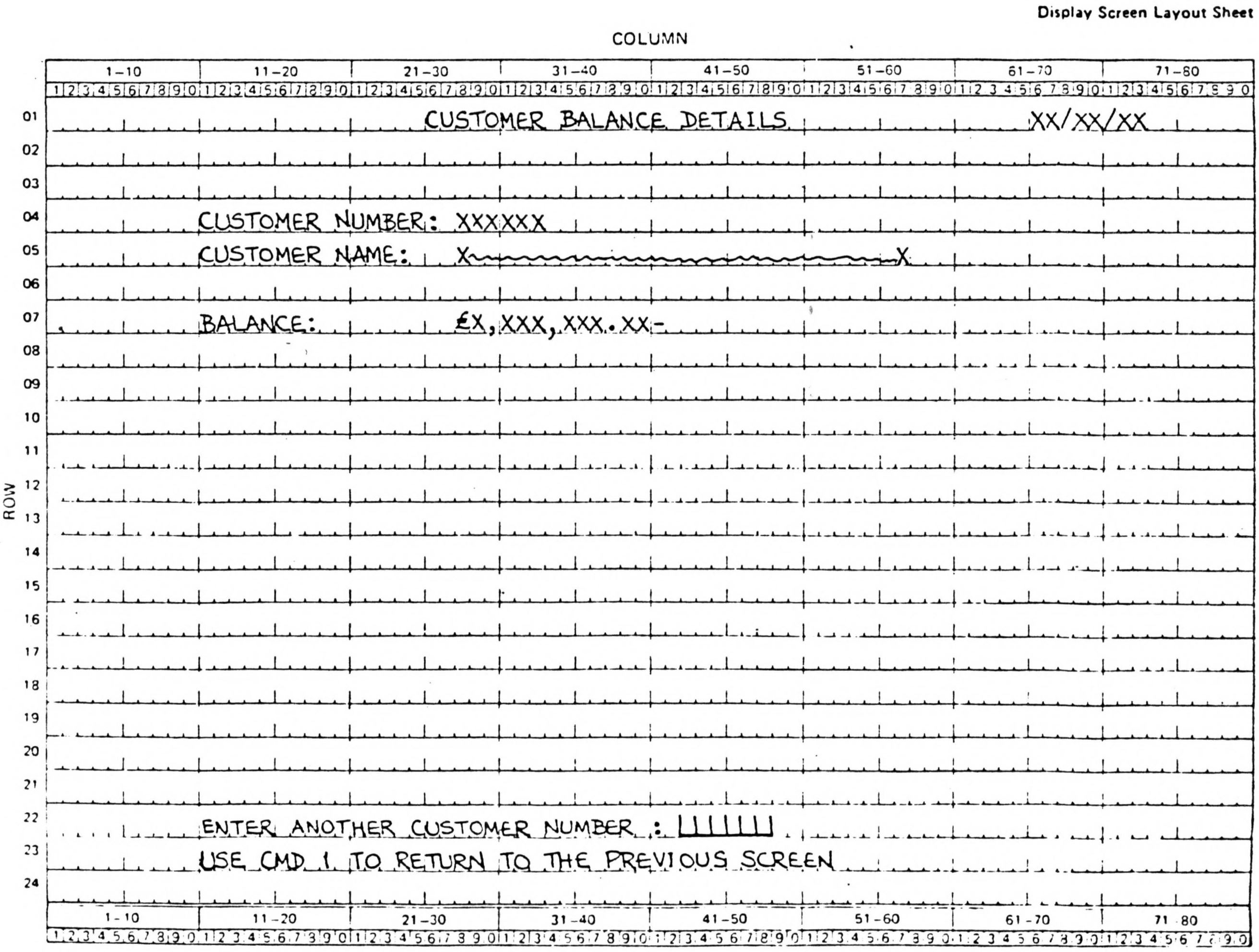
CA does not transfer input capable fields contents back to the program, whereas CF keys do.

see p. 3.3 note

command key can be enabled at file level or at format level

o <u>Describing display record formats - DDS</u>

The Data Description Specifications form is used to describe screen record formats in a Display Device File to the System. It is also used to describe data to the System for data base files, for printer device files, and for communications files. (It is not an RPG III coding sheet!)
Use a Display Screen Layout form to design the record format.



CF16: if you want to execute a command without going through all the prompts

Data Description Specifications

File: CUSBALD

Programmer: Date:

```
     A*
     A*CUSTOMER BALANCE INQUIRY FORMAT
     A*
     A                                      REF(FLDREF)
     A          R CUSBALF                   TEXT('CUSTOMER BALANCE INQUIRY')
     A                                      CA01(01 'END OF INQUIRY')
     A                                  1 26'CUSTOMER BALANCE DETAILS'
     A                                      DSPATR(RI HI)
     A                                  1 66DATE EDTCDE(Y)
     A                                  4 11'CUSTOMER NUMBER:'
     A            CUSTNO    R           4 28
     A                                  5 11'CUSTOMER NAME:'
     A            CUSNAM    R           5 28
     A                                  7 11'BALANCE:'
     A            CURBAL    R           7 28
     A                                 22 11'ENTER ANOTHER CUSTOMER NUMBER :'
     A            CUSNUM         6  0I 22 43DSPATR(CS)
     A  90                                  ERRMSG('NO SUCH CUSTOMER EXISTS' 90)
     A                                 23 11'USE CMD 1 TO RETURN TO THE PREVIOU+
     A                                      S SCREEN'
```

REF	a File-level keyword naming a Field Ref. File.
R (Record Format)	same naming rules as for Data Base Files.
CA01	how a Command Attention key is enabled and an Indicator associated with it (No. in brackets).
DSPATR	Display Attributes - how the field is to appear.
DATE	the job's date is used. (TIME is also available).
EDTCDE	RPG edit codes are used (see DDS Ref Manual 4-119).
R (Reference)	all input and output field definitions are taken from the physical data base file (FLDREF in the illustration).
ERRMSG	this message is displayed only if indicator 90 is on when the record format is sent to the screen. The system will then turn indicator 90 off.

SECLVL keyword: enables help key and 2nd level msg.

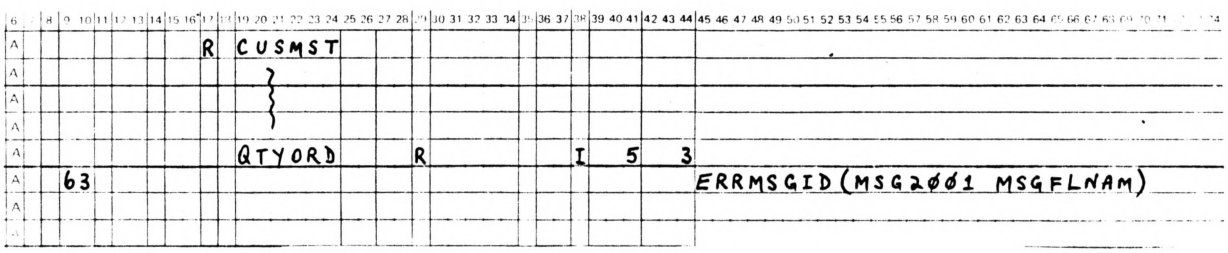
Whilst the ERRMSG keyword allows the sending of an impromptu message, the ERRMSGID keyword allows for pre-defined messages.

This is a good way to establish a set of common messages for an application.

ERRMSGID names a specific Message Identifier and the Message File in which it is included. There is no provision for the System to turn off an indicator governing such an error message - this must be done in the program.

These messages can be defined independent of programs and DDS, can have 2nd level text, and can easily be changed (eg. for the message to be displayed in another language) without change to programs or device files.

Message Files and the inclusion of messages within them will be discussed later.



Display attributes (DSPATR) can specify:

CS	Column Separators	HI	High intensity
UL	Underline	BL	Blink
RI	Reverse image	PR	Protect
ND	Non display	PC	Position cursor

(RI/HI/UL is NOT a good combination!)

COLOR can specify:

GRN	green	WHT	white
RED	red	TRQ	turquoise
YLW	yellow	PNK	pink
BLU	blue		

Existing formats will produce colour without re-programming:

	ATTRIBUTE		
	H/Intensity	C/Separator	Blink
Green	.	.	.
White	x	.	.
Turquoise	.	x	.
Yellow	x	x	.
Red	.	.	x
Red Blink	x	.	x
Pink	.	x	x
Blue	x	x	x

(Underscore and column separator are also blue).

DATA VALIDATION

Validity checking by Data Type

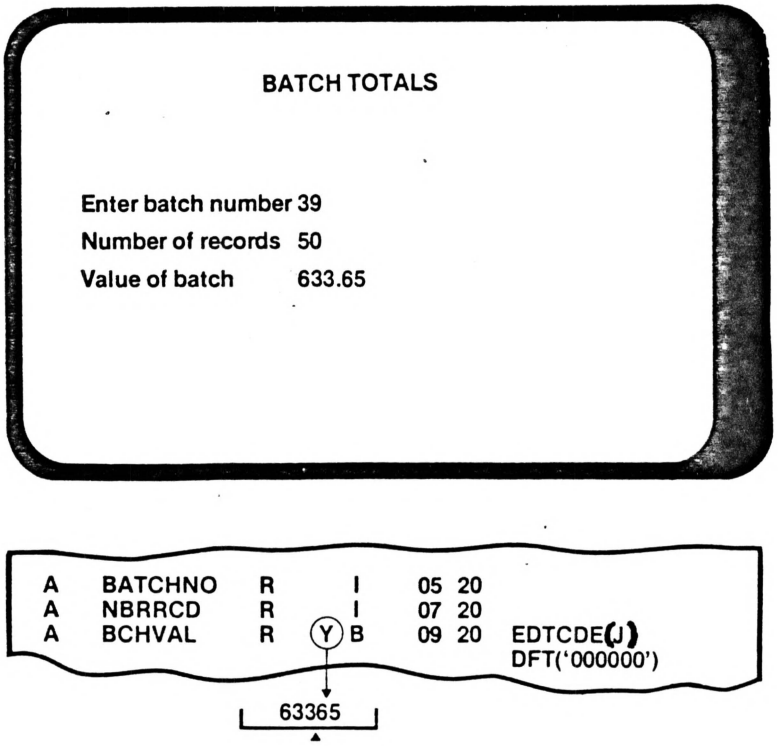
The Display Device Support provides for checks on valid data in fields defined as alpha-only or numeric only in col. 35 of the DDS.

X - Alpha: only A-Z a-z,.-⌀ are acceptable.

S - Signed numeric: only 0-9 are acceptable.
One extra position of the field is used for the sign, seen as - if negative.
This is the DEFAULT if col. 35 is left blank!

Y - Unsigned numeric: only 0-9 +-.,⌀ are acceptable.
Data is decimal aligned on input; editing characters are removed. A negative field is displayed as (eg. 633.6N) if the operator indicates 'negative' by using the Field Exit Minus (Field -) key.

(See also DDS Reference Manual 4-17 to 4-27)



Validity checking by keyword

Validity checks can be picked up from a data base physical file (REF) or specified directly in the display record format DDS.

CHECK(ME)	operator must enter a least one character in this field
CHECK(MF)	operator must fill the field if any characters are keyed
RANGE	data must be in the range given
VALUES	data must be one of the (up to 100) values given
CMP (or COMP)	data must compare (EQ,GT,NL,etc) with a constant
CHECK(M10)	(the data entered must be a self-check number)
CHECK(M11)	(see DDS Manual for details of algorithms used)
CHECK(VN)	the field must be a valid CPF name (begins A-Z £ # @)

sys.val.

For a full discussion of all possible keywords, refer to the DDS Reference Manual, Chapter 4.

Error handling

When the system detects a validity error:

the keyboard locks;

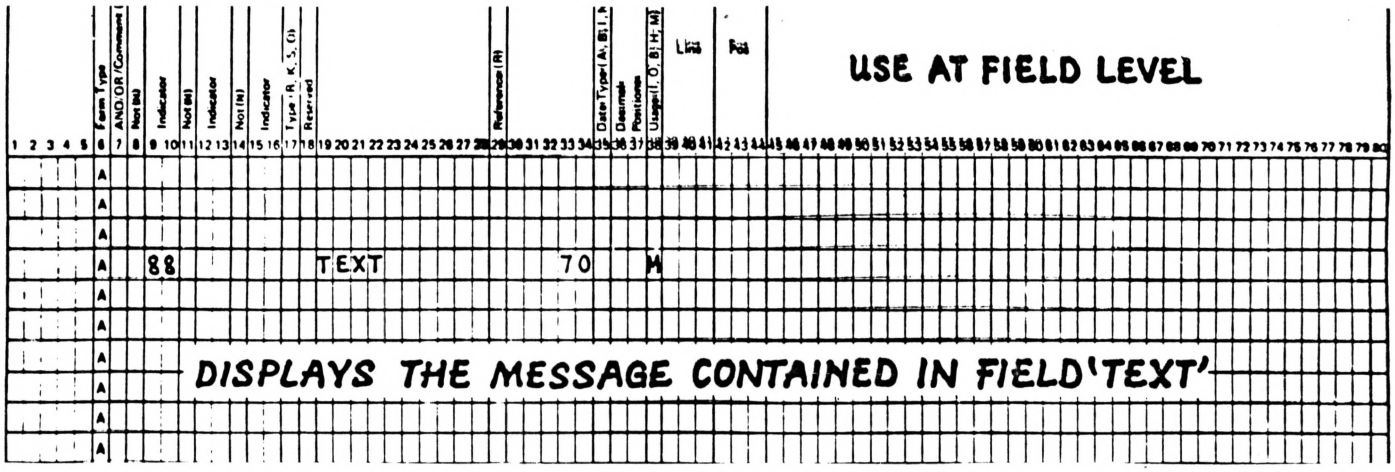
an error message (number) is displayed in line 24;
if the HELP key is now pressed, the screen contents are saved and a 2nd level message (in words) is displayed.

the field in error is shown in reverse image, highlighted (if DSPATR(UL or HI) was not specified), and the cursor (blinking) is positioned under the field.

The operator must press the RESET key to try again.

The ERRMSG (and ERRMSGID) keywords cause User-defined messages to be treated in the same way.

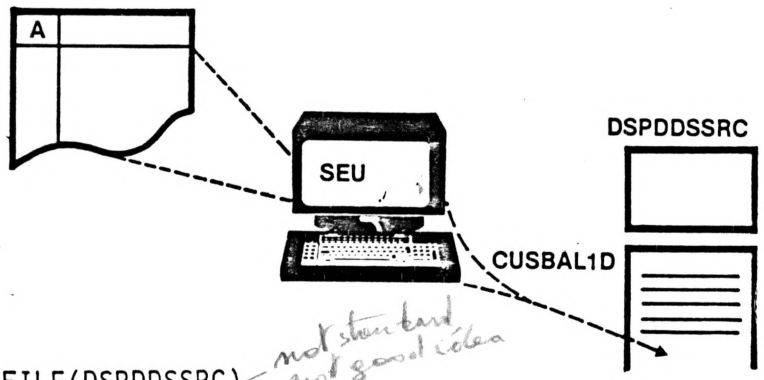
An alternative to ERRMSG/ERRMSGID is to define a field as a 'Message' field ('M' in col. 38 on DDS) optioned with an indicator.
This is a way of displaying a message at any time - not just when there are errors.



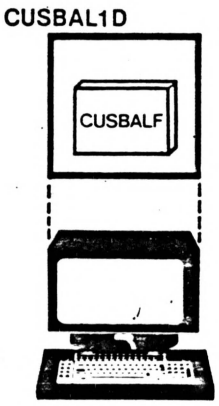
DISPLAY DEVICE FILES

Creating the display device file

Screen record formats are stored in display device files
Generally, a device file will contain all the screen record formats to be used by a particular program or application.




```
::EDTSRC  SRCFILE(DSPDDSSRC)
          SRCMBR(CUSBAL1D)
          TYPE(*DSPF)
          TEXT('Customer Balance Display')
::CRTDSPF CUSBAL1D
          SRCFILE(DSPDDSSRC)
          TEXT('Customer Balance Display')
```



There is an IBM-supplied "Programmer Menu" which (via option 3) allows NON-INTERACTIVE creation of objects including Display Files. (See Appendix A)

```
CALL QPGMMENU
```

```
5714SS1 R03 M00 811204                  DATA DESCRIPTION                CUSBALD.IC38LIB        18/05/82

File name -          CUSBALD.IC38LIB    Type of file -     DISPLAY
Source file -        QDDSSRC.IC38LIB    Member -           CUSBALD      06/01/81  10:05:07
Type of data -       *DATA
Options -            *SRC    *LIST
Authority -          *NORMAL
Text -               Customer balance display file for IC38
Compiler -           IBM System/38 Data Description Processor

                                    DATA DESCRIPTION SOURCE

SEQNBR  *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ... ... 8  DATE

   100       A*
   200       A*CUSTOMER BALANCE INQUIRY FORMAT
   300       A*
   400       A                                        REF(FLDREF)
   500       A          R CUSBALF                     TEXT('CUSTOMER BALANCE INQUIRY')
   600       A                                        CA01(01 'END OF INQUIRY')
   700       A                                      1 26'CUSTOMER BALANCE DETAILS'
   800       A                                        DSPATR(RI HI)
   900       A                                      1 66DATE EDTCDE(Y)
  1000       A                                      4 11'CUSTOMER NUMBER:'
  1100       A            CUSTNO    R               4 28
  1200       A                                      5 11'CUSTOMER NAME:'
  1300       A            CUSNAM    R               5 28
  1400       A                                      7 11'BALANCE:'
  1500       A            CURBAL    R               7 28
  1600       A                                     22 11'ENTER ANOTHER CUSTOMER NUMBER :'
  1700       A            CUSNUM         6  0I 22 43DSPATR(CS)
  1800       A  90                                    ERRMSG('NO SUCH CUSTOMER EXISTS' 90)
  1900       A                                     23 11'USE CMD 1 TO RETURN TO THE PREVIOU+
  2000       A                                        S SCREEN'

                       *****  E N D   O F   S O U R C E  *****
```

```
5714SS1 R03 M00 811204              DATA DESCRIPTION              CUSBALD.IC38LIB           18/05/82

                                     EXPANDED SOURCE
                                                                                         FIELD  BUFFER P
SEQNBR  *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ... ... 8  LEN    OUT

   400
                                            DSPSIZ(*DS3)
          * OPTION INDICATOR OUTPUT BUFFER POSITIONS -
          *  *IN90 0001
          * RESPONSE INDICATOR INPUT BUFFER POSITIONS -
          *  *IN01 0001   *IN90 0002
   500               R CUSBALF                  TEXT('CUSTOMER BALANCE INQUIRY') CA+
   600                                          01(01 'END OF INQUIRY')
   700                                       1 26'CUSTOMER BALANCE DETAILS' DSPATR(R+    24
   800                                          I HI)
   900                                       1 66DATE EDTCDE(Y)                           8
  1000                                       4 11'CUSTOMER NUMBER:'                      16
  1100                 CUSTNO        6S 00   4 28                                          6       2
          * Ref: Fld - CUSTNO      Fmt - REFFMT      File - FLDREF      Lib - IC38LIB
  1100                                          TEXT('CUSTOMER NUMBER               -
  1100                                                                ')
  1200                                       5 11'CUSTOMER NAME:'                        14
  1300                 CUSNAM       30A  0   5 28                                         30       8
          * Ref: Fld - CUSNAM      Fmt - REFFMT      File - FLDREF      Lib - IC38LIB
  1300                                          TEXT('CUSTOMER NAME                 -
  1300                                                                ')
  1400                                       7 11'BALANCE:'                               8
  1500                 CURBAL        9Y 20   7 28                                          9      38
          * Ref: Fld - CURBAL      Fmt - REFFMT      File - FLDREF      Lib - IC38LIB
  1500                                          TEXT('CURRENT BALANCE               -
  1500                                                                ') EDTCDE(J £)
  1500
  1600                                      22 11'ENTER ANOTHER CUSTOMER NUMBER :'       31
  1700                 CUSNUM        6S OI 22 43DSPATR(CS)                                 6
  1800       90                                 ERRMSG('NO SUCH CUSTOMER EXISTS' 90)
  1800
  1900                                      23 11'USE CMD 1 TO RETURN TO THE PREVIOU+    42
  1900                                          S SCREEN'

            * * * * *   E N D   O F   E X P A N D E D   S O U R C E   * * * * *
```

Exercise 3

The object of this exercise is to code and create the menu shown below.

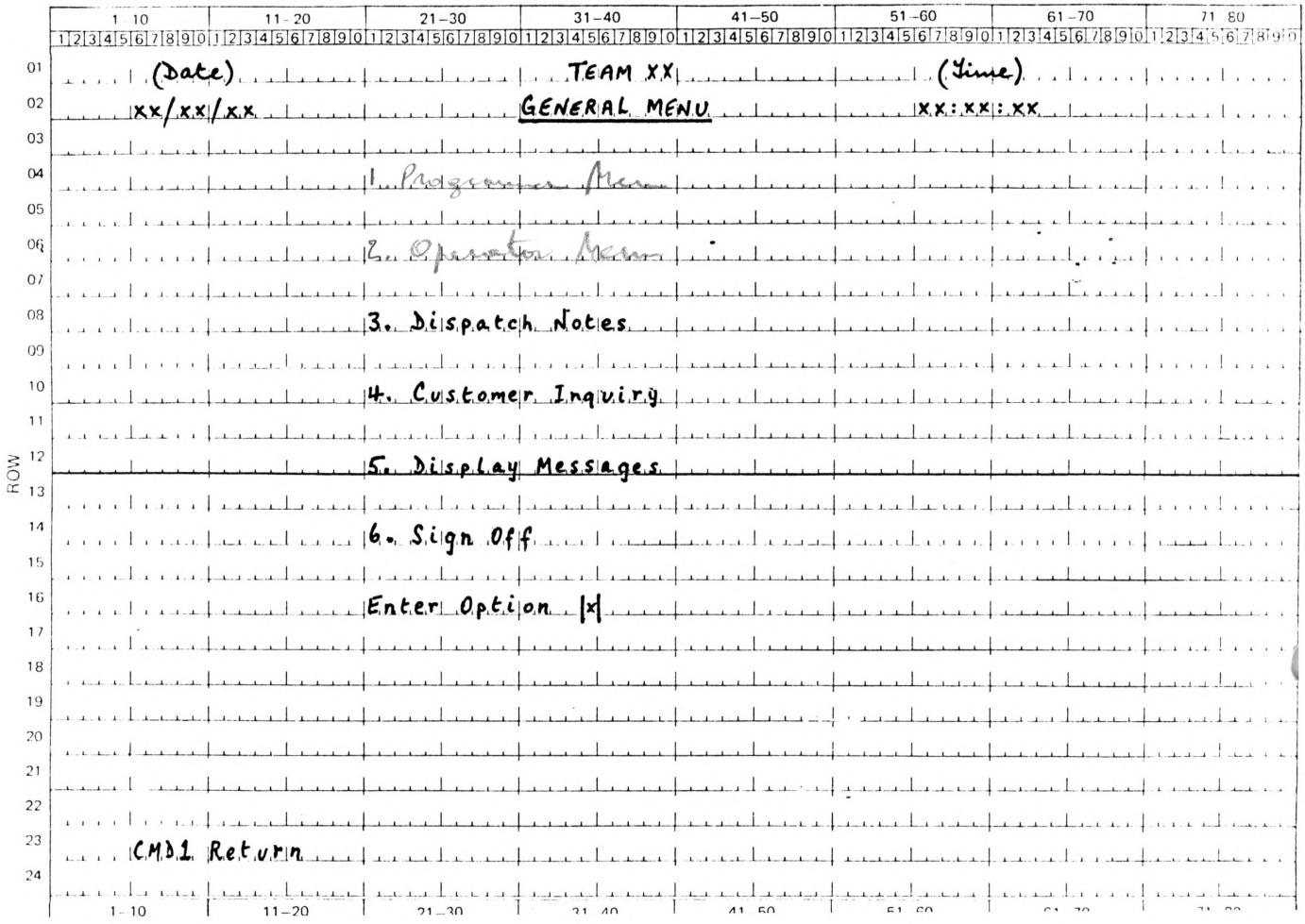
= member name

1. The record format is called SFMENU and the display-file SFDSPF. The source should be stored as a member in QDDSSRC with the same name as the object that you intend to create.

2. Notice the lower-case. You will need to create the fields DATE, TIME and OPTION (the selection field). Ensure that the display file will prevent the operator from entering invalid option numbers by specifying a RANGE check on the OPTION field. Enable Command Key 1 in such a way that any previously chosen option will be ignored by the program.

3. Create the display file.

4. Test, by CALL-ing a CL program named SFMENU.



CONTROL LANGUAGE AND CONTROL LANGUAGE PROGRAMMING.

CL Syntax
Interactive command entry and prompting
CL Programs
Executable commands
Commands used with the Declared Display File

Exercise 4

CALL/TFRCTL/RETURN
Built-in Functions
Data Areas
Batched CL

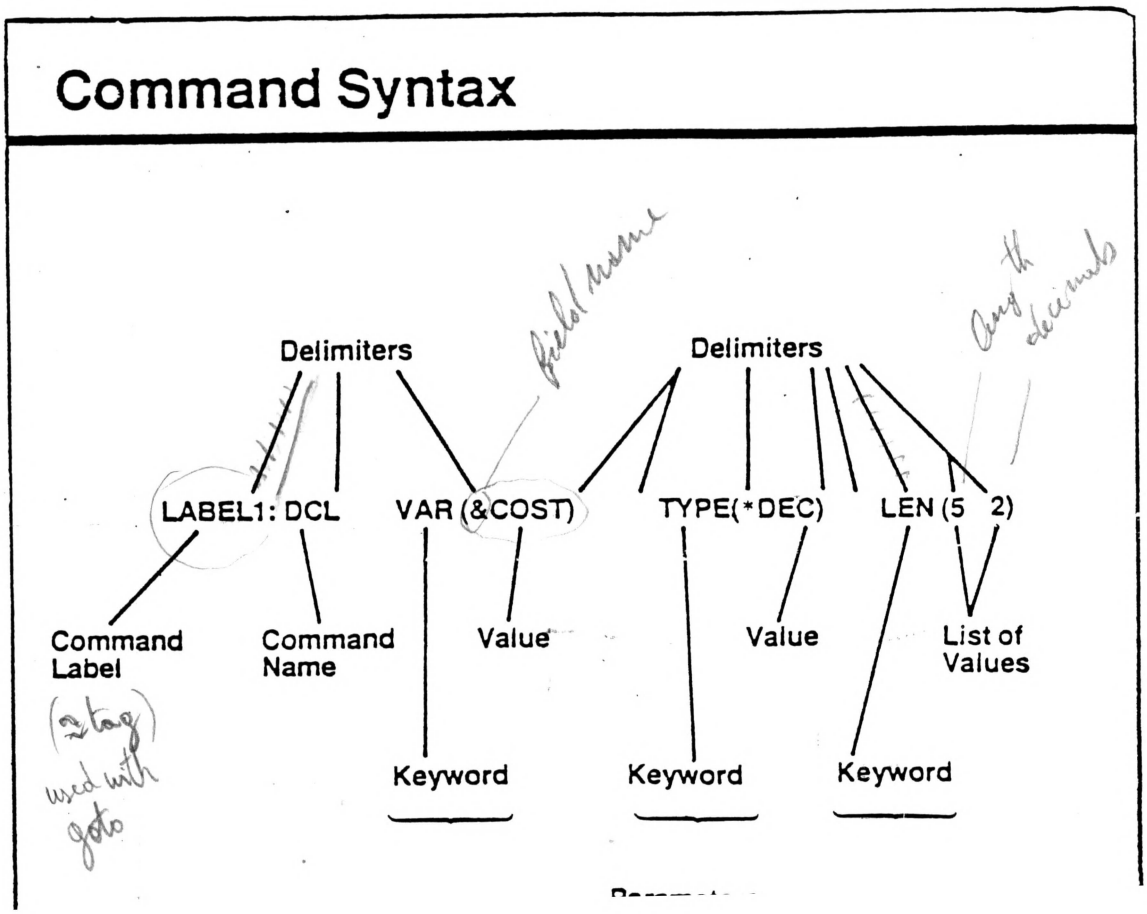
Exercise 5

Control Language.

CL is the interface between CPF and the user of the System/38. All functions that the S/38 supports are accessible to the user via CL. High Level Language programs (RPG, COBOL) would be used to solve the data processing problems of an installation. CL as a language is not able to read, update, write or delete records in a data base file in the same way as RPG or COBOL. It is able to process files and file members through the Copy File command, and other commands which display file information etc, or as sequential input.

CL Syntax

Control language adheres to a general naming convention that makes remembering the commands very easy. The first three characters form a 'verb' phrase and the remaining (up to seven) characters a 'noun' phrase that determines the operation of the command. For instance, all commands that create objects will begin with CRT, all commands that display information begin with DSP and so on. These commands are free-form with one or more blanks delimiting the keywords:



Commands expect their parameters in a particular order, a keyword defining each parameter. The 'Create Library' command, for example, specified in full, looks like this:

CRTLIB LIB(BODLEIAN) TYPE(*PROD) PUBAUT(*NORMAL) TEXT(*BLANK)

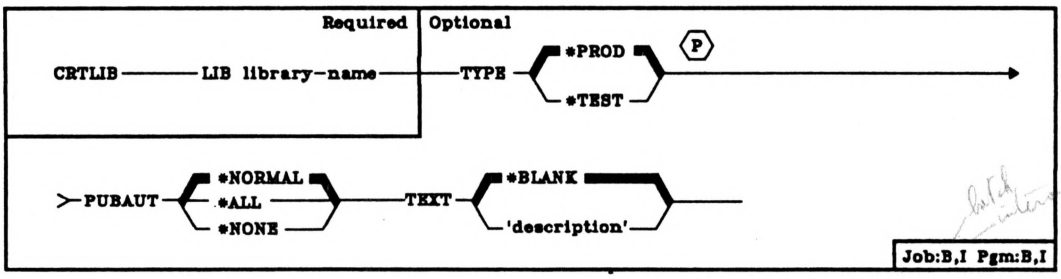
There are two ways of entering parameters on commands:

1. Entering parameters in the order expected (ie. positionally), there is no need to enter the keywords up to a certain point:

CRTLIB BODLEIAN *PROD PUBAUT(*NORMAL) TEXT(*BLANK)

Notice the absence of brackets when the keyword is not used. Brackets are only used without the keyword when a parameter contains more than one value - ie. a list.

The system will assume a DEFAULT value for each parameter you choose not to enter:



The only required parameter in this example is the Library name. A thick black line indicates the default value assumed if the parameter is omitted. You cannot code positionally beyond the symbol P shown in the syntax diagram - ie. keywords must be specified.

2. Supplying some parameters and not others because the defaults are acceptable - you must specify the keyword if it appears out of order. Parameters may be given in any order:

```
CRTLIB TEXT('Master application library') LIB(BODLEIAN)
```

Alternatively, replace the missing parameters with the special value *N in order to maintain positionality and dispense with keywords prior to any positional limits:

```
CRTLIB  BODLEIAN  *N  *N  TEXT('Master application library')
```

for position

Some keywords require a list of values (more than one parameter value per keyword). Brackets must enclose the list to enable the list to be regarded as a set of values for ONE keyword rather than values for many keywords. For example:

```
DCL  VAR(&FRED)  TYPE(*DEC)  LEN(5 2)  VALUE(22222)
```

becomes:

```
DCL  &FRED  *DEC  (5 2)  22222
```

Interactive Command Entry.

CL commands can be entered at the workstation through the Command Entry display. As there are in excess of 300 commands available, some having over 25 parameters, a Command Grouping Menu is available to simplify the entering of commands. This menu is obtained by pressing Command Key 4 or by entering a '?' in the command entry display and pressing the enter key:

```
CMDGRP              COMMAND GROUPING MENU
SELECT ONE OF THE FOLLOWING:
   1. ALL COMMANDS MENU                          ALLCMD
   2. VERB MENU                                  VERB
   3. SUBJECT MENU                               SUBJECT
   4. OBJECT MANAGEMENT MENU                     OBJ
   5. DATA BASE FILE MENU                        DBF
   6. FILE MENU                                  FILE
   7. DEVICE MANAGEMENT MENU                     DEVMGT
   8. PROGRAMMING MENU                           PGM
   9. PROGRAM DEBUG MENU                         DBG
  10. MESSAGE HANDLING MENU                      MSGHDL
  11. UTILITIES MENU (IF INSTALLED)              UTL
  12. INPUT/OUTPUT SPOOLING MENU                 SPL
  13. SYSTEM CONTROL MENU                        SYSCTL
  14. WORK MANAGEMENT MENU                       WRKMGT
  15. CONFIGURATION MENU                         CFG
  16. SECURITY MENU                              SEC
  17. SAVE/RESTORE MENU                          SAVRST
  18. COMMAND MENU                               CMD
  19. SERVICE MANAGEMENT MENU                    SRVMGT


OPTION OR COMMAND:                    OR MENU:
PARTIAL COMMAND NAME:
CF1-EXIT    CF2-PREVIOUS MENU    CF16-EXECUTE WITH NO PROMPT
```

Command Prompting

The system enables you to prompt for the various parameters of all commands in two ways:

(a) enter the command and press CMD 4,
(b) preface the command with a question mark (?CRTLIB)

The second method is useful in CL programs to get the operator to fill in the command parameters before the program executes it. Observe that required parameters have the letter 'R' against them and parameters that may be coded positionally are prefixed with a 'P':

```
                    Create Library (CRTLIB) Prompt
Enter the following:
  Library name:                          LIB          R    oinklib
  Library type (*PROD *TEST):            TYPE         P    *PROD
  Public authority                       PUBAUT
    (*NORMAL *ALL *NONE):                                    *NORMAL
  Text 'description':                    TEXT              *BLANK
```

If you do not know the value for a parameter, enter a question mark instead of a value and the prompter will display the acceptable values:

```
      PARAMETER PROMPT WITH PERMISSIBLE VALUES (PUBAUT)

(*NORMAL *ALL *NONE):                                  *NORMAL


  *NORMAL
  *ALL
  *NONE
```

In addition to the basic prompting seen on the previous page it is now possible to selectively prompt (?) for keywords, filled with a default value or blank, allowing a user to enter or override values or to make a keyword protected. The following four combinations are available:

	Value Displayed	Protected	Value passed if nothing entered	Highlighted
??KEYWORD( )	Default	No	Default	No
??KEYWORD(Value)	Value	No	Value	Yes
?*KEYWORD( )	Default	Yes	Default	No
?*KEYWORD(Value)	Value	Yes	Value	Yes

program can output prompts to screen.

CL Programs

All CL programs begin with the PGM command (which is where you can specify the PARM keyword to name fields to receive parameters passed by a calling program) and end with the ENDPGM command which, when executed, has the same effect as RPG's SETON LR or COBOL's STOP RUN instruction. Following the PGM command there are often a number of declarative commands.

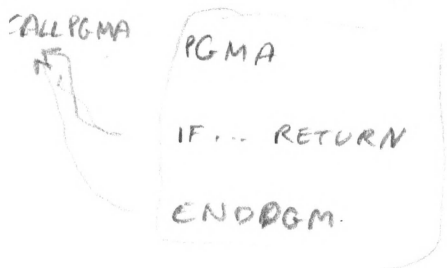
input value of this field to this program.

call PGMA PARM(& ...)

release 7:
?- : do not show this parameter

PGM PARM(& ....)
to specifies fields that are to receive values from other programs.

call to get out of a pgm before endpgm, use return (has same effect: transfer control back to calling pgm)



Declarative Commands.

'Declare' commands are directions to the compiler, are not executable and must appear before any executable command in the program.

DCL

The simple declare command defines CL variables (fields) to be used in a program. This is CL's equivalent of RPG's result field length entry on the calc specs and COBOL's Working Storage specification. The name of the variable can be up to eleven characters long and must always begin with an ampersand (&) to distinguish it from a command or an object name. The usual S/38 naming rules must be adhered to; remember that an underscore (_) is acceptable as the second and subsequent character of the variable name.

```
DCL  VAR(&WS_ID)  TYPE(*CHAR)  LEN(2)  VALUE(XX)
```

The TYPE can be:

a) *DEC - decimal max 15 digits, up to 9 on the right of the implied decimal point,

b.) *CHAR - character max 2000 in length, default 32,

c) *LGL - logical, one character only, the value of which may only be '1' (ON or TRUE) or '0' (OFF or FALSE). Indicators are always logical variables.

The VALUE keyword specifies the initial value of the variable, if required. Any number of DCL commands can be used in a program. If you declare a variable and never refer to it in the program, the compiler will warn you and not include it in the compiled program.

DCLF

A Declare File command, is used to declare a display file (or a data base file for input sequential use only) AND all fields/indicators specified in the record formats in the file. That is to say that the compiler implicitly declares any fields and indicators present in the DDS for all record formats specified. Field names will be prefaced with an ampersand and indicators will be of the form &INxx (xx is the indicator number).

RCDFMT *ALL : only the first 99 formats are declared

```
DCLF  FILE(MENU1DF.QGPL)  RCDFMT(*ALL)
```

The FILE keyword names the file. In the example, the file name is qualified by its library (QGPL). If a library is not specified, a library list is used to locate the object.

The RCDFMT keyword specifies which record format(s) to use in the program. Up to 50 formats may be specified - being selective helps you to reduce program size. The default is *ALL, which is the first 99 formats in the file, a CL restriction - a Display File can contain up to 1024 formats. more efficient to specify them.

The example on the following page illustrates how the DCLF command is treated by the compiler.

IBM Education SF38 4.10

```
Source file:                              QCLSRC.OELIB          Member:      TEMP        12/02/82 11:12:
CL compiler options:                      *SOURCE   *XREF    *GEN
Program generation options:               *NOLIST   *NOXREF    *NOPATCH
User profile:                             *USER
Program logging:                          *JOB
Public authority:                         *NORMAL
Text:
Compiler: IBM System/38 Control Language Compiler

                                      CONTROL LANGUAGE SOURCE

SEQNBR  *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ... ... 8 ... .

    100- PGM
    200- DCL &FRED *CHAR 10
    300- DCLF TEMP

         QUALIFIED DEVICE FILE NAME - TEMP.QTEMP

            RECORD FORMAT NAME - FORMAT1

              CL VARIABLE       TYPE      LENGTH      PRECISION      TEXT
                &IN01           *LGL         1                       End of job command key pressed.
                &IN20           *LGL         1                       Hilite FIELDA on screen.
                &IN21           *LGL         1                       Blink FIELDA on screen.
                &FIELDA         *CHAR       10
                &OPTION         *DEC         1            0

    400- SNDRCVF RCDFMT(FORMAT1)
    500- IF (&IN01 = '1') RETURN
    600- IF (&OPTION *NE 9) DO
    700-    CHGVAR &FRED 'PGMA'
    800-    CALL &FRED (&OPTION)
    900-    ENDDO
   1000- ENDPGM

                              * * * * *  E N D  O F  S O U R C E  * * * * *

                                         CROSS REFERENCE

DECLARED VARIABLES:

  NAME        DEFINED      TYPE      LENGTH      REFERENCES

 FIELDA         300       *CHAR       10          400
 FRED           200       *CHAR       10          700     800
 IN01           300       *LGL         1          400     500
 IN20           300       *LGL         1          400
 IN21           300       *LGL         1          400
 OPTION         300       *DEC         1 0        400     600     800
 CPF0791 00 No labels used in program.
```

Executable Commands

All other commands which can be used in programs are executable. Some commands are used to perform operations which require 'expressions' to provide values for a command parameter. The list of 'symbolic operators' below will give a clue as to what is meant by an 'expression'.
(Further information - CL Reference Manual, Appendix B)

PLUS	+	
DIVIDE	/	(slash)
MINUS	-	(hyphen)
MULTIPLY	*	(asterisk)
NOT	¬	or *NOT
EQUAL	=	or *EQ
LESS THAN	<	or *LT
GREATER THAN	>	or *GT
AND	&	or *AND
OR	|	or *OR
CONCATENATE	||	or *CAT
"	|>	or *BCAT
"	|<	or *TCAT

You can also use *GE, *LE, *NE, *NG and *NL

There are four types of expressions supported by CL:

1. Arithmetic. Operands in the expression must be decimal constants or variables, and complex expressions can be nested to a level of 5. The order in which the expression is resolved is in accordance with normal algebraic rules:

```
eg. CHGVAR &RESULT ((&A + 2) + (&X - 15.7 * (0.8 * &VAT)))
                    ------this is the "expression"----------
```

2. Character string.

Three operators are available for joining together (concatenating) strings of characters - *CAT, *BCAT and *TCAT.
All fields must be declared as *CHAR, but 'quoted' character string constants can be used.
Blanks are maintained when *CAT is used.

```
CHGVAR &A (&FIRST *CAT &SECOND)
```

If &FIRST contains 'FRED ' (notice the blank!) and &SECOND contains 'BLOGGS', the expression would yield (in field &A):

```
FRED BLOGGS
```

So also would CHGVAR &A (&FIRST *CAT 'BLOGGS')

*BCAT will concatenate variables and/or constants with one blank entered after the last non-blank in the first operand.
*TCAT will concatenate variables and/or constants without any blanks following the last non-blank character in the first operand, but leading blanks in the second operand are not truncated.

3. Relational.

The operands in a relational expression can be arithmetic, character or logical provided the symbolic operators are appropriate to the operands concerned. If character strings are of unequal length, the shorter fields are padded on the right with blanks, arithmetic fields are compared algebraically, and a logical '1' is greater than a logical '0'.

```
IF (&Y *GT (&X - 15.2)) ........
```

4. Logical.

The applicable operators are *OR and *AND. *NOT is used to negate logical variables:

means if &X='1' or &Y='1'

```
IF (&X *OR &Y) .........
IF (&X *OR *NOT &Y) ........
```

&Y='0'

Common Commands.

A number of commands will be used in everyday CL programming.

CHGVAR

The Change Variable command is equivalent to instructions in other languages which place a value in a result field. The variable to be updated and the variable, expression or constant used to update it, need not be of the same type (*CHAR, *DEC) - the system will perform implicit conversion. Logical variables, including DDS indicators, are set on and off with this command.

```
CHGVAR VAR(&X) VALUE(&Y)
```

The contents of &X are changed to the value of &Y.

```
CHGVAR &IN01 '1'
```

= turn on indicator Ø1
" 'Ø' = " off " "

Indicator 01 is set on (a Logical variable - hence the quotes).

```
CHGVAR &A  (&B + 20)
```

The contents of &B are incremented by 20 and the result placed in &A.

IF/ELSE

The two commands, IF and ELSE are usually used together.

'IF' evaluates a conditional expression and executes one command only, if the expression is true and (optionally) executes another (the associated 'ELSE'), if the expression is false.
Up to 10 levels of nesting are supported.

only 1 cmd allowed

```
IF  COND(&A *EQ (&B+&C))  THEN(CALL PGMA)
ELSE  GOTO A
```

The shorter form, omitting keywords (but retaining brackets around the expression) may be easier:

```
IF (&A = (&B+&C)) CALL PGMA
ELSE GOTO A
```

Indentation of the source code will aid visualizing the logic flow:

```
IF (&A = &B) GOTO A
ELSE IF (&C = &D) GOTO B
     ELSE IF (&E = &F) GOTO C
          ELSE GOTO ERROR
```

elses are superfluous

'IF' commands do not require a corresponding ELSE.
However, if an ELSE is specified, it must match up with a preceding IF.
(In this last example, the ELSE's are actually superfluous!)

'IF' commands do not need to specify a 'THEN' condition provided there is a corresponding ELSE to execute.
Here is an interesting example!:

```
IF &IN99
ELSE GOTO X
)
)
)    (a group of commands)
)
)
X:   CHGVAR &IN99 '1'
```

If indicator 99 is on (logically TRUE), the statements following the ELSE will be executed - but if it is off, branch to X.
This method avoids using the DO command (see below) to group together the commands to be executed if the condition tested is true.

DO/ENDDO

The DO and ENDDO commands 'delimit' a group of commands.
'Repeatable' DO-groups (DO WHILE/DO UNTIL) are not supported.
Up to 10 levels of nested DO's are supported (but not encouraged!):

```
IF (&A *GT &B) DO
     CHGVAR &X 'ABCD'
     CHGVAR &Y (&FRED + 2)
     ENDDO
ELSE CHGVAR &IN99 '1'
```

Each DO must have an associated ENDDO command.

Commands used with the declared Display File

A previously declared file can be used to communicate with the user of the program (as in RPG and COBOL). All the functions of the display stations can be used, specified via DDS.
Commands associated with 'writing' a format to a display screen and reading data from the display include:

SNDRCVF input and output cmnd to put fmt onto screen

This command is the most commonly used. It sends a named record format to the display station, after data management has incorporated the necessary data from the program variables. Observe that the first parameter is the name of the device to which the record format is to be transmitted - the default is the device from which the program was called - it is usually omitted, in which case the RCDFMT keyword, (the second parameter) naming the record format to be transmitted, must be specified.
(A format can be sent to more than one device - see CPF P/Guide, 3-55).

(*) = default : W.S. where prog is executed. (~~not interactive~~

```
SNDRCVF  DEV(workstation name)  RCDFMT(FORMAT1)
```

or, more usually:

```
SNDRCVF RCDFMT(FORMAT1)
```

Having executed the SND part of the command, the CL program will (by default) wait at this instruction until the operator responds by using the enter key or one of the enabled command keys, probably having keyed data into Input-capable fields of the format. The RCV part of the command is then activated.

An example of a program communicating with a workstation via a Display File is shown on the next page.

wait parameter on SNDRCVF cmnd default is *yes : wait for enter to be pressed
if wait(*no) : send format and carry on processing . in that case use WAIT cmnd . when RCV would be activated to receive any I field on the screen .

```
SNDRCVF  wait(*no)
cmnd
cmnd
```

IBM DATA DESCRIPTION SPECIFICATIONS

```
     A* MENU FOR ITEM ENQUIRY
     A*
     A          R FORMT1
     A                                  3 31'ITEM ENQUIRY MENU' DSPATR(HI)
     A                                  5 20'1. INQUIRY USING ITEM NO.'
     A                                  7 20'2. INQUIRY USING ITEM NAME'
     A                                  9 20'3. EOJ'
     A                                 11 16'ENTER ITEM NO. OR FIRST SIX CHARS.'
     A                                 11 51'OF ITEM NAME'
     A                                 13 26'HERE' DSPATR(HI)
     A            FIELDA         6    I 13 31DSPATR(PC)
     A                                 15 16'ENTER SELECTION (1,2 OR 3)'
     A                                 17 26'HERE' DSPATR(HI)
     A            FIELDB         1  0I 17 31RANGE(1 3)
```

Communicating with the User

```
      PGM
      DCLF FORMATS
A:    SNDRCVF RCDFMT(FORMT1)
      IF (&FIELDB *EQ 1) THEN(CALL ITMINQ PARM(&FIELDA &FIELDB))
      IF (&FIELDB *EQ 2) THEN(CALL ITSRCH PARM(&FIELDA &FIELDB))
      IF (&FIELDB = 3) THEN(GOTO END)
      GOTO A
END:  ENDPGM
```

SNDF

This command sends a named format to the display(s). The program continues from the next command.

RCVF

This command reads a record format's Input-capable fields from the display screen. SNDRCVF is more efficient than a SNDF followed by a RCVF, but you may want to present a format to the operator as soon as possible, allowing keying whilst the program continues (- to initialize variables, for example).

WAIT

This command allows for concurrent keying and program execution. The SNDRCVF and RCVF have a WAIT parameter, the default value being *YES. If WAIT(*NO) is specified, then the commands following the SNDRCVF or RCVF are immediately executed until a WAIT COMMAND is encountered. The System then waits for the enter key (or Cmd key) to be pressed, whereupon the RCV part of SNDRCVF (or the RCVF) is activated as previously described. The commands preceding the WAIT command are NOT RE-executed.

```
SNDRCVF  RCDFMT(FORMAT1)  WAIT(*NO)
)
)    (a group of commands)
)
WAIT
```

Exercise 4

The objective of this exercise is to write a CL program to drive the menu format in the display file which you created earlier.

1. Create a source file called QCLSRC in your library. Use this file to store CL statements.

2. Choosing your own name for it, write a CL program to "drive" the menu taking note of these points:

   Have the program replace the current library list with your own.

   The selection field is called OPTION.

   The command to be executed if CMD1 is used is RETURN.

   The dispatch notes are run by using:

   SBMJOB TEAMxx SF38TEAMxx RQSDTA('CALL PRINT')

   The Customer Inquiry program is called 'CUSINQ'.

   The DSPMSG command enables you to see workstation messages.

   Sign-off is achieved via the command SIGNOFF.

3. Call your program, test options 3, 4 and 5, then CMD1.
   Call your program again and test option 6.

Release 7 : issued in November

IBM Education

CALL/TFRCTL/RETURN

The CALL command invokes another program (regardless of language) and passes control to it. The calling program can pass data in the form of variables or constants, using the PARM keyword on the CALL. Up to 40 parameters can be passed. To receive them, a called CL program must have a PARM keyword on its PGM command (see diagram).
The Declared parameter variables in the called program must match the Declared parameter variables in the calling program. That is, length, type, decimal positions and sequence of the variables passed and received must agree exactly in both programs.
As the system only passes the address of the variables in the calling program, any changes made by the called program will be reflected in the calling program. Also the variable names can be different in the calling and called programs.

For further information on passing parameters, see CPF Programmers Guide, pages 3-90 to 3-100.

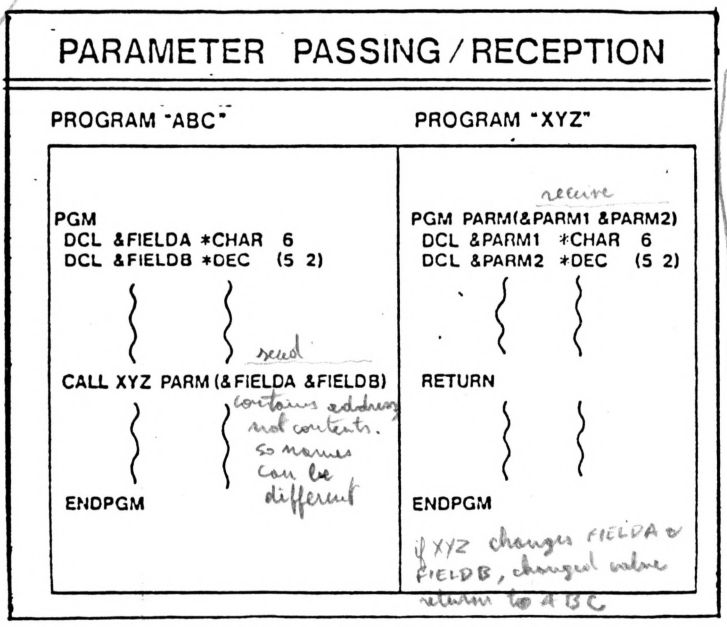
invocation stack : QCL SF38TESM07 QCL

each job needs 100 → 150 K work space (at least)
compilation : 200 – 250 K

THRASHING = too many jobs chasing too little space

if inv. stack ↗, resp. time ↘, hence

recurrence : eg QCL : QCL gives a command entry display



end user initial program mustn't return or endpgm because it would go back to QCL and give them a cmd entry display.
↳ hence have gotos before endpgm.

SDA created driver programs need to be changed if they are initial programs ← hence

The called program can close down and return to its caller by executing the ENDPGM command or the RETURN command - the two commands behave identically. Parameters are then implicitly returned.

It is usually unwise to include in a program designated as an Initial Program (named within a User Profile) the facility to execute either ENDPGM or RETURN. A Command Entry Display would normally result - not necessarily what End-users would appreciate!

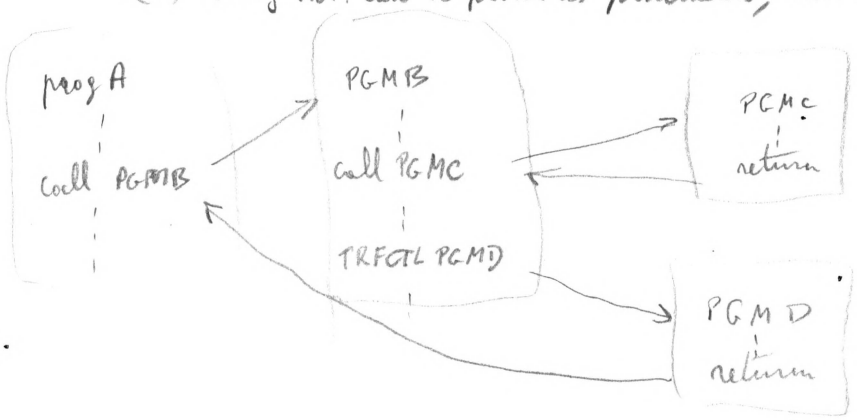
The TFRCTL command behaves in a similar way to CALL <u>but:</u>

(a) The transferring program is shut down and does not receive control back from the program to which it transferred control.

(b) The program to which control has been transferred returns to the caller of the program containing the TFRCTL command.

(c) The parameters passed to a program to which control is transferred <u>must</u> have been passed as parameters from the program to which control will eventually return. ⇒ (no more e no less)

(d) only var. can be passed as parameters, not constants



a constant is passed only but for a var (&...) it is the address that is passed.

<u>data area</u> : to pass parameters between jobs.

<u>Rel 7</u> : data queues.

you can pass up to 40 params. on a pgm

NB: - a call not in a prog allows to pass only constants, not var.
- an initial pgm cannot receive parameters.

Built-in Functions. (Ref: CL Reference Manual - Appendix B)

%SUBSTRING.

This function produces a subset of a character string from a CL variable or quoted character string. It is used with expressions, commonly the CHGVAR (Change Variable) and IF commands. The substring to be produced begins at a specified starting position and has a specified length. For instance: (assume &VAR contains 'ABCD')

```
%SUBSTRING(&VAR 3 2)
```

produces the substring 'CD' and would typically be seen in an expression such as:

```
CHGVAR  &FRED  %SUBSTRING(&VAR 3 2)
```

&FRED would contain 'CD' after the command had executed.

%SUBSTRING can be used with data areas to reference a particular part. There is also a short form of %SUBSTRING - %SST.

```
CHGVAR &Y (&A *CAT %SST(&B 1 2))
```

The first two characters of &B are concatenated to the contents of &A and the result placed in &Y.

%SWITCH.

This is used on the IF command to test one or more of eight 'switches' (U1-U8, previously known as the 'UPSI-byte') in your current job's Job Description (object!).

These switches are external to a program, and are able to be set up via CRTJOBD, CHGJOBD, SBMJOB and //JOB commands.

In your program, you provide a mask for the switch settings for which you are testing, eg:

to look at other job's switch, RTVJOBA

```
IF (%SWITCH(0X1XXXXX))  CALL PGMA
```

tests U1 for OFF and U3 for ON, ignoring the other switch settings. If U1 is off and U3 is on, program PGMA is called.

```
CHGVAR &CLSW8  %SWITCH(XXXXXXX1)
```

&CLSW8 (a Logical variable, length 1) is set to '1' if U8 is on.

Note: This only changes the CL variable (ie. field) - the switch setting remains unaffected.

RTVJOBA

This command is used in a CL program to retrieve chosen attributes of a job (keywords in uppercase) and place them into CL variables.

JOB	job name (for interactive jobs, the terminal name)
USER	user profile name
NBR	system-assigned job number
TYPE	batch (B) or interactive (I)
SWS	U1 - U8 (8 characters)
DATE	date job started (equivalent to UDATE)
EXCPTY	execution priority
OUTQ	output queue
LOGLVL	message logging level
USRLIBL	current user library list
ACGCDE	the job's accounting code
RTNCDE	indicates the completion status of CALLED program - the codes are: 0 = a called program returned control 1 = a called RPG program ended with LR on 2 = a called program was cancelled 3 = a called RPG program ended with HALT indicator on.
CNLSTS	1 = controlled cancel of system or subsystem in progress.

For example, to retrieve the user identity and the terminal name of the user of the program the commands would be:

```
DCL &USER *CHAR 10
DCL &TERMID *CHAR 10
RTVJOBA  USER(&USER) JOB(&TERMID)
```

DLYJOB

Allows a job to wait for a specified time - up to 11 days, 13 hours! or to wait until a time of day.

CHGJOB

To change within a program the attributes of a job (whether it is the current job or some other job, executing or not) the CHGJOB command is required, eg:

```
CHGJOB  JOB(*)  SWS('10000000')
```

The '*' indicates the (current) job in which this command is executing.

Note: If you change the switch settings in a CL variable, U1-U8 are NOT changed until a CHGJOB is executed, eg:

```
DCL &SWITCHES  *CHAR  8

CHGVAR  %SUBSTRING(&SWITCHES 4 1)  '1'
CHGJOB  SWS(&SWITCHES)
```

ALCOBJ/DLCOBJ

The Allocate Object command is used to reserve an object for use by a job. For instance, you may want 'exclusive' use of a file whilst you update it, to prevent the possibility of double updates:

```
ALCOBJ  ((FILEA *FILE *EXCL))
CALL UPDATMSTR
DLCOBJ  ((FILEA *FILE *EXCL))
```

Note the use of double brackets. These commands can operate on up to 50 objects with one use of the command. Double brackets denote a list within a list.

The DLCOBJ releases the lock obtained with a prior ALCOBJ.

QOPRMENU operator menu initial program -

cmd 1 on QOPRMENU sends an escape msg : CPF2320. it must be monitored in the original prog.
QPGMMENU

MONMSG

This important command is used to 'trap' messages sent to your program.

The messages can indicate (like sophisticated return codes) that something has gone wrong with a function executed in the program and that the program must take action. Alternatively, messages may simply communicate information that you have requested from within the program.

MONMSG can be the first executable command in a program so that it acts as a monitor for messages sent to the program as a whole:

```
PGM
DCL &VAR *DEC (5 2)
MONMSG (CPF4131) EXEC(GOTO LABEL)
```

Note: At the 'program level' you may only specify one Command - the GOTO - and if specified, it MUST be preceded by the EXEC keyword.

MONMSG can also follow individual commands to trap any errors or information returned when those individual commands are executed.

```
CHGVAR &VAR (&A/&B)
MONMSG (MCH1211) EXEC(CHGVAR &A (1))
```

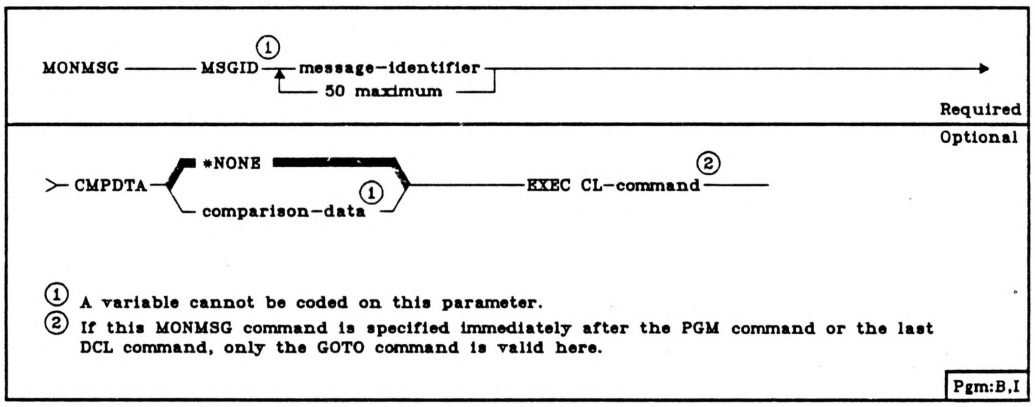
at command level

A particular message type which could be sent to your CL program is an 'Escape' message. If NOT trapped, your job could ABNORMALLY TERMINATE (- through 'Function Check'). A case in which Escape Message CPF2320 is sent to your program is when you execute the supplied programs QPGMMENU and QOPRMENU, and then exit from them via CMD 1.

For further information, see the MONMSG command description in the CL Reference Manual.

If a command is coded for the EXEC parameter on a MONMSG command that is placed at the beginning of a program, *only* the GOTO command may be used, and it must specify the label for the command to which control is to be passed if a monitored message occurs. If a command is not coded for the EXEC parameter, any monitored messages are ignored.

Restrictions: This command is valid only within CL programs. It can be coded after the last declare command (if declare commands are used), following the PGM command that begins the program, or it can be coded following any command allowed in CL programs, except for the following: DO, ELSE, ENDDO, ENDPGM, GOTO, IF, or RETURN. Note that if another program sends a message that is monitored by this command, a return cannot be made to that program.



MSGID Parameter: Specifies the message identifiers of one or more escape, notify, or status messages that are to be monitored by this command. As many as 50 specific and/or generic message identifiers can be specified on one command.

Note: Many CL commands issue one escape message for many different error conditions. Details about the error or failure are given in diagnostic messages that precede the escape message. Although diagnostic messages cannot be monitored, they can be received from the job's external message queue after the escape message has activated the user's message monitor.

The message identifiers must each be 7 characters long and in the following format: *pppmmnn*. The first 3 characters must be an alphabetic program code, and the last 4 characters must be a decimal number. If zeros are specified in either two or all four of the rightmost positions, such as ppmm00, a generic message identifier is being specified. For example, if CPF0000 were specified, all the CPF messages would be monitored. (See the *IBM System/38 Messages Guide: CPF, RPG III, and IDU.*)

Enter the message identifiers of one or more messages that are to be monitored when they arrive at this program's message queue. (For the identifiers of the escape, notify, and status messages that can be sent by the CL commands, refer to Appendix E in Part 3.) CL variables cannot be used to specify any message identifiers.

CHKOBJ

This command verifies the existence of and (optionally) the authority you have to that object, before you actually attempt to use the object. This command should have at least one MONMSG following it to monitor for the messages issued by the command:

```
CHKOBJ   FRED *FILE  AUT(*UPD)
   MONMSG (CPF9801) EXEC(GOTO A) /* Object not found */
   MONMSG (CPF9802) EXEC(GOTO B) /* Not authorised */
```

CHKDKT/CHKTAP

These commands allow the user to check for the existence of a specific volume (or file) label on a diskette or tape before proceeding with a specific function that uses that medium.

```
CHKDKT LOC(*S1) VOL(MASTER) LABEL(CUST)
   MONMSG (CPF6162) EXEC(SNDPGMMSG MSG('Diskette Not Loaded'))
```

DATA AREAS

A Data Area is a particular kind of object on S/38 - a byte string that can be of character (up to 2000 long!), decimal or logical type. In fact, Data Areas behave like fields outside the program.
The rules for length and format are similar to those for declaring CL variables - but names for Data Areas are not preceded by an ampersand. The following commands apply:

	CRTDTAARA	(Create a data area)
	DLTDTAARA	(Delete a data area)
	DSPDTAARA	(Display the contents of a data area) #
	DCLDTAARA	(Declare a data area) *
*	CHGDTAARA	(Change a data area) #
*	RTVDTAARA	(Retrieve the contents of a data area into any named CL variable) * #
	RCVDTAARA	(Read a data area into the CL variable of equivalent name) *
	SNDDTAARA	(Update a data area with the contents of the CL variable of equivalent name) *

(*) indicates the command can only be used in a CL program.
(#) indicates only the commands which can reference *LDA (see over).

Use in a CL program.

Data can be moved in and out of the Data Area using the SNDDTAARA and RCVDTAARA commands (see example on next page). Alternatively, data can be retrieved into a program - into a previously declared CL variable - and the contents of any CL variable can be used to change a Data Area.

Except for the use of RTVDTAARA and CHGDTAARA, the Data Area must have been 'declared' to the compiler with the DCLDTAARA commmand. The compiler will also implicitly declare a variable in your program that has the same name as the declared data area but preceded with &.

```
DCLDTAARA  POLICYNO
ALCOBJ

RCVDTAARA POLICYNO
CHGVAR &POLICYNO (&POLICYNO + 1) /* &POLICYNO declared +
                                    implicitly by compiler */
SNDDTAARA POLICYNO  /* Write out to data area */
DEALCOBJ
```

There are some important considerations regarding data areas;

To provide a unique copy of a data area for a user, the system creates a Local Data Area (*LDA) - see CPF P/Guide 12-3.

COBOL cannot use data areas other than *LDA.

Note that for RTVDTAARA, the data area does not have to be declared to the program. The CHGDTAARA command can be used to update the contents of a data area which has been retrieved. Both commands can use the %SUBSTRING built in function.

Batched CL

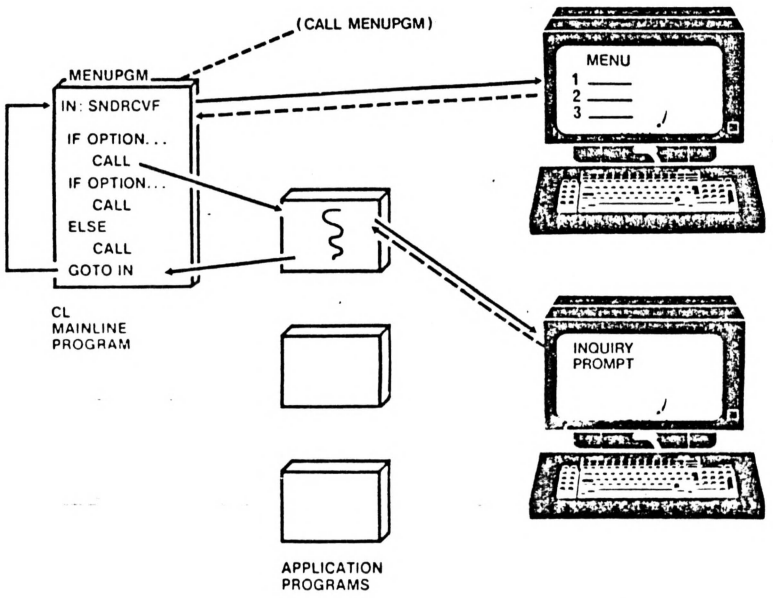
CL commands can be entered into a database file (via EDTSRC TYPE(*CL)) and can be processed as a batch job stream using the Start Data Base Reader (STRDBRDR) or Submit Data Base Job (SBMDBJOB) command. This enables serial execution of the commands from a job queue, and is similar in concept to an OCL or JCL procedure without IF/ELSE logic on other systems.

o Interactive jobs

There are two main approaches to the design of an interactive application, or an interactive job, on the System/38.

A small number of multiple-function programs could be coded to handle all aspects of the application. The machine's paging activity would tend to favour only those pages of the program currently in use, so program size would not be important. The programmer must observe all the rules of structured programming, and develop by the 'top down' approach.

A mainline program could call sub-programs. This approach has been adopted on many small interactive systems, which are constrained by the necessity to keep program sizes down. It can be implemented on the System/38 by use of the Call/Return mechanism. The mainline program could be written in CL, COBOL or RPG, although CL provides functions (like access to message files and queues) not directly available in RPG or COBOL.



o Batch jobs

In many commercial applications, there are many batch-oriented jobs. Long print jobs (invoices, aged trial balance reports, and statements) are examples. So are functions like:

CRTRPGPGM
CRTCLPGM
CRTDSPF
CRTPF
CRTLF

The 'application-create' phase of Query and DFU are also batch-type functions. It is a good idea to isolate these from interactive jobs, and run them as batch jobs.

On System/38, a batch job is often a series of CALLs to batch programs, in sequence. The job would normally be stored in QCLSRC and then run using either the command SBMDBJOB or STRDBRDR.

```
//JOB  WEEKLYP  JOBD(PAY)
OVRDBF  EMPLOYEES  MBR(WEEKLY)
CALL  PAYWKLY
CALL  PAYSLIPS
CALL  PAYCHKS
//ENDJOB
```

The example above has a fixed sequence, but there are occasions when more flexibility to provide run time variables is needed. For example:

```
SBMJOB  PAYROLL  JOBD(PAY) +
    RQSDTA('CALL PAYJOB PARM(' *CAT &PMBR *CAT ')')
```

The previous command might well be part of a menu program, where the value of the field &PMBR is entered in the menu display.

The Batch job might then be designed, like an interactive job, around a main line CL program.

```
PGM  PARM(&PMBR)
DCL  &PMBR  *CHAR  10
OVRDBF  EMPLOYEES  MBR(&PMBR)
CALL  PAYPGM  PARM(&PMBR)
CALL  PAYSLIPS  PARM(&PMBR)
CALL  PAYCHKS
ENDPGM
```

This can represent the nested OCL or JCL procedures on other systems. A program must be used if the system operator, or the user who submitted the job, is required to enter parameters affecting the way the job runs.

Summary.

CL provides a powerful interface to the system that can be employed by any authorised workstation user. You would typically use a CL program for:

- Controlling application logic flow.
- Controlling Menus.
- Executing commands whose parameter values change at execution time.
- Manipulating data areas.
- Conditional execution of commands.
- Using built-in functions (%SWITCH & %SUBSTRING).
- Message monitoring and handling.
- Checking an object's existence and authorisation.

Exercise 5

The objective of this exercise is to expand your menu and CL program.

1. Add to your menu:

    1. Programmer Menu

    2. System Operator Menu

2. Re-create your Display File.

3. Modify your CL program to take account of the above.

    The Programmers Menu program is called 'QPGMMENU'.

    The System Operators Menu program is called 'QOPRMENU'.

    These programs return the escape message CPF2320 to your program when they terminate. This is normal, but you must monitor for this to prevent a Function Check. Use the MONMSG command in such a way that only one is needed in your program and that your menu is re-presented upon receipt of the message.

4. Re-create your CL program.

Optional Additional Exercise

1. Build into your CL program the creation - if it does not already exist - of a Data Area called TIME in your team library.
   It should be character, length 6.

2. Use the RTVSYSVAL command to retrieve the system time (QTIME) into a CL variable.

3. Update the data area to contain the time you started the current run of your menu program.

4. The contents of the data area can be displayed (DSPDTAARA) using option 5 from the Programmer's Menu or from the command entry screen.


stack
msgq
*EXT
QCL
QCL
to send msg To user
if it is a status msg, it is displayed on line 24

Optional Additional Exercise

Change your Display File and CL program as follows:-

1. If Command Attention Key 12 is pressed the following information should be displayed on line 23, starting in position 51 -

   WORKSTATION ID - xxxxxxxxxx

   where xxxxxxxxxx is the identification of the workstation using the program (ie. the correct workstation id should be displayed for whatever workstation is used to call the program).

2. Code this change to the display file as an output field.

3. Change your CL program to set up the output field to contain the information that should be displayed as described in (1).

   DON'T FORGET - the workstation id should be displayed ONLY when CA12 is pressed!

Optional Additional Exercise

Create a new CL program which could be used as an INITIAL program after exhaustive testing.

It should include:

1. Setting up a complete and correct Library List for you.

2. Tailoring and calling a Programmer Menu for you.

3. NO opportunity for RETURNING to any program.

Let the Instructor know the qualified name of the CL program if you would like your User Profile to be changed.

MESSAGES

External message handling
Message queues
Message commands
Message files

Exercise 6

USER → USER : SNDMSG to ~~terminal~~ W.S. MSG Q

delivery mode: NOTIFY — if is signed on
HOLD — " not signed on or switched on
BREAK
DEFAULT : gives default reply to INQ msg automatically

msg type : ESC
INFO
INQ (⇒ reply)
NFY
RPY } type for P. to P. type of msg

msg Q's : WORKSTN for W.S.
USER for user.
QSYSOPR
QCONSOLE

Type (cont)
DIAG
COMP
SNDR { keeps a copy of msg sent so that can get reply
STATUS

→ stick this in initial program ?
severity ..
CHGMSGQ

SNDBRKMSG
SNDPGMMSG program to program
SNDUBRMSG
RCVMSG
SNDRPY

External Message Handling

A message is a communication between one point in the system and another. Those points could be program to user, user to user, or program to program. The most obvious messages are those sent to a display station to inform the operator of an error.

Responding to error messages.

When an error message is displayed you can respond in one of five ways:

1. Press Error Reset, if necessary, and continue.

2. If the Input Inhibited light is on, press the HELP key to obtain second level text.

3. If the Input Inhibited light is off, position the cursor on the same line as the message and press HELP to obtain second level text.

4. If using the Command Entry Display, press Command key 7 to obtain further messages relating to the error and position the cursor against one of the error messages to obtain second level text by pressing HELP.

5. If using the Programmer's Menu, press Command key 3 to obtain the Command Entry Display and proceed as in step 4.

Message Queues.

All messages are sent to queues. When workstation Device Description objects are created, the system automatically creates a message queue for that device that has the same name. When you sign on to the display station the DELIVERY MODE of the queue is set by CPF to NOTIFY mode. There are four modes of delivery:

BREAK
Messages arriving in your queue that meet or exceed the severity level of your message queue are displayed to you as soon as they arrive, interrupting what you are doing. Those messages that are transmitted with a severity of less than that of your queue's filter are kept until you ask to see them.

NOTIFY
Your work is not interrupted; the Message Waiting indicator illuminates and the alarm sounds briefly. You display the message(s) when you feel inclined to do so.

HOLD
The message queue holds all messages that arrive and does not notify you.

DEFAULT
Messages requiring a reply are answered with the default reply, which is specified in the message description. Information only messages are ignored. No messages will be found on the queue, if it is displayed.

When you sign on, the message queue for the workstation is set to NOTIFY mode by the system. When you sign off, the queue reverts to HOLD mode. The DSPMSG command is used to display messages.

Special Queues.

QCONSOLE
The name given to the system console message queue. It is always in NOTIFY mode unless specifically reset. The Attention light on the console indicates that one or more messages are waiting.

QSYSOPR (REF: Chapter 11 System Operators Guide)
An important queue. Messages intended for the system operator are sent to this queue. The system allocates the QSYSOPR queue to the first person who signs on at Start CPF (IPL) time; the queue is set to BREAK delivery mode.

NOTE: When that person signs off, the QSYSOPR queue reverts to HOLD mode. The person responsible for system operations must enter the Change Message Queue (CHGMSGQ) command at another workstation to 'connect' him with the system operator's queue. This feature allows anyone to be the system operator.

Changing a message queue.

The Change Message Queue (CHGMSGQ) command is used to alter the status of a queue. The parameters are:

Delivery (DLVRY) -	*HOLD *BREAK *NOTIFY *DFT
Severity (SEV)	Severity filter for messages on a queue. Only messages of a given severity or higher notify the user to whom the queue is allocated of their arrival.

When the operator signs off terminal TERM34, the system implicitly executes the command:

```
CHGMSGQ TERM34 *HOLD
```

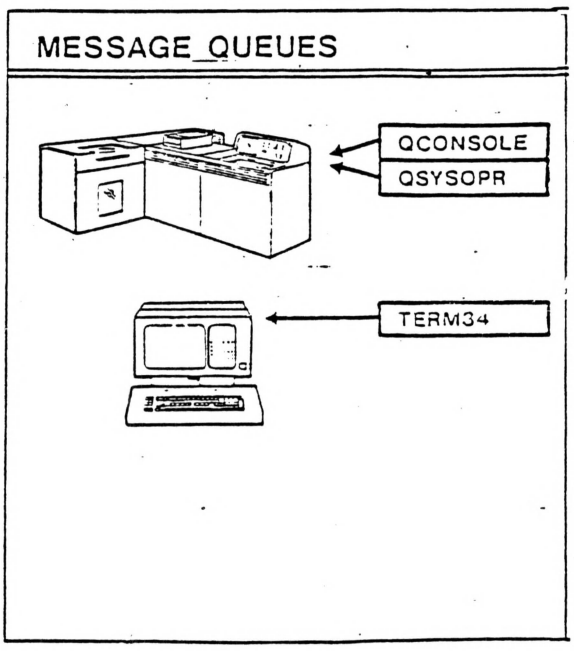
and the queue holds all messages that arrive at the terminal queue. Any user (subject to security) can sign on and the system implicitly executes the command:

```
CHGMSGQ TERM34 *NOTIFY
```

The queue is now 'active' and will notify the operator of any messages arriving on the queue. The operator can set the queue to *BREAK mode and the severity filter of a queue can be altered such that only messages of a given severity or higher will interrupt or notify the user to whom the queue has been allocated:

```
CHGMSGQ TERM34 *BREAK SEV(61)
```

Note that this last command is a permanent change to a queue however.



User Message Queues

You can create your own queues with the Create Message Queue (CRTMSGQ) command. These queues are independent of workstations but can be allocated to workstations with the CHGMSGQ XXXXXX DLVRY(*BREAK or *NOTIFY) command. Message queues cannot be saved.

Sending Messages

The SNDMSG and SNDBRKMSG commands are used to send messages to queues. Messages sent by SNDMSG are severity 80 and those sent by SNDBRKMSG are severity 99. For example:

```
SNDMSG  'Hullo Sailor, wots your game?' TOMSGQ(TERMINAL1)

SNDBRKMSG 'The system is shutting down in ten minutes. Please wrap
          it up now.' *ALLWS
```

Very often you need to send a message to somebody and you aren't sure which terminal they are signed on at. Consider creating message queues for each individual who uses the system and have these queues 'connected' to the terminals by their initial programs:

```
CRTMSGQ MIKE.QGPL
CRTMSGQ FRED.QGPL etc..
```

and their initial programs would contain: CHGMSGQ MIKE *BREAK or CHGMSGQ FRED *NOTIFY or whatever.

Now anybody wanting to communicate with MIKE could send a message to the queue MIKE:

SNDMSG 'Please come to meeting in room 20 at 10.00 Tuesday' MIKE

Even if MIKE is not signed on, the message will be sent to his queue and remain there until it is removed.

Often, when sending messages, you require a reply from the recipient of the message so you can send an inquiry message:

```
SNDBRKMSG 'Will you be in the Queens Arms at lunchtime?' +
          TOMSGQ(TERM34) MSGTYPE(*INQ) RPYMSGQ(FRED)
```

Because it is a Break Message, the terminal message queue is interrupted and this would appear on the screen:

```
Will you be in the Queens Arms at lunchtime?
? _____________________________________________________
```

The response MIKE makes will be sent to the RPYMSGQ message queue specified by the sender (it is not necessarily the sender - the default is QSYSOPR).

Message Files

Messages can be stored in a message file for future use by a program. Contrast this method with the sending of impromptu messages. Message files are created with the CRTMSGF command.

```
CRTMSGF FXMSGSM
```

Individual messages are added into a Message File with the Add Message Description command:

```
ADDMSGD MSGID(UFX0010)
        MSGF(FXMSGSM)
        MSG('This is not a valid currency code')
        SECLVL('Refer to your Foreign Exchange +
        handbook for a list of valid FX currency codes')
```

- Message identifier. Takes the form XXXnnnn. Acts as a sort of 'key', and must be unique within the file.

- First level text consists of no more than 132 characters, but remember that 75 is all that can be seen on the display station.

- Second level text consists of no more than 1435 characters. The "help" key is enabled by the use of the keyword SECLVL.

- A severity code (00 - 99) can indicate the urgency of the message.

The usefulness of the ERRMSGID keyword for display files was discussed previously.

The DSPMSGD command can be used to view messages in a file,but the DSPMSGF (Display Message File) is often more useful. Message Descriptions are ADD-ed, CHG-ed or RMV-ed.

Exercise 6

1. Your Display File incorporates a RANGE check on the screen format to ensure that invalid options are not entered by the operator. Remove this range check and let the program deal with invalid options selected.

   Create a message file and add a message description to the effect that an invalid option has been selected. Also supply a second level text for the operator.

   Re-write your display file to incorporate an ERRMSGID keyword that will be actioned as a result of your program detecting the invalid option.

   Re-write your program to incorporate the necessary logic.

2. Change your terminal message queue to BREAK mode and then run 'Dispatch Notes' from your menu.

3. If there is time, (the instructor will let you know if there is) enhance your program as follows:

   Enable Command key 3 to obtain the Command Entry Display (CALL QCL).
   Ensure there is text on your menu to describe this function.

   If a command key is pressed ignore any options entered by the operator.

DATA BASE FILES

DATA BASE FILE DEFINITION AND CREATION

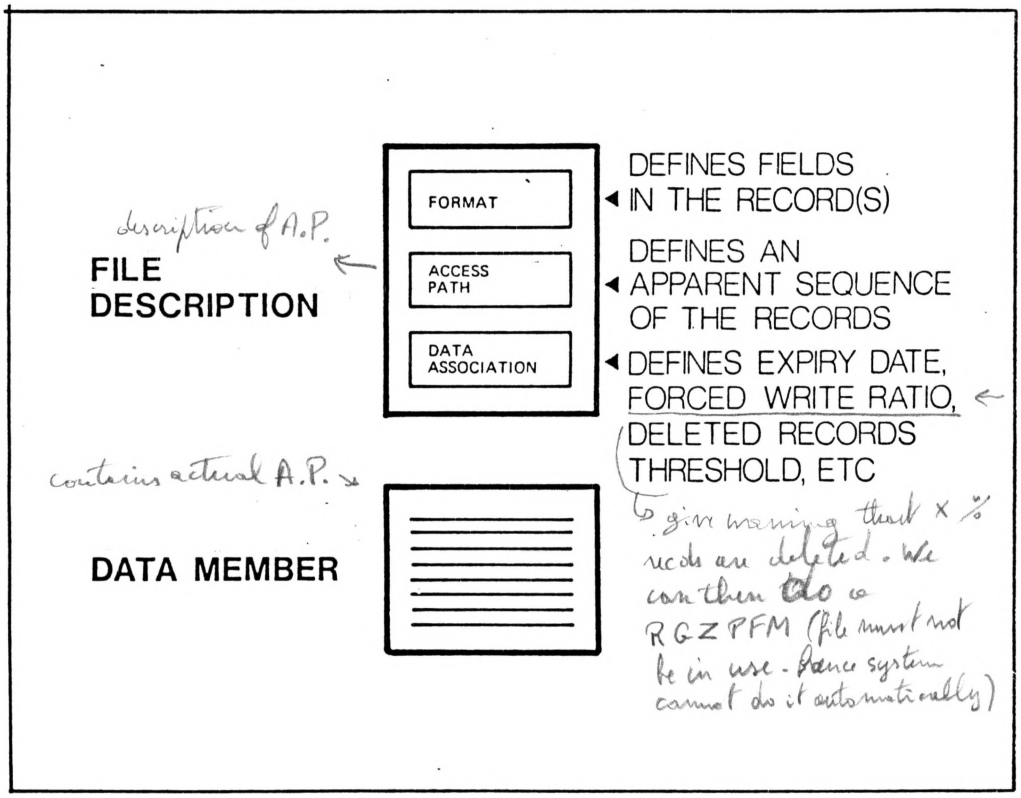
File structure
Creating a file
Creating a physical file
The Field Reference File
Logical files
Creating a logical file
Logical files using two or more physical files
Access path maintenance
The file creation commands

Exercise 7

Exercise 8

DATE BASE FILE DEFINITION AND CREATION

- o File Structure



All files on the System/38 must have a file description. For data base files this can have three components.

PF: only 1 rec. fmt.
LF: up to 32 R.F. (i.e. view of up to 32 P.F.)
Access path = index

Record format - describes the layout of fields in a record of the file.
For example:

ORDNO	CUSNO	ORDATE	ORDVAL	SHIPBY
6,0	6,0	6,0	8,2	30,A

The record format describes how the record looks when it is presented to the user program. For a physical file, it also affects the way data is actually stored. The data base will pack fields defined as numeric when storing them, unless they are defined otherwise.

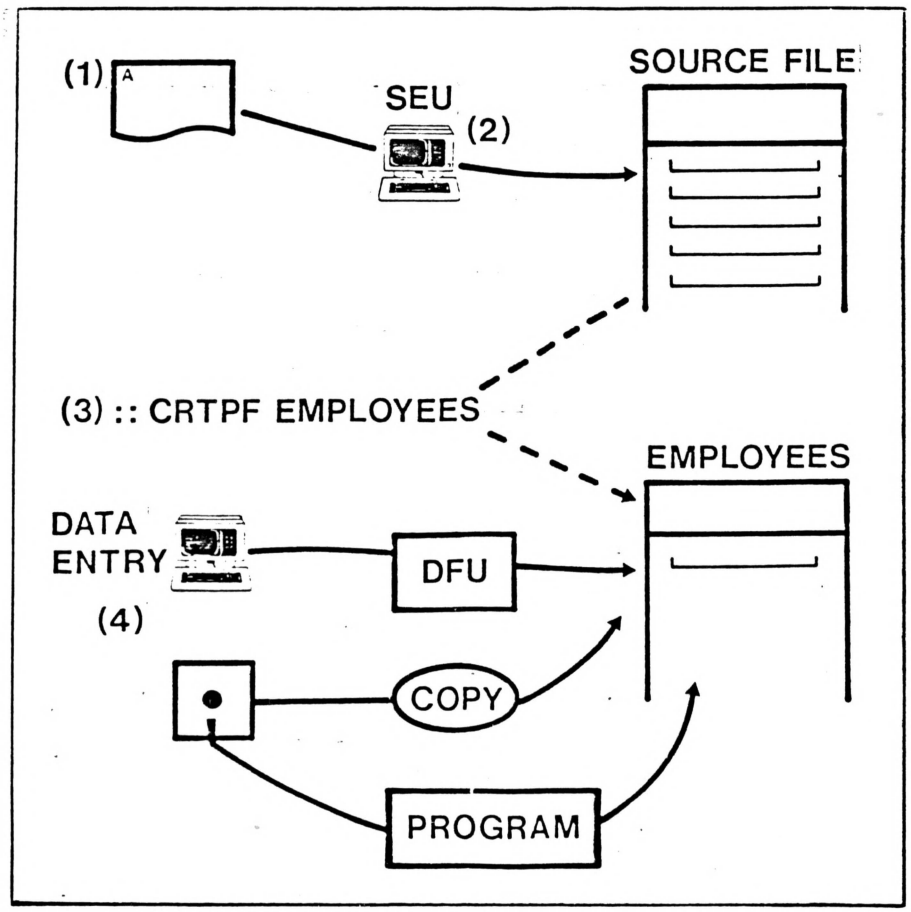
Access path - records are stored in physical files in arrival sequence. If records need to be processed in a different sequence (usually based on the value of a field or set of fields) a keyed sequence access path can be defined. This behaves like an index to the records in the file.

Data Association - these are details provided, when the file is created, about expiry date, size, space allocation and other file characteristics.

Files can also have one or more members, which group records together. When a file is created, a member is normally created at the same time with the same name as the file.

o Creating a file

Physical files are created with a Create Physical File command. The command takes source statements to define the record format and access path from a source file. (A source file is a specific kind of physical file.) These source statements are Data Description Specifications, which are entered through the Source Entry Utility, SEU.

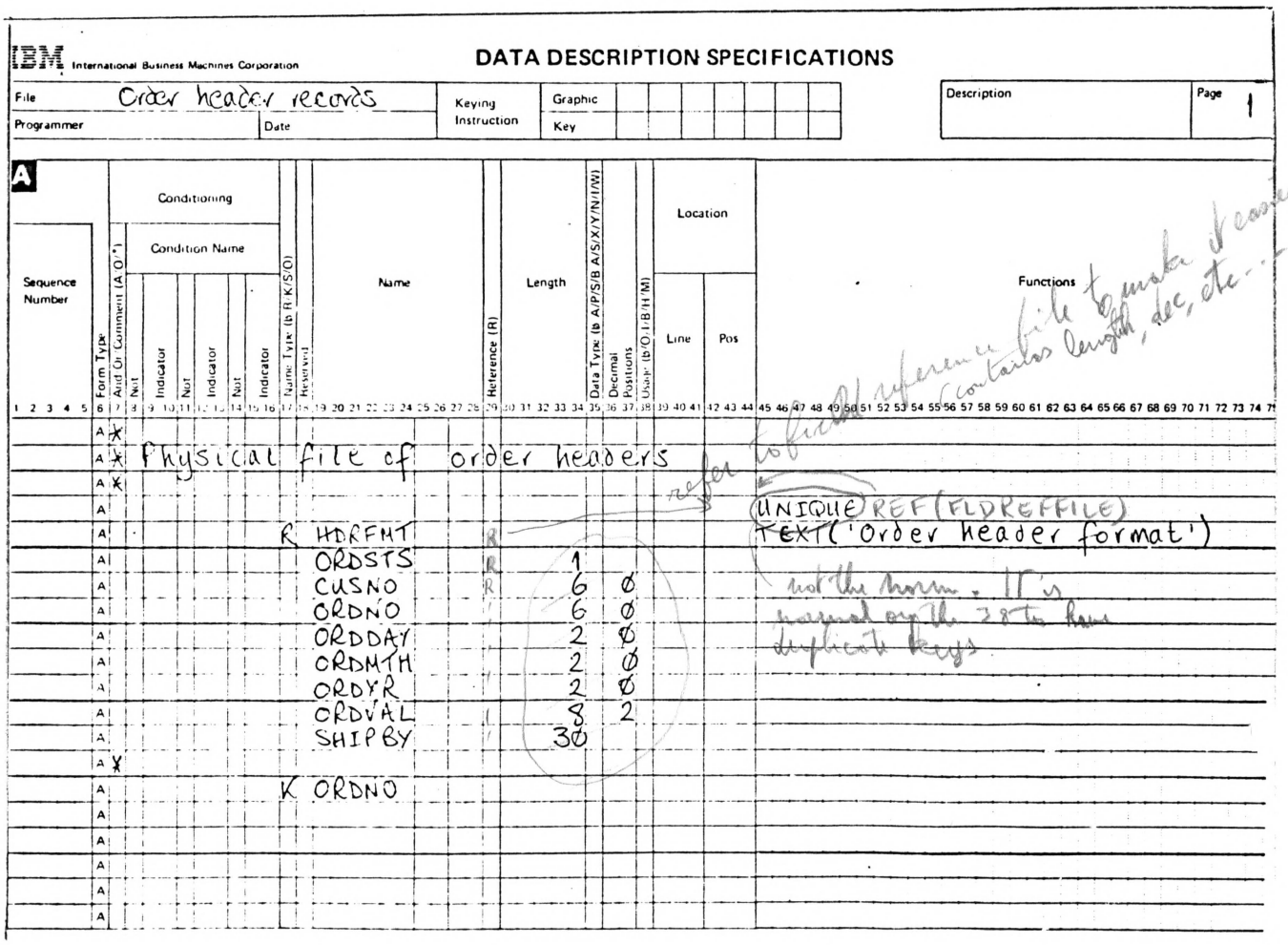


Data Description Specifications.

DDS supply a record format name (R in column 17) and the names of all fields in the format, and optionally a keyed access path (K in column 17).

Keywords in the 'Functions' area of the form provide optional facilities.

Editing keywords
Text keywords
Validity checking
Reference
Access path: UNIQUE will ensure that records added or updated in the file will have unique keys. This keyword is entered at file level - that is to say, before a line in which a record format is named.



IBM International Business Machines Corporation

DATA DESCRIPTION SPECIFICATIONS

File: Order header records | Programmer | Date | Keying Instruction | Graphic / Key | Description | Page 1

Sequence Number	Form Type	Conditioning: And Or	Conditioning: Comment (A,O,*)	Conditioning: Condition Name: Not	Conditioning: Condition Name: Indicator	Conditioning: Condition Name: Not	Conditioning: Condition Name: Indicator	Conditioning: Condition Name: Not	Conditioning: Condition Name: Indicator	Name Type (b,R,K/S/O)	Reserved	Name	Reference (R)	Length	Data Type (b,A/P/S/B,A/S/X/Y/N/1/W)	Decimal Positions	Usage (b/O,I,B/H/M)	Location: Line	Location: Pos	Functions
	A	*										Physical file of order headers								
	A																			(UNIQUE) REF(FLDREFFILE)
	A									R		HDRFMT	R							TEXT('Order header format')
	A											ORDSTS		1						
	A											CUSNO	R	6		Ø				
	A											ORDNO		6		Ø				
	A											ORDDAY		2		Ø				
	A											ORDMTH		2		Ø				
	A											ORDYR		2		Ø				
	A											ORDVAL		8		2				
	A											SHIPBY		30						
	A	X																		
	A									K		ORDNO								

Format fields

ORDNO is to be the key field of the P.F.

refer to field reference file to make header contains length, dec, etc...)

not the norm - IT is required on the 38 to have duplicate keys

The second line in our example shows how text may be entered to appear on compilation listings.
In this case, we are specifying that we are associating the text with a record format name. Therefore, the TEXT keyword is at record level.

The remaining lines are at field level.

Entering the DDS source

The file definition source statements are entered in a member of the default source file QDDSSRC. They can be copied in (from diskettes keyed off-line) or entered with SEU (which provides syntax-checking).

```
::EDTSRC SRCFILE(QDDSSRC) SRCMBR(ORDHDRSP) TYPE(*PF) +
     TEXT('Order Header File')
```

As before, you should give the source member the same name as the file that is going to be created from it.

Creating the file

```
::CRTPF   ORDHDRSP.Library name
```

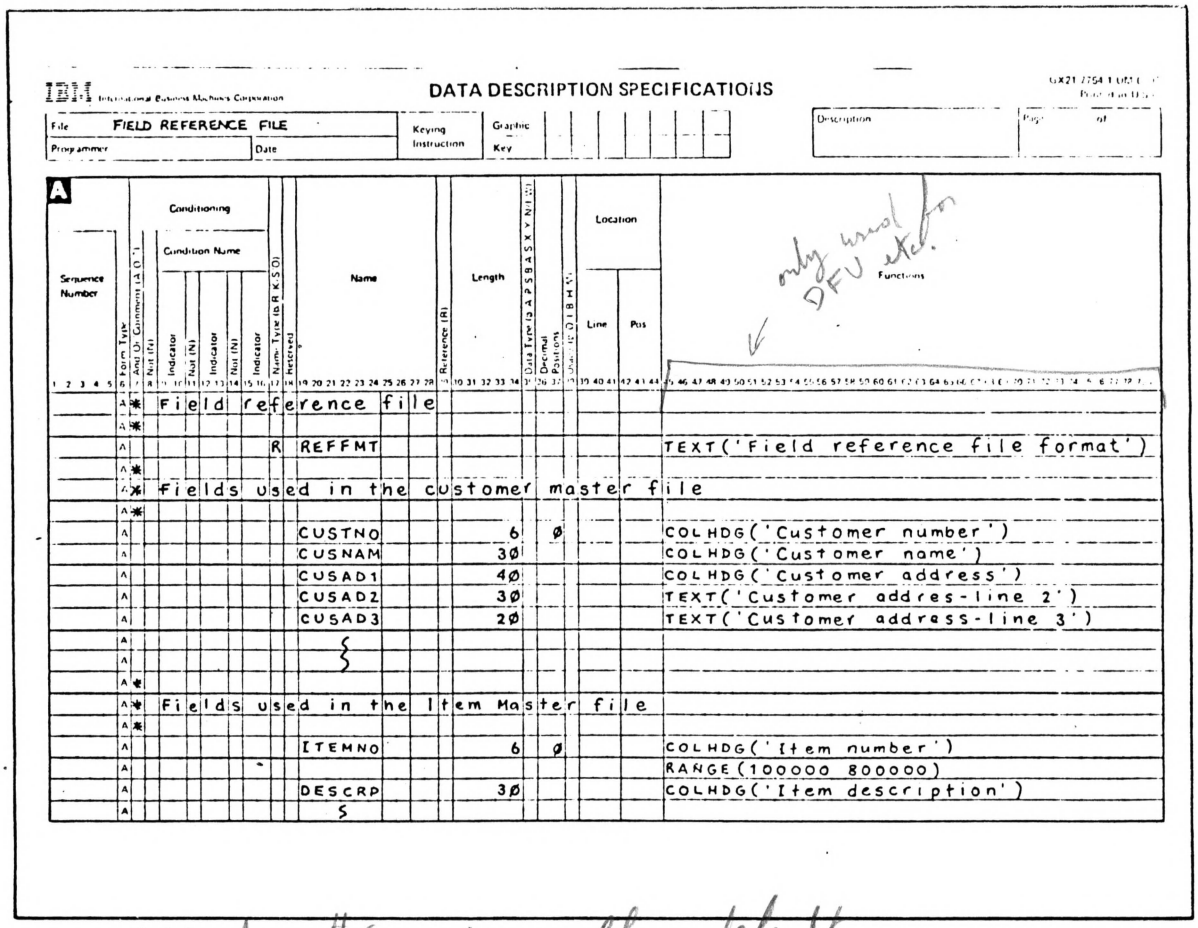
Loading the file

The file contains no data records but can now be used in the compilation of an RPG program. Records can later be put into the file through a COBOL or RPG program, the DFU function of the Interactive Data Base Utilities, or a copy command:

```
::CPYF   FROMFILE(DISKET)   TOFILE(ORDHDRSP)  +
     FROMMBR(data set name)
```

o The Field Reference File

Any file, except a logical file, can refer to any other file for its field definitions. A good way to force standardisation of fields across an application is to define all fields in a single physical file. This would never actually contain data records but would be used to refer to when other physical files are created. the referred-to field definitions will be used in the new physical files.



DATA DESCRIPTION SPECIFICATIONS

File: FIELD REFERENCE FILE | Keying Instruction | Graphic Key | Description | Page | of

Programmer | Date

Sequence Number	Form Type	Conditioning: And/Or	Conditioning: Not	Condition Name: Indicator	Condition Name: Not	Condition Name: Indicator	Condition Name: Not	Condition Name: Indicator	Name-Type (R,K,S,O)	Reserved	Name	Reference (R)	Length	Data Type (blank,A,P,S,B,4,S,X,Y,N,1,2)	Decimal Positions	Usage (blank,I,O,B,H,X)	Location: Line	Location: Pos	Functions
	A *										Field reference file								
	A *																		
	A								R		REFFMT								TEXT('Field reference file format')
	A *																		
	A *										Fields used in the customer master file								
	A *																		
	A										CUSTNO		6		Ø				COLHDG('Customer number')
	A										CUSNAM		3Ø						COLHDG('Customer name')
	A										CUSAD1		4Ø						COLHDG('Customer address')
	A										CUSAD2		3Ø						TEXT('Customer addres-line 2')
	A										CUSAD3		2Ø						TEXT('Customer address-line 3')
	A										{								
	A										}								
	A *																		
	A *										Fields used in the Item Master file								
	A *																		
	A										ITEMNO		6		Ø				COLHDG('Item number')
	A																		RANGE(100000 800000)
	A										DESCRP		3Ø						COLHDG('Item description')
	A										S								

only used for DFU etc.

→ 1 member with same name as file = default

CRTPF MBR(*FILE)

CRTPF MBR(*NONE) for a reference file

```
5714SS1 R03 M00 811204              DATA DESCRIPTION              FLDREF.IC38LIB          18/05/82

File name -        FLDREF.IC38LIB        Type of file -     PHYSICAL
Source file -      QDDSSRC.IC38LIB       Member -           FLDREF        18/05/82  10:51:31
Type of data -     *DATA
Options -          *SRC    *LIST
Authority -        *NORMAL
Text -             Field Reference File for IC38
Compiler -         IBM System/38 Data Description Processor

                                   DATA DESCRIPTION SOURCE

SEQNBR  *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ... ... 8  DATE

   100       A*
   200       A*FIELD REFERENCE FILE
   300       A*
   400       A          R REFFMT                    TEXT('FIELD REFERENCE FILE FORMAT')
   500       A*
   600       A*FIELDS USED IN THE CUSTOMER MASTER FILE
   700       A*
   800       A********** R CUSFMT                    TEXT('CUSTOMER MASTER FILE FORMAT')  18/05/82
   900       A            CUSTNO         6  0       COLHDG('CUSTOMER NUMBER')
  1000       A            CUSNAM        30         COLHDG('CUSTOMER NAME')
  1100       A            CUSAD1        40         COLHDG('CUSTOMER ADDRESS')
  1200       A            CUSAD2        30         TEXT('CUSTOMER ADDRESS - LINE 2')
  1300       A            CUSAD3        20         TEXT('CUSTOMER ADDRESS - LINE 3')
  1400       A            CUSVAT        10  0      COLHDG('VAT REG.NUMBER')
  1500       A            CSLSMN         6  0      COLHDG('SALESMAN NUMBER')
  1600       A            CRLMIT         9  2      COLHDG('CREDIT LIMIT')
  1700       A                                     EDTCDE(J £)
  1800       A            CURBAL         9  2      COLHDG('CURRENT BALANCE')
  1900       A                                     EDTCDE(J £)
  2000       A            LSTPMT         9  2      COLHDG('LAST PAYMENT' 'AMOUNT')
  2100       A                                     EDTCDE(J £)
  2200       A            PMTDTE         6  0      COLHDG('LAST PAYMENT' 'DATE')
  2300       A                                     EDTCDE(Y)
  2400       A            CUSCDE         6         COLHDG('SEARCH CODE')
  2500       A*
  2600       A*FIELDS USED IN THE ITEM MASTER FILE
  2700       A*
  2800       A********** R ITMFMT                    TEXT('ITEM MASTER FILE FORMAT')      18/05/82
  2900       A*                                                                           18/05/82
  3000       A            ITEMNO         6  0      COLHDG('ITEM NUMBER')
  3100       A                                     RANGE(100000 800000)
  3200       A            DESCRP        30         COLHDG('ITEM DESCRIPTION')
  3300       A*ETC. ETC. ETC.

                        *****  E N D   O F   S O U R C E  *****

5714SS1 R03 M00 811204              DATA DESCRIPTION              CUSMASTP.QGPL           18/05/82

File name -        CUSMASTP.QGPL         Type of file -     PHYSICAL
Source file -      QDDSSRC.IC38LIB       Member -           CUSMASTP      06/01/81  10:04:20
Type of data -     *DATA
Options -          *SRC    *LIST
Authority -        *NORMAL
Text -             Customer Master Physical File for IC38
Compiler -         IBM System/38 Data Description Processor

                                   DATA DESCRIPTION SOURCE

SEQNBR  *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ... ... 8  DATE

   100       A*
   200       A*PHYSICAL CUSTOMER MASTER FILE
   300       A*
   400       A                                     UNIQUE
   500       A                                     REF(FLDREF)
   600       A          R CUSMASTF                 TEXT('CUSTOMER MASTER RECORD FORMAT+
   700       A                                     ')
   800       A            CUSTNO     R
   900       A            CUSNAM     R
  1000       A            CUSAD1     R
  1100       A            CUSAD2     R
  1200       A            CUSAD3     R
  1300       A            CUSVAT     R
  1400       A            CSLSMN     R
  1500       A            CRLMIT     R
  1600       A            CURBAL     R
  1700       A            LSTPMT     R
  1800       A            PMTDTE     R
  1900       A            CUSCDE     R
  2000       A*
  2100       A*KEY FIELD SPECIFICATION FOLLOWS
  2200       A*
  2300       A          K CUSTNO
```

```
5714SS1 R03 M00 811204              DATA DESCRIPTION              CUSMASTP.QGPL          18/05/82 10:56:50
                                    EXPANDED SOURCE
                                                                                  FIELD  BUFFER POSITION
SEQNBR  *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ... ... 8   LEN     OUT     IN
   400                                      UNIQUE
   600                R CUSMASTF            TEXT('CUSTOMER MASTER RECORD FORMAT+
   600                                      ')
   800                  CUSTNO       6P 0                                          4       1
          * Ref: Fld - CUSTNO     Fmt - REFFMT     File - FLDREF     Lib - IC38LIB
   800                                      TEXT('CUSTOMER NUMBER                 -
   800                                                           ') COLHDG('CUS+
   800                                      TOMER NUMBER')
   900                  CUSNAM       30A                                          30      5
          * Ref: Fld - CUSNAM     Fmt - REFFMT     File - FLDREF     Lib - IC38LIB
   900                                      TEXT('CUSTOMER NAME                   -
   900                                                           ') COLHDG('CUS+
   900                                      TOMER NAME')
  1000                  CUSAD1       40A                                          40      35
          * Ref: Fld - CUSAD1     Fmt - REFFMT     File - FLDREF     Lib - IC38LIB
  1000                                      TEXT('CUSTOMER ADDRESS                -
  1000                                                           ') COLHDG('CUS+
  1000                                      TOMER ADDRESS')
  1100                  CUSAD2       30A                                          30      75
          * Ref: Fld - CUSAD2     Fmt - REFFMT     File - FLDREF     Lib - IC38LIB
  1100                                      TEXT('CUSTOMER ADDRESS - LINE 2       -
  1100                                                           ') COLHDG('CUS+
  1100                                      AD2')
  1200                  CUSAD3       20A                                          20      105
          * Ref: Fld - CUSAD3     Fmt - REFFMT     File - FLDREF     Lib - IC38LIB
  1200                                      TEXT('CUSTOMER ADDRESS - LINE 3       -
  1200                                                           ') COLHDG('CUS+
  1200                                      AD3')
  1300                  CUSVAT       10P 0                                         6       125
          * Ref: Fld - CUSVAT     Fmt - REFFMT     File - FLDREF     Lib - IC38LIB
  1300                                      TEXT('VAT REG.NUMBER                  -
  1300                                                           ') COLHDG('VAT-
  1300                                      REG.NUMBER')
  1400                  CSLSMN       6P 0                                          4       131
          * Ref: Fld - CSLSMN     Fmt - REFFMT     File - FLDREF     Lib - IC38LIB
  1400                                      TEXT('SALESMAN NUMBER                 -
  1400                                                           ') COLHDG('SAL+
  1400                                      ESMAN NUMBER')
  1500                  CRLMIT       9P 2                                          5       135
          * Ref: Fld - CRLMIT     Fmt - REFFMT     File - FLDREF     Lib - IC38LIB
  1500                                      TEXT('CREDIT LIMIT                    -
  1500                                                           ') EDTCDE(J £)
  1500                                      COLHDG('CREDIT LIMIT')
  1600                  CURBAL       9P 2                                          5       140
          * Ref: Fld - CURBAL     Fmt - REFFMT     File - FLDREF     Lib - IC38LIB
  1600                                      TEXT('CURRENT BALANCE                 -
  1600                                                           ') EDTCDE(J £)
  1600                                      COLHDG('CURRENT BALANCE')
  1700                  LSTPMT       9P 2                                          5       145
          * Ref: Fld - LSTPMT     Fmt - REFFMT     File - FLDREF     Lib - IC38LIB
  1700                                      TEXT('LAST PAYMENT AMOUNT             -
  1700                                                           ') EDTCDE(J £)
  1700                                      COLHDG('LAST PAYMENT' 'AMOUNT')

5714SS1 R03 M00 811204              DATA DESCRIPTION              CUSMASTP.QGPL          18/05/82 10:56:50
                                    EXPANDED SOURCE
                                                                                  FIELD  BUFFER POSITION
SEQNBR  *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ... ... 8   LEN     OUT     IN
  1800                  PMTDTE       6P 0                                          4       150
          * Ref: Fld - PMTDTE     Fmt - REFFMT     File - FLDREF     Lib - IC38LIB
  1800                                      TEXT('LAST PAYMENT DATE               -
  1800                                                           ') EDTCDE(Y) C+
  1800                                      OLHDG('LAST PAYMENT' 'DATE')
  1900                  CUSCDE       6A                                            6       154
          * Ref: Fld - CUSCDE     Fmt - REFFMT     File - FLDREF     Lib - IC38LIB
  1900                                      TEXT('SEARCH CODE                     -
  1900                                                           ') COLHDG('SEA+
  1900                                      RCH CODE')
  2300                K CUSTNO

          * * * * *  E N D  O F  E X P A N D E D  S O U R C E  * * * * *
```

give myths of every field

Field definitions given in a physical file record format (probably by reference to a field reference file) are carried forward into logical files, and into programs which use them, so that programs using the external description of a file do not need to define fields again.

Text descriptions of fields (and record formats) will also appear in compilation listings. Editing and validation specifications, however, have no effect on the way in which data base files are processed in programs.

The following keywords affect the use of fields in display files, if specified in a field reference file.

CHECK - CMP - RANGE - VALUES - EDTCDE- EDTWRD

The Screen Design Aid will pick up the use of these keywords from a referenced data base file, and include them in the specification of a field in a display file format, and allow their modification. It will also allow the use of column headings specified for fields (the COLHDG keyword). Otherwise column headings are only used by the Interactive Database Utilities DFU and Query.

Any specifications as to the use of fields in display files given in a field reference file can be overridden by DDS for a display file record format in which a field is used.

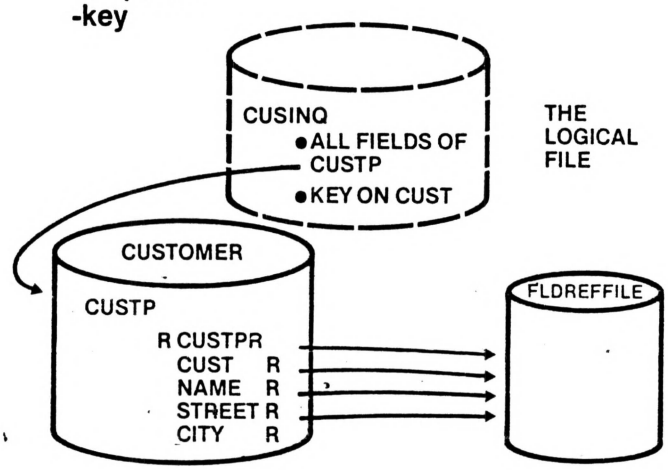
o Logical Files

The logical file defines the way in which we use physical file data records, both by selecting and grouping them, and by specifying the sequence in which they are to be processed or randomly accessed by key.

Logical Files

- Defined to the system as
- Selection of fields from physical records
- Access to data
- Definition of (logical)
  - -File by name
  - -Record by name
  - -Fields by name (optional)
  - -Access by
    - -sequence
    - -key

See 6.13



Data Description Specifications

IBM International Business Machines Corporation

DATA DESCRIPTION SPECIFICATIONS

File ORDVAL1L | Keying Instruction | Graphic Key | Description | Page of

Programmer | Date

A

Sequence Number	Form Type	Name	Length	Location Line	Location Pos	Functions
	A *					
	A *	LOGICAL FILE OF ORDERS BY DATE/VALUE				
	A *					
	A	R ORDVALIF				TEXT('ORDERS BY DATE/VALUE')
	A					PFILE(ORDHDRP)
	A	ORDER				
	A	CUST				
	A	ORDTOT				
	A	ORDDAY				
	A	ORDMTH				
	A	ORDYR				
	A	ORDATE				CONCAT(ORDDAY ORDMTH ORDYR)
	A	STATUS				
	A *					
	A	K ORDYR				
	A	K ORDMTH				
	A	K ORDDAY				
	A	K ORDTOT				
	A	S STATUS				CMP(EQ 2)
	A	S STATUS				CMP(EQ 4)

overhead = ±3K / file
multi member file: each member has own access path (PF or LF)
key (*none) to process all members in a file (col 7 = *all)

IBM Education

SF38 6.13

DATA DESCRIPTION SPECIFICATIONS

International Business Machines Corporation

GX21-7754-1 UM 050*
Printed in U.S.A.

Keying Instruction	Graphic / Key	Description	Page of

```
A*      ITEML1   LOGICAL ITEM FILE BY ITEM NO
A*
A                                        UNIQUE
A       R ITEMPR                         PFILE(ITEMP)
A                                        TEXT('LOGICAL ITEM FILE BY ITEM NO')
A       K ITEM
```

P for physical to distinguish it from logical. not nec.

The PFILE keyword defines the physical file providing data records. If the logical file is to use all the fields in the physical file record format, without any changes, no fields need be specified (just the physical file record format name).

The access path specifications follow the record format. Key fields must be present in the physical file record, and in the logical record format, when a selection of fields from the physical file format is made by naming them. Records can be selected or omitted based on the value of any field in the physical record. Selection and omission criteria can be specified in 'AND' and 'OR' relationships.

PF1 [A|B|C|D|E]   LF1 [A|C|E] key = A

if unique not specified, there can be dup recds in PF1. But if we construct LF2 [C|D|E] key = C with unique and there are dupl. C keys in PF1, then creation will terminate abnormally. If not, but if later LF1 adds dupl "C"'s, problem, hence —

Common keywords: see the DDS reference manual for a complete discussion.

MAINT (*IMMED)

ACCPTH If a logical file is to provide the same records from one or more physical files in the same sequence as another file which already exists, (but is to use a different record format), then it can share the access path of that file. This reduces system overhead in building and maintaining duplicate access paths. You cannot provide additional select/omit or key specifications when sharing access paths.

from R.T, automatic sharing of identical A.P.

CONCAT Fields from a physical file record can be joined to form a new field in a logical file record (but they cannot be sub-divided in a logical record).

DESCEND Gives a descending sequence on the value of this field.

UNIQUE Note that duplicate keys will not be allowed in the physical file if any of the using logical files specifies UNIQUE on their particular key.

RENAME : give other name to fld in LF than it was in PF

Entering the DDS source
The same as for a physical file except the type is *LF.

Creating the file
:CRTLF ORDVAL1L

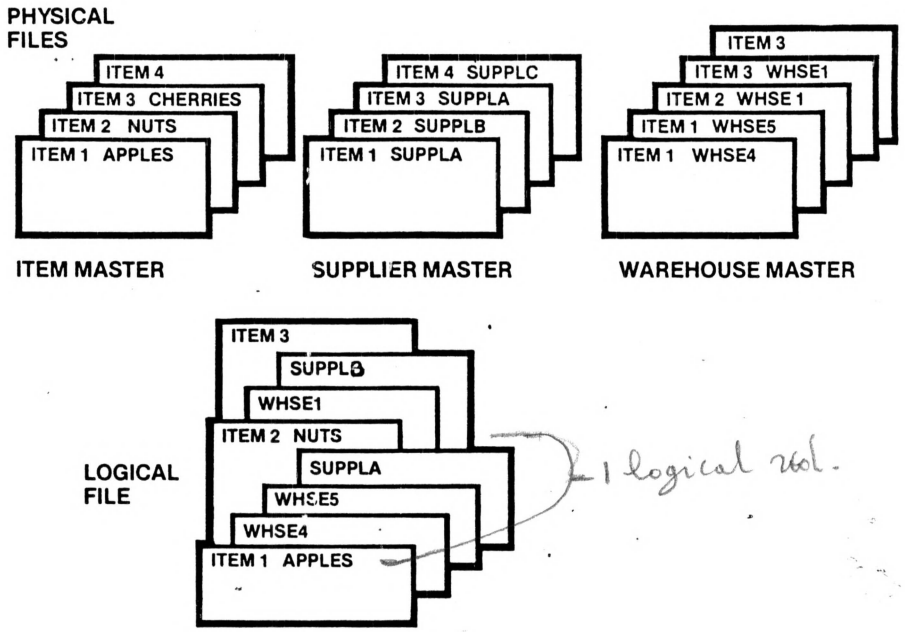
Loading the file
Why isn't this necessary?

if 2 ≠ LF have same AP, AP can be shared
ACC

FORMAT can be shared too (eg ≠ AP for 2 LF but with same recds)

o Logical files using two or more physical files

Records from up to 32 physical files can be presented through a single logical file. If a common set of key fields is specified, the effect is similar to RPG's matching record function. Records from the files are merged on their key value (not joined).*



The logical file DDS must specify a different record format and PFILE keyword for each physical file which provides records. Records are presented in sequence by complete key. When keys match from different files, the order of specification of formats on the DDS form is used by the system to sequence the records.

* R.7: joined PF

IBM International Business Machines Corporation

DATA DESCRIPTION SPECIFICATIONS

File: Full Item details | Keying Instruction: Graphic / Key | Description

Programmer: MN | Date: 30.5

A

Sequence Number	Form Type	Conditioning: And/Or/Comment (A/O/*)	Conditioning: Condition Name: Not	Indicator	Not	Indicator	Not	Indicator	Name Type (B/R/K/S/O)	Reserved	Name	Reference (R)	Length	Data Type (B/A/P/S/B/A/S/X/Y/N/I/W)	Decimal Positions	Usage (B/O/I/B/H/M)	Location: Line	Location: Pos	Functions
	A	*									Logical file providing complete details of all items - ITMDETIL.								
	A	*																	
	A								R		ITMFMT								PFILE(ITMMASTP)
	A								K		ITGMNO								
	A	*																	
	A								R		WSEFMT								PFILE(WSEMASTP)
	A								K		ITGMNO								
	A								K		WHSEID								
	A	*																	
	A								R		SUPFMT								PFILE(SUPMASTP)
	A								K		ITGMNO								
	A																		
	A																		
	A																		
	A																		
	A																		
	A																		
	A																		
	A																		
	A																		

*Number of sheets per pad may vary slightly.

common key across the 3 files (R7: joined files won't need common key)

ITEM

- 01 APPLES
- 02 BANANAS
- 03 CARROTS
- 04 DATES

SUPPLIER

- 01 TOM
- 02 RANJIT
- 03 SAM
- 04 BERTHA

WAREHOUSE

- 01 WHSEA
- 01 WHSEB
- 02 WHSEF
- 02 WHSEA
- 03 WHSEC

- 03 CARROTS
  - 02 RANJIT
    - 02 WHSEF
    - 02 WHSEA
- 02 BANANAS
  - 01 TOM
    - 01 WHSEB
    - 01 WHSEA
- 01 APPLES

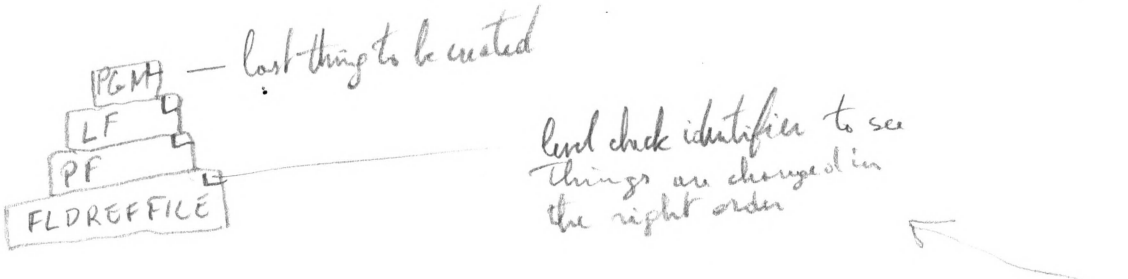
o The file create commands

The Create Physical File (CRTPF) and Create Logical File (CRTLF) commands expect DDS source statements from a source file member to provide the record format and access path specifications for the file. There are certain keywords in the commands which are of particular importance

File Creation Considerations

Data Base

- **Commands**
  - **CRTPF**
  - **CRTLF**
  - **CONTROL OF SIZE**
  - **FORCE RATIO**
  - **LEVEL CHECKING**
  - **ACCESS PATH MAINTENANCE**
  - **ACCESS PATH RECOVERY**



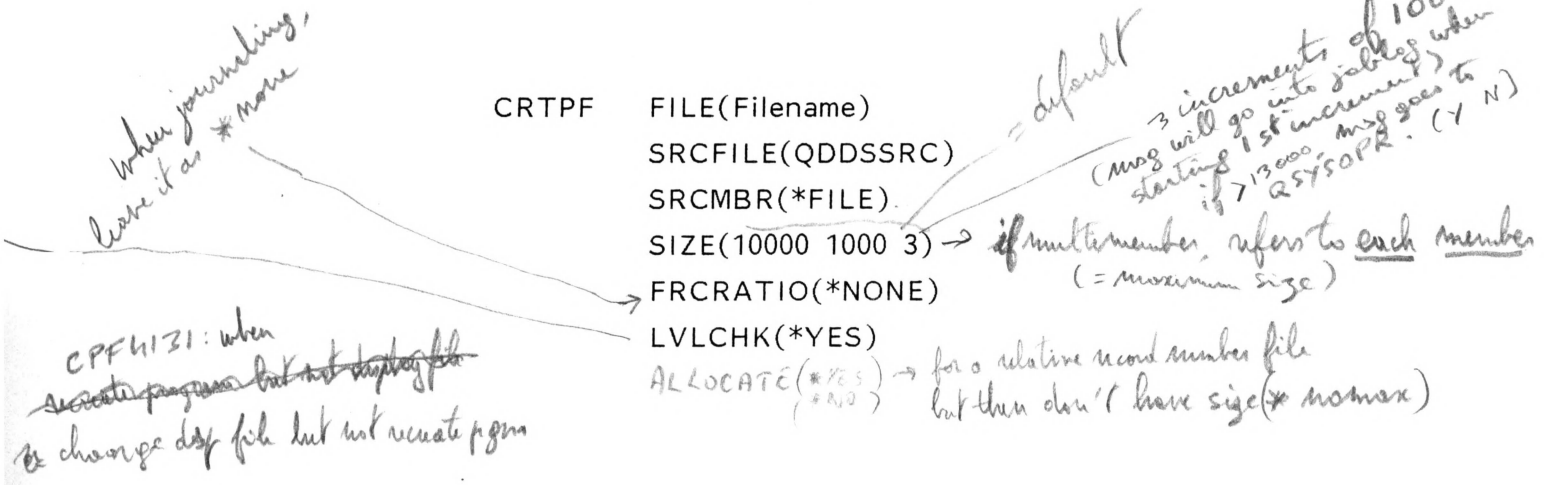
Create Physical File

The command identifies the name of the file to be created and the library (by default QGPL). It also identifies the source file and its library (the default is QDDSSRC) and the source member (which is by default the member with the same name as the file to be created).

The default size per file member is 10,000 records, which may be followed by 3 increments of 1,000 records before immediate warnings are given to the user.

The keyword FRCRATIO determines how many records can be added, deleted or updated before any changed pages of the file in main memory are written out when required by the system. Transaction or log files should be given a force ratio of 1, unless Journalling is implemented.
Other (master) files can take the default of 'no limit', because any update to them is represented by transaction file records.

Level checking is the process of ensuring that the program contains the latest level of external data descriptions for the files used. When a file is created, it is date/time stamped. When a program is compiled, this date/time stamp and a format-id is copied into it. If a file's description is changed following compilation of a program using it, it is highly likely that the program will no longer work. Level checking will ensure that a program does not execute using an incorrect format.



Create Logical File

```
CRTLF     FILE(Filename)
          SRCFILE(QDDSSRC)
          SRCMBR(*FILE)
          MAINT(*IMMED)
          RECOVER(*AFTSTRCPF)
          FRCRATIO(*NONE)
          LVLCHK(*YES)
```

Two keywords of importance here are MAINT and RECOVER.

MAINT relates to access path maintenance of files NOT open, the default being immediate, ie. any change taking place in the physical file or files over which the access path is built which require a change to the access path, is to be actioned immediately. The alternative is for the access path to be updated or rebuilt when the file is opened (*DLY, *REBLD). Any open file will have its access path updated immediately regardless of this parameter's value.

RECOVER, refers to the rebuilding of an access path following an abnormal system termination, and the timing of that rebuilding. An access path found flagged as out of synchronisation with the physical data will be rebuilt as soon as the system next comes up, or when the file is first opened following start CPF.

Regarding LVLCHK, changes in the format of a Physical file can affect the Logical files referencing it. However, if the format of the Logical file is unchanged, then it is unnecessary to recompile the program using it. For further information, see CPF Programmers Guide, Chapter 4, pages 61 to 67.

Physical File Members

There are some commands that relate to data base file members of which you should be aware:

ADDPFM (Add a Physical File Member) SEU uses this and automatically

Adds a member to a file to allow insertion of records. (SEU uses this command when creating a new source member.) Each member has its own access path.

R7: → BRWPFM - display any rcd in PFM (can be in hex)

RMVM (Remove a file member)

Also frees the storage occupied by the records in the member. relative rcds number

INZPFM (Initialise Physical File Member) for a RRN file

This enables a file to be used as a direct (RRN) processing. Numeric fields are zeroised and character fields are blanked.

CLRPFM (Clear Physical File Member)

As RMVM but member remains for subsequent use.

RGZPFM (Reorganize Physical File Member) to improve paging, grouping rcds together

This command enables you to get rid of deleted records and organize the records in the member to a sequence of either the physical file's keyed sequence or to the sequence of a logical file scoped over the member. The records are physically rearranged in storage and this is typically used before an override is issued for the file which specifies NBRRCDS(x).

RNMM (Rename Member).

Exercise 7

1. Code the description of a Customer File as a physical file, with the following fields (which are already defined in the Field Reference File, names as shown):

- Customer Number	CUST
- Name	NAME
- Address Line 1	CUSAD1
- Address Line 2	CUSAD2
- Address Line 3	CUSAD3
- Ship Via Code	SVIA
- Payment Method	PAYM
- Discount Code	DISC
- Credit Limit	CRLIM
- Current Balance	BALANC
- Total Business year to date	YRTOTL
- Last Order No.	LORDER
- Update Count	UPDATC
- Search Name	NAMARG

The format name is CUSTPR.

2. Code the descripton of a logical file based on the physical Customer File defined above. Specify that the physical file is in the library SF38LIB. This logical file is to be keyed on customer number.

3. Create the files in your library SF38TEAMxx using these names:

Physical file	CUSTP
Logical file	CUSTL1

4. Create a second logical file, identical to the first, with the name CUSTLTxx

FLDREFFILE

```
SEQUENCE ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8

    1.00  0001 A* FIELD REFERENCE FILE FOR ORDER ENTRY APPLICATION
    2.00  0002 A**          FLDREFFILE                ORDER ENTRY FIELD REFERENCE FIL
    3.00  0003 A*************************************************************************
    4.00  0004 A          R ORDREFFILE                TEXT('FIELD REFERENCE FILE')
    5.00  0005 A            ACTIVE         1  0      COLHDG('ACTIVE') RANGE(0 2)
    6.00  0006 A                                     TEXT('ITEM STATUS')
    7.00  0007 A            BALANC         6  2                  COLHDG('BALANCE')
    8.00  0008 A                                     TEXT('ACCOUNT BALANCE')
    9.00  0009 A            CAMOUN         6  2      EDTCDE(K) COLHDG('CURR' 'VALUE')
   10.00  0010 A                                            TEXT('CURR. ORDER VALUE')
   11.00  0013 A            CRLIM          6  2                       COLHDG('CREDIT +
   12.00  0014 A                                     LIMIT') TEXT('CREDIT LIMIT')
   13.00  0085 A            CUSAD1        25        COLHDG('STREET')
   14.00  0086 A                                     TEXT('STREET')
   15.00  0011 A            CUSAD2        25        COLHDG('CITY')
   16.00  0012 A                                     TEXT('CITY')
   17.00  0113 A            CUSAD3        25        COLHDG('ZIP')
   18.00  0114 A                                     TEXT('ZIP CODE')
   19.00  0015 A            CUSORD        10        COLHDG('CUSTOMER' 'ORDERNO')
   20.00  0016 A                                     TEXT('CUSTOMER ORDER NO')
   21.00  0017 A            CUST           6  0     COLHDG('CUSTOMER-NO') CHECK(MF)
   22.00  0018 A                                     TEXT('CUSTOMER NO')
   23.00        A            DESARG         6        TEXT('SEARCH BY DESCRIPTION')
   24.00  0019 A            DESCR         30        COLHDG('DESCRIPTION')
   25.00  0020 A                                     TEXT('ITEM DESCRIPTION')
   26.00  0021 A            DISC           2  0     COLHDG('DISCOUNT') CHECK(MF) +
   27.00  0022 A                                     TEXT('DISCOUNT CODE')
   28.00  0023 A                                     RANGE(01 10)
   29.00  0024 A            DISCT     R             REFFLD(DISC *SRC)
   30.00  0025 A            ITDISC    R             REFFLD(DISC *SRC)  COLHDG('ITEM' 'D+
   31.00  0026 A                                     ISCNT') TEXT('ITEM DISCOUNT')
   32.00  0027 A            ITEM           6  0     COLHDG('ITEMNO') CHECK(MF)
   33.00  0028 A                                     TEXT('ITEM NO')
   34.00  0029 A            ITDSCT    R             REFFLD(ITDISC *SRC)
   35.00  0030 A            LINENO         2  0     COLHDG('LINENO') RANGE(1 99)
   36.00  0031 A                                     TEXT('LINE NO')
   37.00  0032 A            LINET     R             REFFLD(LINENO *SRC)
   38.00  0033 A            LORDER     00005  0     COLHDG('LAST ORDER')
   39.00  0034 A                                          TEXT('NO. OF LAST ORDER')
   40.00  0035 A            MONTH1         6  0                  COLHDG('LAST''MO+
   41.00  0036 A                                     NTH') TEXT('QUANTITY PREV.MONTH')
   42.00  0037 A            MONTH2    R             REFFLD(MONTH1 *SRC) COLHDG('TWO' 'M+
   43.00  0038 A                                     ONTHS' 'AGO')
   44.00  0039 A                                     TEXT('SALES MONTH BEFORE LAST')
   45.00  0040 A            MONTHC    R             REFFLD(MONTH1)     COLHDG('CURR' 'M+
   46.00  0041 A                                     ONTH') TEXT('QUANT SOLD CURR MONTH')
   47.00  0042 A            MSGTXT        60        TEXT('MESSAGE TEXT FIELD')
   48.00  0043 A            NAMARG         6        TEXT('SEARCH BY NAME ARGUMENT')
   49.00  0044 A            NAME          25        COLHDG('CUSTOMER' 'NAME')
   50.00  0045 A                                     TEXT('CUSTOMER NAME')
   51.00  0048 A            ONHAND         4  0     COLHDG('ONHAND') EDTCDE(Z)
   52.00  0049 A                                     TEXT('QUANTITY ON HAND')
   53.00  0050 A            ONORD          4  0     COLHDG('ON ORDER')
   54.00  0051 A                                     TEXT('QUANTITY ON ORDER')
   55.00  0052 A            ORDACT         1        TEXT('ORDER ACTIVE FOR RESTART')
   56.00        A            ORDDAY         2  0     COLHDG('ORDER' 'DAY')
   57.00  0053 A            ORDER          5  0     COLHDG('ORDERNO') CHECK(MF)
   58.00  0054 A                                     TEXT('ORDER NO')
   59.00  0055 A            ORDERT    R             REFFLD(ORDER *SRC)
   60.00        A            ORDMTH         2  0     COLHDG('ORDER' 'MONTH')
   61.00  0056 A            ORDTOT    R             REFFLD(CAMOUN *SRC) COLHDG('ORDER T+
   62.00  0057 A                                     OTAL') TEXT('TOTAL ORDER VALUE')
   63.00        A            ORDYR          2  0     COLHDG('ORDER' 'YEAR')
   64.00  0058 A            PAYM           2  0     COLHDG('PAYMENT') CHECK(MF) +
   65.00  0059 A                                     TEXT('PAYMENT METHOD CODE')
   66.00  0060 A                                     RANGE(01 20)
   67.00  0061 A            PAYMT     R             REFFLD(PAYM *SRC)
   68.00  0062 A            POS            2  0     COLHDG('POSITION')
   69.00  0063 A                                     TEXT('ORDER POSITION')
   70.00  0064 A            PRICE          5  2     COLHDG('PRICE') EDTCDE(K)
   71.00  0065 A                                     TEXT('ITEM PRICE')
   72.00  0066 A            PRICET    R             REFFLD(PRICE *SRC)
   73.00  0067 A            QTYSHP         4  0     COLHDG('QUANTITY' 'SHIPPED')
   74.00  0068 A                                     TEXT('QUANTITY SHIPPED')
   75.00  0069 A                                     EDTCDE(Z)
   76.00  0070 A            QUANT          4  0     COLHDG('QUANTITY') EDTCDE(Z)
   77.00  0071 A                                     TEXT('QUANTITY ORDERED')
   78.00  0080 A            SADLN1        25        COLHDG('SHIP TO' 'STREET')
   79.00  0081 A                                     TEXT('SHIP TO ADDRESS STREET')
   80.00  0091 A            SADLN3        25        COLHDG('SHIP TO' 'ZIP')
   81.00  0092 A                                     TEXT('SHIP TO ADDRESS ZIP CODE')
   82.00  0072 A            SADLN2        25        COLHDG('SHIP TO' 'CITY')
   83.00  0073 A                                     TEXT('SHIP TO ADDRESS CITY')
   84.00  0074 A            SINSTR        40        COLHDG('SHIPPING' 'INSTRUCTIONS')
   85.00  0075 A                                     TEXT('SHIPPING INSTRUCTIONS 1')
   86.00  0078 A            SNAME         25        COLHDG('SHIP TO' 'NAME')
   87.00  0079 A                                     TEXT('SHIP TO ADDRESS NAME')
   88.00  0082 A            STATT          1  0     RANGE(1 4)
   89.00  0083 A            STATUS    R             COLHDG('STATUS') REFFLD(STATT *SRC)
   90.00  0084 A                                     TEXT('ORDER STATUS')
   91.00  0087 A            SVIA           2  0     COLHDG('SHIP VIA') CHECK(MF) +
   92.00  0088 A                                     TEXT('SHIP VIA CODE')
   93.00  0089 A                                     RANGE(01 20)
   94.00  0090 A            SVIAT     R             REFFLD(SVIA *SRC)
   95.00  0093 A            TOTAL     R             COLHDG('TOTAL') REFFLD(ORDTOT *SRC)
   96.00  0094 A                                     TEXT('ORDER LINE VALUE')
   97.00  0095 A            UPDATC         3  0     COLHDG('UPDATE COUNT')
   98.00  0096 A                                     TEXT('CUST MASTER UPDATE COUNT')
   99.00  0097 A            UDATHD    R             REFFLD(UPDATC *SRC)
  100.00  0098 A                                     TEXT('ORDER HEADER UPDATE COUNT')
  101.00  0099 A            UPDATI    R             REFFLD(UPDATC *SRC)
  102.00  0100 A                                     TEXT('ITEM MASTER UPDATE COUNT')
  103.00  0101 A            UDATIL    R             REFFLD(UPDATC *SRC)
  104.00  0102 A                                     TEXT('ORDER LINE UPDATE COUNT')
  105.00  0103 A            WHSEQ          5  0     COLHDG('WARE HOUSE ' 'SEQUENCE')
  106.00  0104 A                                     TEXT('WAREHOUSE SEQUENCE CODE')
  107.00  0105 A            WSOPNA        10        COLHDG('WORKSTN' 'OPERATOR' 'NAME')
  108.00  0106 A                                     TEXT('USER NAME')
  109.00  0107 A            YRAMT          6  2     COLHDG('YEAR TOTAL')
  110.00  0108 A                                           TEXT('$ VALUE SOLD YTD')
  111.00  0109 A            YRQTY          6  0     COLHDG('YEAR' 'QUANTITY')
  112.00  0110 A                                     TEXT('QUANTITY SOLD YEAR TO DATE')
  113.00  0111 A            YRTOTL         6  2     COLHDG('YEAR TOTAL')
  114.00  0112 A                                          TEXT('TOTAL SALES YR TO DATE')
```

Exercise 8

The object of this exercise is for you to write a program using the DFU to key data into a newly created file.

Check with your instructor to see if it is alright to continue and then run the exercise using the run-book provided in appendix C.

FURTHER DISPLAY FUNCTIONS AND SDA

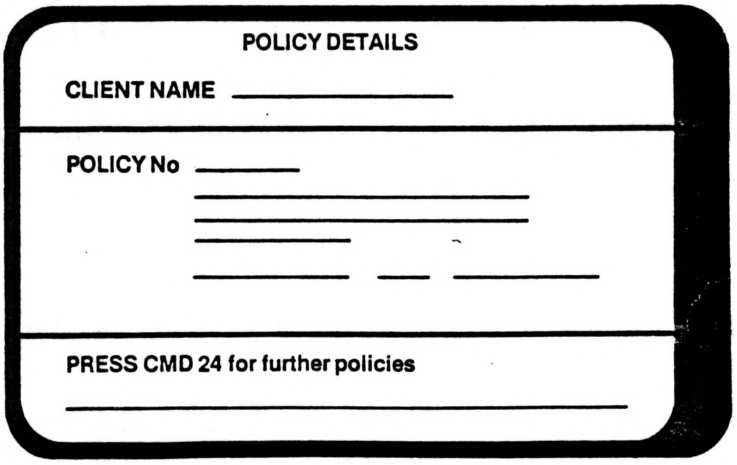
Using Multiple Display Record Formats
Other DDS Keywords
Using a display file in RPG and COBOL programs
Screen Design Aid

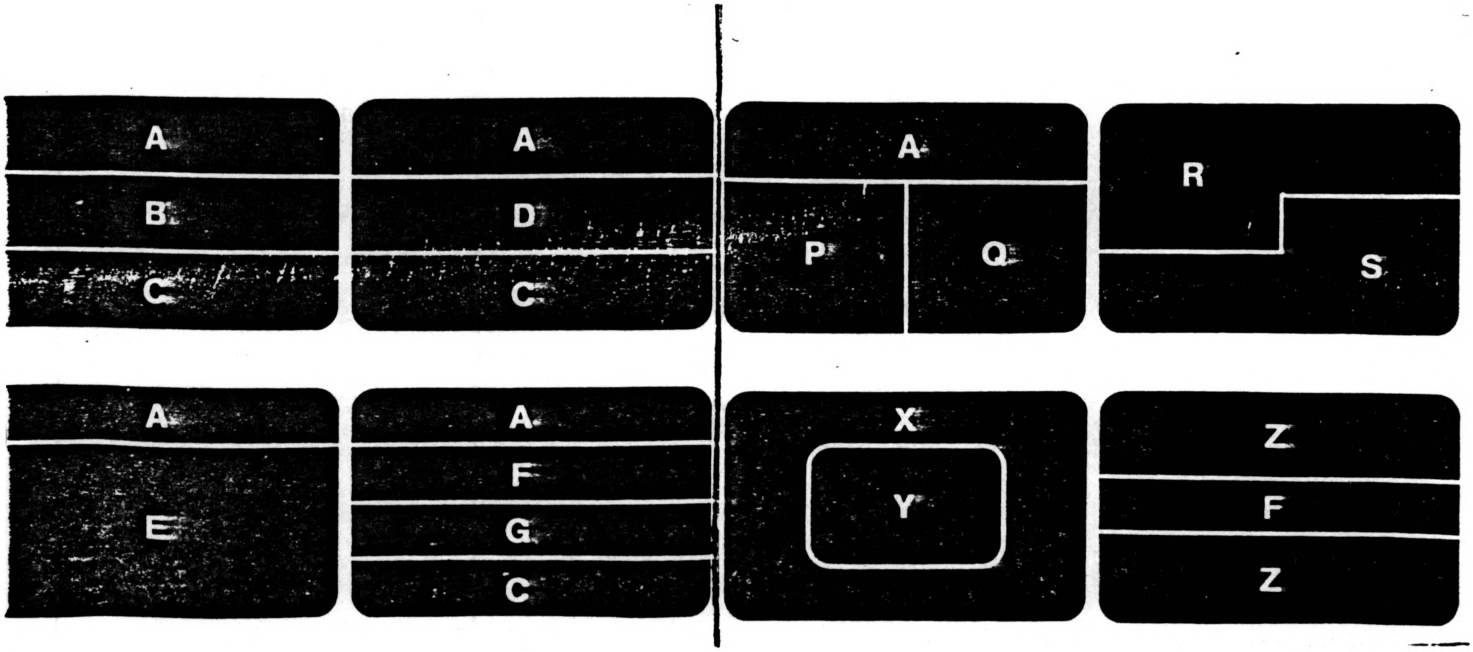
Exercise 9

o Using multiple display record formats

Only one display device file is normally used in a program. The program can send and receive specific record formats in the file by simply specifying the format name. Several record formats can be used for input from a display provided OVERLAY is specified. The SLNO (start line) keyword can specify either a literal of 1 to 24 or a variable start line, and used together with CLRL(*NO) (clear lines) keyword allows a variety of new screen layouts.

Using Multiple Record Formats





o Other DDS Keywords

For a complete discussion, look first at the CPF Programmer's Guide, chapter 7, and then the DDS Reference manual, chapter 4.

ALARM	sounds when the record is sent
AUTO(RAB)	automatically right-adjust and blank this field
AUTO(RAZ)	or zero-fill (when the Field Exit key is pressed)
AUTO(RA)	automatic record advance. Has the same effect as pressing the Enter key when this field is filled
KEEP/ASSUME	allows a program to leave a screen record format at a display when it terminates (normally the display is erased) and the next program can assume the record format is already there without having to write it first. (cf PRUF under CCP on the System/3)
HELP/CLEAR	lets you know (via an indicator) if these keys were pressed
MSGLOC	change the message line location from the system default, line 24
PRINT	allows the current record at the display to be printed (exactly as it looks on the display) 1 on an associated workstation printer on the same 5250 control unit, using a direct copy or 2 on any other device file defined to the system or 3 not printed but an indicator passed back to the program
UNLOCK	useful for fast data entry. The system will erase input fields and unlock the keyboard immediately it has read the data from the display and passed it to the program. The program can be processing the record while the operator is keying a new record.

o Using a display device file in an RPG or COBOL program

```
5714RG1 RPG   R04M01  820813     CUSINQG.IC38LIB     18/02/83  13:18:14  PAGE     2

SEQUENCE      1         2         3         4         5         6         7      IND    DO    LAST
NUMBER    6789012345678901234567890123456789012345678901234567890123456789012345  USE    NUM   UPDA


NAME OF PROGRAM WILL BE CUSINQG IN LIBRARY IC38LIB

          H

     100  FCUSBALD CF  E                    WORKSTN
     200  F*
           RECORD FORMAT(S):  FILE CUSBALD LIB OELIB
                     EXTERNAL FORMAT CUSBALF RPG NAME CUSBALF

     300  FCUSMASTLIF  E           K         DISK
           RECORD FORMAT(S):  FILE CUSMASTL LIB OELIB
                     EXTERNAL FORMAT CUSMASTF RPG NAME CUSMASTF

1000000    INPUT  FIELDS FOR RECORD CUSBALF FILE CUSBALD FORMAT CUSBALF        CUSTOMER BALANCE IN
1000001                                          1   1 *IN01                   END OF INQUIRY
1000002                                          2   2 *IN90                   NO SUCH CUSTOMER EX
1000003                                          3   8CUSNUM
2000000    INPUT  FIELDS FOR RECORD CUSMASTF FILE CUSMASTL FORMAT CUSMASTF     CUSTOMER MASTER REC
2000001                                        P   1   40CUSTNO               CUSTOMER NUMBER
2000002                                            5  34 CUSNAM               CUSTOMER NAME
2000003                                           35  74 CUSAD1               CUSTOMER ADDRESS
2000004                                           75 104 CUSAD2               CUSTOMER ADDRESS - L
2000005                                          105 124 CUSAD3               CUSTOMER ADDRESS - L
2000006                                        P 125 1300CUSVAT               VAT REG.NUMBER
2000007                                        P 131 1340CSLSMN               SALESMAN NUMBER
2000008                                        P 135 1392CRLMIT               CREDIT LIMIT
2000009                                        P 140 1442CURBAL               CURRENT BALANCE
2000010                                        P 145 1492LSTPMT               LAST PAYMENT AMOUNT
2000011                                        P 150 1530PMTDTE               LAST PAYMENT DATE
2000012                                          154 159 CUSCDE               SEARCH CODE
     400  C             START     TAG
     500  C*
     600  C*SEND THE RECORD FORMAT TO THE WORK STATION
     700  C*WHICH CALLED THIS PROGRAM
     800  C*
     900  C                      EXFMTCUSBALF
    1000  C*
    1100  C*IF CMD 1 WAS PRESSED,WE'VE FINISHED
    1200  C*
    1300  C   01                 GOTO RETURN
    1400  C*
    1500  C*GET CUSTOMER DETAILS,AND DISPLAY THEM
    1600  C*
    1700  C             CUSNUM    CHAINCUSMASTF             90                  1
    1800  C*                                                                                18/05
    1900  C*IF IND. 90 IS ON,THE ERROR MESSAGE WILL BE DISPLAYED                           18/05
    2000  C*                                                                                18/05
    2100  C                      GOTO START
    2200  C*
    2300  C*RETURN TO CALLING PROGRAM
    2400  C*
    2500  C             RETURN    TAG
    2600  C                      SETON                     LR                  1
3000000    OUTPUT  FIELDS FOR RECORD CUSBALF FILE CUSBALD FORMAT CUSBALF       CUSTOMER BALANCE IN
3000001                          *IN90      1  CHAR    1                      NO SUCH CUSTOMER EX
3000002                          CUSTNO     7  ZONE  6,0                      CUSTOMER NUMBER
3000003                          CUSNAM    37  CHAR   30                      CUSTOMER NAME
3000004                          CURBAL    46  ZONE  9,2                      CURRENT BALANCE
```

automatically generated

```
5714CB1 R04 M01 820813              COBOL SOURCE LISTING                    10/05/83 13:47:28

STMT SEQNBR -A 1 B.. ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 .IDENTFCN S  COPYNAME   CHG/DATE

   1  000100 IDENTIFICATION DIVISION.
      000200*
   2  000300 PROGRAM-ID.    CUSINQIC.                                                      10/05/83
   3  000400 AUTHOR.        BRUCE WASSELL.                                                 10/05/83
   4  000500 INSTALLATION.  BASINGSTOKE EDUCATION CENTRE.                                  10/05/83
   5  000600 DATE-COMPILED.  10/05/83 13:47:28    .                                       10/05/83
      000700*
   6  000800 ENVIRONMENT DIVISION.
      000900*
   7  001000 CONFIGURATION SECTION.
      001100*
   8  001200 SOURCE-COMPUTER. IBM-S38.
   9  001300 OBJECT-COMPUTER. IBM-S38.
      001400*
  10  001500 INPUT-OUTPUT SECTION.
      001600*
  11  001700 FILE-CONTROL.
  12  001800     SELECT CUSTOMER-FILE
  13  001900         ASSIGN TO DATABASE-CUSMASTL                                          09/05/83
  14  002000         ORGANIZATION IS INDEXED
  15  002100         ACCESS MODE IS RANDOM
  16  002200         RECORD KEY IS CUSTNO OF CUSMASTF.
  17  002300     SELECT DISPLAY-FILE
  18  002400         ASSIGN TO WORKSTATION-CUSBALD-SI
  19  002500         ORGANIZATION IS TRANSACTION
  20  002600         ACCESS MODE IS SEQUENTIAL.
      002700*
  21  002800 DATA DIVISION.
      002900*
  22  003000 FILE SECTION.
      003100*
  23  003200 FD  CUSTOMER-FILE
  24  003300     LABEL RECORDS ARE STANDARD.
      003400*
  25  003500 01  CUSTOMER-RECORD.
  26  003600     COPY DDS-CUSMASTF OF CUSMASTL.
  28 +000001                                                                        CUSMASTF
     +000002*    I-O FORMAT:CUSMASTF   FROM FILE CUSMASTL   OF LIBRARY BRUCELIB    CUSMASTF
     +000003*                          CUSTOMER MASTER RECORD FORMAT              CUSMASTF
     +000004*THE KEY DEFINITIONS FOR RECORD FORMAT  CUSMASTF                       CUSMASTF
     +000005*  NUMBER              NAME              RETRIEVAL     TYPE    ALTSEQ  CUSMASTF
     +000006*   0001            CUSTNO                ASCENDING      N       NO    CUSMASTF
  29 +000007       05  CUSMASTF.                                                   CUSMASTF
  30 +000008           06 CUSTNO          PIC S9(6)        COMP-3.                CUSMASTF
     +000009*                         CUSTOMER NUMBER                            CUSMASTF
  31 +000010           06 CUSNAM          PIC X(30).                             CUSMASTF
     +000011*                         CUSTOMER NAME                              CUSMASTF
  32 +000012           06 CUSAD1          PIC X(40).                             CUSMASTF
     +000013*                         CUSTOMER ADDRESS                           CUSMASTF
  33 +000014           06 CUSAD2          PIC X(30).                             CUSMASTF
     +000015*                         CUSTOMER ADDRESS - LINE 2                  CUSMASTF
  34 +000016           06 CUSAD3          PIC X(20).                             CUSMASTF
     +000017*                         CUSTOMER ADDRESS - LINE 3                  CUSMASTF
  35 +000018           06 CUSVAT          PIC S9(10)       COMP-3.               CUSMASTF
     +000019*                         VAT REG.NUMBER                             CUSMASTF
```

```
5714-CB1 R04 M01 920313        COBOL SOURCE LISTING        CUSINQIC        10/05/83 13:47:21

STMT SEQNBR -A 1 B.. ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 .IDENTFCN S COPYNAME   CHG DATE

 36 +000020            06 CSLSMN            PIC S9(6)          COMP-3.                 CUSMASTF
    +000021*                           SALESMAN NUMBER                                 CUSMASTF
 37 +000022            06 CRLMIT            PIC S9(7)V9(2)     COMP-3.                 CUSMASTF
    +000023*                           CREDIT LIMIT                                    CUSMASTF
 38 +000024            06 CURBAL            PIC S9(7)V9(2)     COMP-3.                 CUSMASTF
    +000025*                           CURRENT BALANCE                                 CUSMASTF
 39 +000026            06 LSTPMT            PIC S9(7)V9(2)     COMP-3.                 CUSMASTF
    +000027*                           LAST PAYMENT AMOUNT                             CUSMASTF
 40 +000028            06 PMTDTE            PIC S9(6)          COMP-3.                 CUSMASTF
    +000029*                           LAST PAYMENT DATE                               CUSMASTF
 41 +000030            06 CUSCDE            PIC X(6).                                  CUSMASTF
    +000031*                           SEARCH CODE                                     CUSMASTF
     003700*
 42  003800 FD  DISPLAY-FILE
 43  003900     LABEL RECORDS ARE OMITTED.
     004000*
 44  004100 01  DISPLAY-RECORD.
 45  004200     COPY DDS-ALL-FORMATS OF CUSBALD.
 47 +000001       05  CUSBALD-RECORD PIC X(46).                                       <-ALL-FMTS
 48 +000002                                                                            <-ALL-FMTS
    +000003*  INPUT FORMAT:CUSBALF      FROM FILE CUSBALD    OF LIBRARY BRUCELIB       <-ALL-FMTS
    +000004*                            CUSTOMER BALANCE INQUIRY                       <-ALL-FMTS
 49 +000005       05  CUSBALF-I       REDEFINES CUSBALD-RECORD.                        <-ALL-FMTS
 50 +000006           06 CUSNUM             PIC S9(6).                                 <-ALL-FMTS
 51 +000007                                                                            <-ALL-FMTS
    +000008* OUTPUT FORMAT:CUSBALF      FROM FILE CUSBALD    OF LIBRARY BRUCELIB       <-ALL-FMTS
    +000009*                            CUSTOMER BALANCE INQUIRY                       <-ALL-FMTS
 52 +000010       05  CUSBALF-O       REDEFINES CUSBALD-RECORD.                        <-ALL-FMTS
 53 +000011           06 CUSTNO             PIC S9(6).                                 <-ALL-FMTS
    +000012*                            CUSTOMER NUMBER                                <-ALL-FMTS
 54 +000013           06 CUSNAM             PIC X(30).                                 <-ALL-FMTS
    +000014*                            CUSTOMER NAME                                  <-ALL-FMTS
 55 +000015           06 CURBAL             PIC S9(7)V9(2).                            <-ALL-FMTS
    +000016*                            CURRENT BALANCE                                <-ALL-FMTS
     004300*
 56  004400 WORKING-STORAGE SECTION.
     004500*
 57  004600 01  DISPLAY-FILE-INDICATORS.                                                   09/05/83
     004700*                                                                               09/05/83
 58  004800     05  END-OF-INQUIRY           PIC 1   VALUE B"0"   INDICATOR 01.            09/05/83
 59  004900         88  END-OF-INQUIRY-REQUESTED     VALUE B"1".                           09/05/83
 60  005000     05  INVALID-ENTRY            PIC 1   VALUE B"0"   INDICATOR 90.            09/05/83
 61  005100         88  VALID-CUSTOMER-NUMBER        VALUE B"0".                           09/05/83
 62  005200         88  INVALID-CUSTOMER-NUMBER      VALUE B"1".                           09/05/83
     005300*                                                                               09/05/83
 63  005400 PROCEDURE DIVISION.                                                            09/05/83
     005500*                                                                               09/05/83
     005600 000-DISPLAY-CUSTOMER-BALANCE.                                                  09/05/83
     005700*                                                                               09/05/83
 64  005800     OPEN INPUT CUSTOMER-FILE                                                   09/05/83
     005900          I-O   DISPLAY-FILE.                                                   09/05/83
 65  006000     MOVE ZERO TO CUSTNO OF CUSBALF-O, CURBAL OF CUSBALF-O.                     10/05/83
 66  006100     MOVE SPACES TO CUSNAM OF CUSBALF-O.                                        10/05/83
 67  006200     PERFORM 100-EXECUTE-DISPLAY-RECORD                                         09/05/83
     006300         UNTIL END-OF-INQUIRY-REQUESTED.                                        09/05/83
 68  006400     CLOSE CUSTOMER-FILE                                                        09/05/83
     006500           DISPLAY-FILE.                                                        09/05/83
 69  006600     STOP RUN.                                                                  09/05/83
     006700*                                                                               09/05/83
     006800 100-EXECUTE-DISPLAY-RECORD.                                                    09/05/83
     006900*                                                                               09/05/83
 70  007000     WRITE DISPLAY-RECORD                                                       09/05/83
     007100         FORMAT IS "CUSBALF"                                                    09/05/83
     007200         INDICATORS ARE DISPLAY-FILE-INDICATORS.                                09/05/83
 71  007300     READ DISPLAY-FILE                                                          09/05/83
     007400         FORMAT IS "CUSBALF"                                                    09/05/83
     007500         INDICATORS ARE DISPLAY-FILE-INDICATORS.                                09/05/83
 72  007600     MOVE CUSNUM OF CUSBALF-I TO CUSTNO OF CUSMASTF.                            09/05/83
 73  007700     READ CUSTOMER-FILE                                                         09/05/83
     007800         FORMAT IS "CUSMASTF"                                                   09/05/83
 74  007900         INVALID KEY SET INVALID-CUSTOMER-NUMBER TO TRUE.                       09/05/83
 75  008000     IF  VALID-CUSTOMER-NUMBER                                                  09/05/83
 76  008100         MOVE CORRESPONDING CUSMASTF TO CUSBALF-O.                              09/05/83

                    * * * * *   E N D   O F   S O U R C E   * * * * *
```

Exercise 9

The object of this exercise is to generate the two layouts following using SDA. Appendix B and HELP are available to assist you.

1. The record formats will be named CUSFMT1 and CUSFMT2. The Field Reference File will be used to provide field definitions where possible.

   Code for an error message to be displayed if an invalid Customer has been entered. Use indicator 20 and set off in the format.

   CMD 4 (indicator 04) should be used to display a prompt message:

   "PRESS CMD 1 TO RETURN TO THE MENU"

   The display file will be CUSINQDF.

2. Create in YOUR library an RPG program named CUSINQ, the source for which is in QRPGSRC.SF38LIB.

3. Test the formats by running a 'Customer Inquiry' from your menu.

IBM Education

IBM

5250 Information Display System
Keyboard Template Assignment Sheet and
Display Screen Layout Sheet

Format Name CUSFMT1 Description

Job Name Sheet

Originated by Date

Display Mode	13	14	15	16	17	18	19	20	21	22	23	24	Clear
	1 END OF INQUIRIES	2	3	4 PROMPT	5	6	7	8	9	10	11	12	Test Request

*date *time

Display Screen Layout Sheet

COLUMN

ROW	1–10	11–20	21–30	31–40	41–50	51–60	61–70	71–80
01	xx/xx/xx	xx:xx:xx		CUSTOMER INQUIRY				TEAM xx
07				ENTER CUSTOMER NO. xxxxxx (CUST)				
20				Press CMD1 to return to the menu (only display if operator presses CMD4)				

CA01
CA04

IBM Education

SF38 7.9

IBM

5250 Information Display System
Keyboard Template Assignment Sheet and
Display Screen Layout Sheet

Format Name CUSFMT2 Description _____

Job Name _____ Sheet _____ of _____

Originated by _____ Date _____

Display Mode	13	14	15	16	17	18	19	20	21	22	23	24	Clear
	1	2	3	4	5	6	7	8	9	10	11	12	Test Request

Display Screen Layout Sheet

COLUMN

1–10 | 11–20 | 21–30 | 31–40 | 41–50 | 51–60 | 61–70 | 71–80

ROW

01 xx/xx/xx xx:xx:xx CUSTOMER INQUIRY TEAMxx

03 Customer Number xxxxxx (CUST) Name x (NAME) x

04 Address x (CUSAD1) x

05 x (CUSAD2) x

06 x (CUSAD3) x

08 ORDERING INFORMATION

10 Shipment Method xx (SVIA) Payment Code xx (PAYM) Discount Code xx (DISC)

12 Last Order Number xxxx (LORDER)

15 FINANCIAL INFORMATION

17 Credit Limit £xxxx.xx (CRLIM) Account Balance £xxxx.xx (BALANC)

22 Press ENTER to continue.

FLDREFFILE

```
SEQUENCE ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8

  1.00  0001 A* FIELD REFERENCE FILE FOR ORDER ENTRY APPLICATION
  2.00  0002 A**          FLDREFFILE                ORDER ENTRY FIELD REFERENCE FIL
  3.00  0003 A*************************************************************************
  4.00  0004 A          R ORDREFFILE                TEXT('FIELD REFERENCE FILE')
  5.00  0005 A            ACTIVE         1  0      COLHDG('ACTIVE') RANGE(0 2)
  6.00  0006 A                                     TEXT('ITEM STATUS')
  7.00  0007 A            BALANC         6  2                COLHDG('BALANCE')
  8.00  0008 A                                     TEXT('ACCOUNT BALANCE')
  9.00  0009 A            CAMOUN         6  2      EDTCDE(K) COLHDG('CURR' 'VALUE')
 10.00  0010 A                                           TEXT('CURR. ORDER VALUE')
 11.00  0013 A            CRLIM          6  2                COLHDG('CREDIT +
 12.00  0014 A                                     LIMIT') TEXT('CREDIT LIMIT')
 13.00  0085 A            CUSAD1        25        COLHDG('STREET')
 14.00  0086 A                                     TEXT('STREET')
 15.00  0011 A            CUSAD2        25        COLHDG('CITY')
 16.00  0012 A                                     TEXT('CITY')
 17.00  0113 A            CUSAD3        25        COLHDG('ZIP')
 18.00  0114 A                                     TEXT('ZIP CODE')
 19.00  0015 A            CUSORD        10        COLHDG('CUSTOMER' 'ORDERNO')
 20.00  0016 A                                     TEXT('CUSTOMER ORDER NO')
 21.00  0017 A            CUST           6  0      COLHDG('CUSTOMER-NO') CHECK(MF)
 22.00  0018 A                                     TEXT('CUSTOMER NO')
 23.00        A            DESARG         6        TEXT('SEARCH BY DESCRIPTION')
 24.00  0019 A            DESCR         30        COLHDG('DESCRIPTION')
 25.00  0020 A                                     TEXT('ITEM DESCRIPTION')
 26.00  0021 A            DISC           2  0      COLHDG('DISCOUNT') CHECK(MF) +
 27.00  0022 A                                     TEXT('DISCOUNT CODE')
 28.00  0023 A                                     RANGE(01 10)
 29.00  0024 A            DISCT     R             REFFLD(DISC *SRC)
 30.00  0025 A            ITDISC    R             REFFLD(DISC *SRC)  COLHDG('ITEM' 'D+
 31.00  0026 A                                     ISCNT') TEXT('ITEM DISCOUNT')
 32.00  0027 A            ITEM           6  0      COLHDG('ITEMNO') CHECK(MF)
 33.00  0028 A                                     TEXT('ITEM NO')
 34.00  0029 A            ITDSCT    R             REFFLD(ITDISC *SRC)
 35.00  0030 A            LINENO         2  0      COLHDG('LINENO') RANGE(1 99)
 36.00  0031 A                                     TEXT('LINE NO')
 37.00  0032 A            LINET     R             REFFLD(LINENO *SRC)
 38.00  0033 A            LORDER     00005  0      COLHDG('LAST ORDER')
 39.00  0034 A                                          TEXT('NO. OF LAST ORDER')
 40.00  0035 A            MONTH1         6  0                COLHDG('LAST''MO+
 41.00  0036 A                                     NTH') TEXT('QUANTITY PREV.MONTH')
 42.00  0037 A            MONTH2    R             REFFLD(MONTH1 *SRC) COLHDG('TWO' 'M+
 43.00  0038 A                                     ONTHS' 'AGO')
 44.00  0039 A                                     TEXT('SALES MONTH BEFORE LAST')
 45.00  0040 A            MONTHC    R             REFFLD(MONTH1)      COLHDG('CURR' 'M+
 46.00  0041 A                                     ONTH') TEXT('QUANT SOLD CURR MONTH')
 47.00  0042 A            MSGTXT        60        TEXT('MESSAGE TEXT FIELD')
 48.00  0043 A            NAMARG        26        TEXT('SEARCH BY NAME ARGUMENT')
 49.00  0044 A            NAME          25        COLHDG('CUSTOMER' 'NAME')
 50.00  0045 A                                     TEXT('CUSTOMER NAME')
 51.00  0048 A            ONHAND         4  0      COLHDG('ONHAND') EDTCDE(Z)
 52.00  0049 A                                     TEXT('QUANTITY ON HAND')
 53.00  0050 A            ONORD          4  0      COLHDG('ON ORDER')
 54.00  0051 A                                     TEXT('QUANTITY ON ORDER')
 55.00  0052 A            ORDACT         1        TEXT('ORDER ACTIVE FOR RESTART')
 56.00        A            ORDDAY         2  0      COLHDG('ORDER' 'DAY')
 57.00  0053 A            ORDER          5  0      COLHDG('ORDERNO') CHECK(MF)
 58.00  0054 A                                     TEXT('ORDER NO')
 59.00  0055 A            ORDERT    R             REFFLD(ORDER *SRC)
 60.00        A            ORDMTH         2  0      COLHDG('ORDER' 'MONTH')
 61.00  0056 A            ORDTOT    R             REFFLD(CAMOUN *SRC) COLHDG('ORDER T+
 62.00  0057 A                                     OTAL') TEXT('TOTAL ORDER VALUE')
 63.00        A            ORDYR          2  0      COLHDG('ORDER' 'YEAR')
 64.00  0058 A            PAYM           2  0      COLHDG('PAYMENT') CHECK(MF) +
 65.00  0059 A                                     TEXT('PAYMENT METHOD CODE')
 66.00  0060 A                                     RANGE(01 20)
 67.00  0061 A            PAYMT     R             REFFLD(PAYM *SRC)
 68.00  0062 A            POS            2  0      COLHDG('POSITION')
 69.00  0063 A                                     TEXT('ORDER POSITION')
 70.00  0064 A            PRICE          5  2      COLHDG('PRICE') EDTCDE(K)
 71.00  0065 A                                     TEXT('ITEM PRICE')
 72.00  0066 A            PRICET    R             REFFLD(PRICE *SRC)
 73.00  0067 A            QTYSHP         4  0      COLHDG('QUANTITY' 'SHIPPED')
 74.00  0068 A                                     TEXT('QUANTITY SHIPPED')
 75.00  0069 A                                     EDTCDE(Z)
 76.00  0070 A            QUANT          4  0      COLHDG('QUANTITY') EDTCDE(Z)
 77.00  0071 A                                     TEXT('QUANTITY ORDERED')
 78.00  0080 A            SADLN1        25        COLHDG('SHIP TO' 'STREET')
 79.00  0081 A                                     TEXT('SHIP TO ADDRESS STREET')
 80.00  0091 A            SADLN3        25        COLHDG('SHIP TO' 'ZIP')
 81.00  0092 A                                     TEXT('SHIP TO ADDRESS ZIP CODE')
 82.00  0072 A            SADLN2        25        COLHDG('SHIP TO' 'CITY')
 83.00  0073 A                                     TEXT('SHIP TO ADDRESS CITY')
 84.00  0074 A            SINSTR        40        COLHDG('SHIPPING' 'INSTRUCTIONS')
 85.00  0075 A                                     TEXT('SHIPPING INSTRUCTIONS 1')
 86.00  0078 A            SNAME         25        COLHDG('SHIP TO' 'NAME')
 87.00  0079 A                                     TEXT('SHIP TO ADDRESS NAME')
 88.00  0082 A            STATT          1  0      RANGE(1 4)
 89.00  0083 A            STATUS    R             COLHDG('STATUS') REFFLD(STATT *SRC)
 90.00  0084 A                                     TEXT('ORDER STATUS')
 91.00  0087 A            SVIA           2  0      COLHDG('SHIP VIA') CHECK(MF) +
 92.00  0088 A                                     TEXT('SHIP VIA CODE')
 93.00  0089 A                                     RANGE(01 20)
 94.00  0090 A            SVIAT     R             REFFLD(SVIA *SRC)
 95.00  0093 A            TOTAL     R             COLHDG('TOTAL') REFFLD(ORDTOT *SRC)
 96.00  0094 A                                     TEXT('ORDER LINE VALUE')
 97.00  0095 A            UPDATC         3  0      COLHDG('UPDATE COUNT')
 98.00  0096 A                                     TEXT('CUST MASTER UPDATE COUNT')
 99.00  0097 A            UDATHD    R             REFFLD(UPDATC *SRC)
100.00  0098 A                                     TEXT('ORDER HEADER UPDATE COUNT')
101.00  0099 A            UPDATI    R             REFFLD(UPDATC *SRC)
102.00  0100 A                                     TEXT('ITEM MASTER UPDATE COUNT')
103.00  0101 A            UDATIL    R             REFFLD(UPDATC *SRC)
104.00  0102 A                                     TEXT('ORDER LINE UPDATE COUNT')
105.00  0103 A            WHSEQ          5  0      COLHDG('WARE HOUSE ' 'SEQUENCE')
106.00  0104 A                                     TEXT('WAREHOUSE SEQUENCE CODE')
107.00  0105 A            WSOPNA        10        COLHDG('WORKSTN' 'OPERATOR' 'NAME')
108.00  0106 A                                     TEXT('USER NAME')
109.00  0107 A            YRAMT          6  2      COLHDG('YEAR TOTAL')
110.00  0108 A                                           TEXT('VALUE SOLD YTD')
111.00  0109 A            YRQTY          6  0      COLHDG('YEAR' 'QUANTITY')
112.00  0110 A                                     TEXT('QUANTITY SOLD YEAR TO DATE')
113.00  0111 A            YRTOTL         6  2      COLHDG('YEAR TOTAL')
114.00  0112 A                                          TEXT('TOTAL SALES YR TO DATE')
```

DEVICE AND COMMUNICATIONS FILES

1. I/O DEVICES ON SYSTEM/38
   - How I/O devices are defined
   - How the application program uses devices
   - Supplied device files

2. PRINTER DEVICE FILES
   - The Create Print File command
   - Creating output queues
   - Changing/overriding printer files
   - Using a printer file in an RPG program
   - Creating printer record formats
   - Using printer record formats in programs

3. DISKETTE DEVICE FILES
   - The Create Diskette File command

4. CARD DEVICE FILES
   - The Create Card File command

5. TAPE DEVICE FILES
   - The Create tape File command

6. DATA COMMUNICATIONS ON SYSTEM/38

I/O DEVICES ON SYSTEM/38

o How I/O devices are defined

At the time the system is installed, all attached devices need to be defined, through device descriptions. This gives the device name and type, and its address on the system.

```
CRTDEVD QSYSPRT DEVADR(000018) DEVTYPE(3262) MODEL(B1)
```

from CL manual

When a device description is created, a device file is automatically created with the same name.

The system comes complete with device descriptions for the integral devices (QDKT and QCONSOLE) and may have DEVD's for system printer(s), MFCU and Magnetic Tape devices (as ordered).

You will have to create device descriptions for workstations and workstation printers after the installation of the system software. You will also have to create descriptions of communications lines and control units.

```
CRTCUD  WSC1 TYPE(*WSC) MODEL(*NONE) CTLADR(0030) DEV(WS24)
```

always 0 for WS

```
CRTDEVD WS24 DEVADR(000000) DEVTYPE(5251) MODEL(11) +
          CTLU(WSC1) WSCADR(080004) WSCKBD(TUKB)
```

o How the application program uses devices

On System/38 all files must be defined to the system before a program can use them. The program refers to a device not by device name but through a device file.

o Supplied device files

A number of device files are supplied with the system

QPRINT
QPRINT2
QPRINTS
QDKT
QDKTSRC
QCARD96
QCRDSRC
QPUNCH
QTAPE
QTAPSRC

The RPG programmer has to use a defined file name in his program.

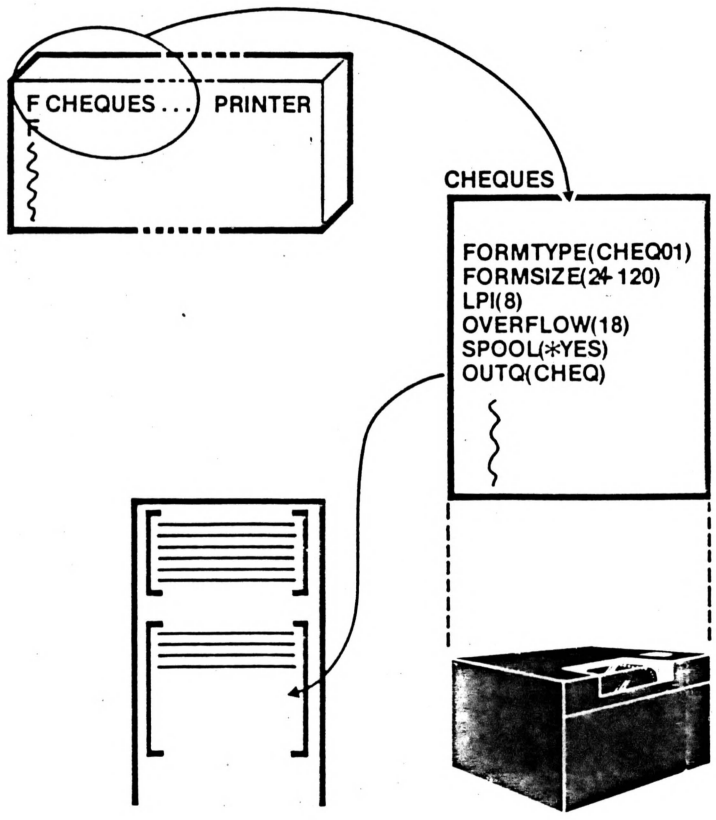
PRINTER DEVICE FILES

o The Create Printer File command

The Create Printer File (CRTPRTF) command creates a printer file for use by a program. You can further describe the print file by way of DDS to enable the system to look after spacing, underlining, pagination etc. Either way, if you have a printer file named in your program, that printer file must exist on the machine. If you wish to describe the file in your program (the traditional method) you must still create a printer file on the machine that corresponds to the file description entry (or FD Section) in your program. For example:

```
CRTPRTF CHEQUES FORMTYPE(CHEQ01) FORMSIZE(24 120) LPI(8) +
        OVERFLOW(18) SPOOL(*YES) OUTQ(CHEQ)
```

@PJOBLOG
print(er) file for JOBLOG
contains format for JOBLOG
CPF manual
app. B



o Creating output queues

You can create your own output queues with the Create Output Queue commands:

CRTOUTQ PGMRQ DSPDTA(*YES)

o Changing printer files

Like data base files, printer files can be changed permanently (CHGPRTF command) before the execution of a program. As an illustration, a permanent change made to the IBM-supplied printer file QSYSPRT (a file heavily used by many CPF functions) in the Education installation has been:

CHGPRTF QSYSPRT HOLD(*YES) SCHEDULE(*FILEEND)

This permanent modification means that all output formatted via QSYSPRT will be held on the output queue until explicitly released, and that the output will be made available for printing when the program closes the file. (The default is that the output is unavailable until the job that produced it - not the program - goes to EOJ. For interactive jobs this is SIGNOFF and is operationally very awkward).

As has already been seen the CRTPRTF command can also nominate an output queue (default QPRINT) to which it will deliver its output and the stationery type to be used for the output. One way to manage differing stationary types is to let the program describe the print file but deliver its output to a queue devoted entirely to handling that particular stationery type.

The system operator then needs only to start a print writer to a given queue after changing the stationery. All the output on that queue would require that stationery, rather than different files on the same queue needing different stationery.
For instance, at the Education installation the IBM-supplied print file, QPSRVDMP, which handles dumps, has been changed:

```
CHGPRTF  QPSRVDMP  OUTQ(QPRINTS) SCHEDULE(*FILEEND)
```

Alternatively, STRPRTWTR can be addressed to a forms-type.

- Using a printer file in an RPG program

Up to 8 printer device files can be used in an RPG program. The files can be described within the program (Output Specifications are used) or externally described (the device file contains one or more printer record formats).

- Creating printer record formats

Just as with workstation displays, printer record formats can be described using DDS forms. These DDS statements need to be stored in a source file (QDDSSRC) member, then the printer file can be created:

```
CRTPRTF     EMPLISTR     SRCFILE(QDDSSRC)
```

- Using printer record formats in programs

RPG and COBOL programs can use externally described printer files, but CL programs cannot.

DDS for p. 8.12

```
5714SS1 R03 M00 811204                  DATA DESCRIPTION                ORDREPTR.IC38LIB        18/05/82 11:53:1

File name -         ORDREPTR.IC38LIB    Type of file -      PRINTER
Source file -       QDDSSRC.IC38LIB     Member -            ORDREPTR    03/09/81  11:26:39
Type of data -      *DATA
Options -           *SRC    *LIST
Authority -         *NORMAL
Text -              Order value printer file for IC38
Compiler -          IBM System/38 Data Description Processor

                                    DATA DESCRIPTION SOURCE

SEQNBR  *... ... 1 ... ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 ... ... 8  DATE

   100      A* PRINTER RECORD LAYOUT FOR CUSTOMER ORDERS REPORT BY DATE AND TI
   200      A*
   300      A                                      REF(FLDREFFILE.OELIB)
   400      A        R ORDRPT1F                    TEXT('TITLE AND HEADINGS')          03/09/81
   500      A                                      SKIPB(1)
   600      A                                      SPACEA(3)
   700      A                                     20DFT('CUSTOMER ORDER LIST BY +
   800      A                                       DATE AND VALUE') UNDERLINE
   900      A                                    +10DATE EDTCDE(Y)
  1000      A                                    + 5TIME
  1100      A                                       SPACEA(2)
  1200      A                                     15DFT('ORDER DATE')
  1300      A                                    + 5DFT('CUSTOMER NUMBER')
  1400      A                                    + 5DFT('ORDER NUMBER')
  1500      A                                    + 5DFT('ORDER VALUE')
  1600      A        R ORDRPT2F                    TEXT('DETAIL RECORD FORMAT')        03/09/81
  1700      A          ORDATE          6  0      16EDTCDE(Y)
  1800      A          CUST       R             + 9
  1900      A          ORDER      R             +12
  2000      A          ORDTOT     R             + 7EDTCDE(J £) SPACEA(1)
  2100      A        R ORDRPT3F                    TEXT('TOTALS RECORD FORMAT')        03/09/81
  2200      A                                      SKIPB(1)
  2300      A                                     20DFT('CUSTOMER ORDER TOTALS')
  2400      A                                       UNDERLINE
  2500      A                                    + 5DATE EDTCDE(Y)
  2600      A                                    + 5TIME
  2700      A                                       SPACEA(3)
  2800      A                                     15DFT('TOTAL ORDER VALUE')
  2900      A                                       UNDERLINE
  3000      A                                    + 5DFT('TOTAL NUMBER OF ORDERS')
  3100      A                                       UNDERLINE
  3200      A                                       SPACEA(2)
  3300      A          TOTVAL          9  2      15EDTCDE(J £)
  3400      A          TOTNBR          3  0      37EDTCDE(Z)

                      ***** END OF SOURCE *****
```

```
5714RG1 RPG  R03M00  811204   ORDLISTG.IC38LIB     18/05/82  15:57:59  PAGE     2

SEQUENCE       1         2         3         4         5         6         7       IND    DO    LAST
NUMBER     6789012345678901234567890123456789012345678901234567890123456789012345 USE    NUM   UPDATE


NAME OF PROGRAM WILL BE ORDLISTG IN LIBRARY IC38LIB

           H


      100  FORDVAL1LIF  E            K        DISK
            RECORD FORMAT(S):  FILE ORDVAL1L LIB IC38LIB
                     EXTERNAL FORMAT ORDVAL1F RPG NAME ORDVAL1F

      200  FORDREPTRO   E                     PRINTER
      300  F                                                   KINFDS PRINT
            RECORD FORMAT(S):  FILE ORDREPTR LIB IC38LIB
                     EXTERNAL FORMAT ORDRPT1F RPG NAME ORDRPT1F
                     EXTERNAL FORMAT ORDRPT2F RPG NAME ORDRPT2F
                     EXTERNAL FORMAT ORDRPT3F RPG NAME ORDRPT3F


  1000000   INPUT  FIELDS FOR RECORD ORDVAL1F FILE ORDVAL1L FORMAT ORDVAL1F       ORDERS BY DATE/VALUE
  1000001                                     P   1   30ORDER                      ORDER NO
  1000002                                     P   4   70CUST                       CUSTOMER NO
  1000003                                     P   8  112ORDTOT                     TOTAL ORDER VALUE
  1000004                                     P  12  130ORDDAY                     ORDER DAY
  1000005                                     P  14  150ORDMTH                     ORDER MONTH
  1000006                                     P  16  170ORDYR                      ORDER YEAR
  1000007                                        18  230ORDATE
  1000008                                     P  24  240STATUS                     ORDER STATUS
      400  IPRINT       DS
      500  I                                  B 367 3680CURLIN


      600  C*   First Page output is produced by writing the printer header
      700  C                   WRITEORDRPT1F
      800  C*   DO loop reads records from file
      900  C                   DO   *HIVAL                                          B001
     1000  C                   READ ORDVAL1F             LR                  3     001    18/05/82
     1100  C    LR             GOTO ENDIT                                          001
     1200  C                   WRITEORDRPT2F                                       001    18/05/82
     1300  C                   ADD  ORDTOT    TOTVAL                               001
     1400  C                   ADD  1         TOTNBR                               001
     1500  C*   Check current line value for overflow
     1600  C         CURLIN    COMP 60                   99                  3     001
     1700  C*   Output printer header if overflow
     1800  C    99             WRITEORDRPT1F                                       001
     1900  C                   END                                                 E001
     2000  C         ENDIT     TAG
     2100  C                   WRITEORDRPT3F
  2000000   OUTPUT  FIELDS FOR RECORD ORDRPT1F FILE ORDREPTR FORMAT ORDRPT1F     TITLE AND HEADINGS
  3000000   OUTPUT  FIELDS FOR RECORD ORDRPT2F FILE ORDREPTR FORMAT ORDRPT2F     DETAIL RECORD FORMAT
  3000001                        ORDATE     6  ZONE  6,0
  3000002                        CUST      12  ZONE  6,0                         CUSTOMER NO
  3000003                        ORDER     17  ZONE  5,0                         ORDER NO
  3000004                        ORDTOT    23  ZONE  6,2                         TOTAL ORDER VALUE



5714RG1 RPG  R03M00  811204   ORDLISTG.IC38LIB     18/05/82  15:57:59  PAGE     3

SEQUENCE       1         2         3         4         5         6         7       IND    DO    LAST
NUMBER     6789012345678901234567890123456789012345678901234567890123456789012345 USE    NUM   UPDATE

  4000000   OUTPUT  FIELDS FOR RECORD ORDRPT3F FILE ORDREPTR FORMAT ORDRPT3F     TOTALS RECORD FORMAT
  4000001                        TOTVAL     9  ZONE  9,2
  4000002                        TOTNBR    12  ZONE  3,0


* * * * * E N D   O F   S O U R C E * * * * *
```

automatically generated because externally defined

```
5714CB1 R04 M01 820813                COBOL SOURCE LISTING                                  10/05/83 13:51:0
 STMT SEQNBR -A 1 B.. ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 .IDENTFCN S  COPYNAME   CHG/DATE

    1  000100 IDENTIFICATION DIVISION.
       000200*
    2  000300 PROGRAM-ID.    ORDLISTC.                                                                 10/05/83
    3  000400 AUTHOR.        BRUCE WASSELL.                                                            10/05/83
    4  000500 INSTALLATION.  BASINGSTOKE EDUCATION CENTRE.                                             10/05/83
    5  000600 DATE-COMPILED.  10/05/83 13:51:03    .                                                   10/05/83
       000700*
    6  000800 ENVIRONMENT DIVISION.
       000900*
    7  001000 CONFIGURATION SECTION.                                                                   10/05/83
       001100*                                                                                         10/05/83
    8  001200 SOURCE-COMPUTER. IBM-S38.                                                                10/05/83
    9  001300 OBJECT-COMPUTER. IBM-S38.                                                                10/05/83
   10  001400 SPECIAL-NAMES.   I-O-FEEDBACK IS PRINTER-FEEDBACK.                                       10/05/83
       001500*
   11  001600 INPUT-OUTPUT SECTION.
       001700*
   12  001800 FILE-CONTROL.
   13  001900        SELECT ORDER-FILE
   14  002000            ASSIGN TO DATABASE-ORDVAL1L
   15  002100            ORGANIZATION IS INDEXED
   16  002200            ACCESS MODE IS SEQUENTIAL
   17  002300            RECORD KEY IS ORDYR WITH DUPLICATES.
   18  002400        SELECT PRINTER-FILE
   19  002500            ASSIGN TO FORMATFILE-ORDREPTR.
       002600*
   20  002700 DATA DIVISION.
       002800*
   21  002900 FILE SECTION.
       003000*
   22  003100 FD   ORDER-FILE
   23  003200      LABEL RECORDS ARE STANDARD.
       003300*
   24  003400 01   ORDER-RECORD.
   25  003500      COPY DDS-ORDVAL1F OF ORDVAL1L.
   27 +000001                                                                               ORDVAL1F
      +000002*     I-O FORMAT:ORDVAL1F   FROM FILE ORDVAL1L   OF LIBRARY BRUCELIB           ORDVAL1F
      +000003*                           ORDERS BY DATE/VALUE                               ORDVAL1F
      +000004*THE KEY DEFINITIONS FOR RECORD FORMAT  ORDVAL1F                               ORDVAL1F
      +000005*  NUMBER               NAME              RETRIEVAL     TYPE    ALTSEQ          ORDVAL1F
      +000006*    0001            ORDYR                 ASCENDING      N       NO            ORDVAL1F
      +000007*    0002            ORDMTH                ASCENDING      N       NO            ORDVAL1F
      +000008*    0003            ORDDAY                ASCENDING      N       NO            ORDVAL1F
      +000009*    0004            ORDTOT                ASCENDING      N       NO            ORDVAL1F
   28 +000010       05   ORDVAL1F.                                                          ORDVAL1F
   29 +000011            06 ORDER            PIC S9(5)         COMP-3.                      ORDVAL1F
      +000012*                          ORDER NO                                            ORDVAL1F
   30 +000013            06 CUST             PIC S9(6)         COMP-3.                      ORDVAL1F
      +000014*                          CUSTOMER NO                                         ORDVAL1F
   31 +000015            06 ORDTOT           PIC S9(4)V9(2)    COMP-3.                      ORDVAL1F
      +000016*                          TOTAL ORDER VALUE                                   ORDVAL1F
   32 +000017            06 ORDDAY           PIC S9(2)         COMP-3.                      ORDVAL1F
      +000018*                          ORDER DAY                                           ORDVAL1F
   33 +000019            06 ORDMTH           PIC S9(2)         COMP-3.                      ORDVAL1F
      +000020*                          ORDER MONTH                                         ORDVAL1F
```

```
5716CB4 V04 M01 820813              COBOL SOURCE LISTING            ORDLISTC              10/05/83 13:51:03

STMT SEQNBR -A 1 B.. ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 .IDENTFCN S  COPYNAME   CHG/DATE

  34 +000021           06 ORDYR            PIC S9(2)        COMP-3.                    ORDVAL1F
     +000022*                            ORDER YEAR                                    ORDVAL1F
  35 +000023           06 ORDATE           PIC S9(6).                                  ORDVAL1F
  36 +000024           06 STATUS-DDS       PIC S9(1)        COMP-3.                    ORDVAL1F
     +000025*                            ORDER STATUS                                  ORDVAL1F
      003600*
  37  003700 FD  PRINTER-FILE
  38  003800     LABEL RECORDS ARE STANDARD.
      003900*
  39  004000 01  PRINTER-RECORD          PIC X(132).                                   10/05/83
      004100*
  40  004200 WORKING-STORAGE SECTION.
      004300*
  41  004400 01  HEADING-LINES.
  42  004500     COPY DDS-ORDRPT1F-O OF ORDREPTR.
  43 +000001                                                                          ORDRPT1F
     +000002*    I-O FORMAT:ORDRPT1F   FROM FILE ORDREPTR   OF LIBRARY BRUCELIB       ORDRPT1F
     +000003*                          TITLE AND HEADINGS                            ORDRPT1F
     +000004*      05  ORDRPT1F.                                                     ORDRPT1F
  45 +000005             07 FILLER      PIC X(1).                                    ORDRPT1F
      004600*
  46  004700 01  DETAIL-LINE.
  47  004800     COPY DDS-ORDRPT2F-O OF ORDREPTR.
  49 +000001                                                                          ORDRPT2F
     +000002* OUTPUT FORMAT:ORDRPT2F   FROM FILE ORDREPTR   OF LIBRARY BRUCELIB       ORDRPT2F
     +000003*                          DETAIL RECORD FORMAT                          ORDRPT2F
  50 +000004       05  ORDRPT2F-O.                                                   ORDRPT2F
  51 +000005           06 ORDATE           PIC S9(6).                                ORDRPT2F
  52 +000006           06 CUST             PIC S9(6).                                ORDRPT2F
     +000007*                            CUSTOMER NO                                 ORDRPT2F
- 53 +000008           06 ORDER            PIC S9(5).                                ORDRPT2F
     +000009*                            ORDER NO                                    ORDRPT2F
  54 +000010           06 ORDTOT           PIC S9(4)V9(2).                           ORDRPT2F
     +000011*                            TOTAL ORDER VALUE                           ORDRPT2F
      004900*
  55  005000 01  TOTAL-LINES.
  56  005100     COPY DDS-ORDRPT3F-O OF ORDREPTR.
  58 +000001                                                                          ORDRPT3F
     +000002* OUTPUT FORMAT:ORDRPT3F   FROM FILE ORDREPTR   OF LIBRARY BRUCELIB       ORDRPT3F
     +000003*                          TOTALS RECORD FORMAT                          ORDRPT3F
  59 +000004       05  ORDRPT3F-O.                                                   ORDRPT3F
  60 +000005           06 TOTVAL           PIC S9(7)V9(2).                           ORDRPT3F
  61 +000006           06 TOTNBR           PIC S9(3).                                ORDRPT3F
      005200*
  62  005300 01  FEEDBACK-DATA.
      005400*
  63  005500     05  FILLER                PIC X(144).
  64  005600     05  LINE-NUMBER           PIC S99  COMP-4.
      005700*
  65  005800 01  OVERFLOW-LINE             PIC S99  COMP-4  VALUE 60.
      005900*
  66  006000 01  END-OF-FILE-FLAG          PIC 1            VALUE B"0".
  67  006100     88  END-OF-ORDERS                           VALUE B"1".
  68  006200 PROCEDURE DIVISION.
      006300*
```

```
5714CB1 R04 M01 820912              COBOL SOURCE LISTING              ORDLISTC              10/05/83 13:51:07

STMT SEQNBR -A 1 B.. ... 2 ... ... 3 ... ... 4 ... ... 5 ... ... 6 ... ... 7 .IDENTFCN S COPYNAME   CHG/DATE

       006400 000-PRINT-ORDER-LIST.
       006500*
  69   006600     OPEN INPUT  ORDER-FILE
       006700          OUTPUT PRINTER-FILE.
  70   006800     MOVE ZERO TO TOTVAL OF TOTAL-LINES TOTNBR OF TOTAL-LINES.                            10/05/83
  71   006900     PERFORM 100-PRINT-ORDER-HEADER.
  72   007000     PERFORM 200-PROCESS-DETAIL-RECORD
       007100         UNTIL END-OF-ORDERS.
  73   007200     PERFORM 300-PRINT-TOTAL-LINES.
  74   007300     CLOSE ORDER-FILE
       007400           PRINTER-FILE.
  75   007500     STOP RUN.
       007600*
       007700 100-PRINT-ORDER-HEADER.
       007800*
  76   007900     WRITE PRINTER-RECORD FROM HEADING-LINES
       008000         FORMAT IS "ORDRPT1F".
       008100*
       008200 200-PROCESS-DETAIL-RECORD.
       008300*
  77   008400     PERFORM 210-READ-ORDER-FILE.
  78   008500     IF NOT END-OF-ORDERS
  79   008600         MOVE CORRESPONDING ORDVAL1F TO ORDRPT2F-O
  80   008700         ADD ORDTOT OF ORDVAL1F TO TOTVAL OF ORDRPT3F-O
  81   008800         ADD 1 TO TOTNBR OF ORDRPT3F-O
  82   008900         PERFORM 220-PRINT-ORDER-DETAIL
  83   009000         ACCEPT FEEDBACK-DATA FROM PRINTER-FEEDBACK
  84   009100         IF LINE-NUMBER IS EQUAL TO OVERFLOW-LINE
  85   009200             PERFORM 100-PRINT-ORDER-HEADER
       009300         ELSE NEXT SENTENCE
       009400     ELSE NEXT SENTENCE.
       009500*
       009600 300-PRINT-TOTAL-LINES.
       009700*
  86   009800     WRITE PRINTER-RECORD FROM TOTAL-LINES
       009900         FORMAT IS "ORDRPT3F".
       010000*
       010100 210-READ-ORDER-FILE.                                                                      10/05/83
       010200*
  87   010300     READ ORDER-FILE
       010400         FORMAT IS "ORDVAL1F"
  88   010500         AT END SET END-OF-ORDERS TO TRUE.
       010600*
       010700 220-PRINT-ORDER-DETAIL.
       010800*
  89   010900     WRITE PRINTER-RECORD FROM DETAIL-LINE
       011000         FORMAT IS "ORDRPT2F".

                  * * * * *   E N D   O F   S O U R C E   * * * * *
```

CUSTOMER_ORDER_LIST_BY_DATE_AND_VALUE 18/05/82 17:51:26

ORDER DATE	CUSTOMER NUMBER	ORDER NUMBER	ORDER VALUE
12/03/82	332191	11629	£9.00
12/03/82	1	11626	£17.85
12/03/82	635601	11643	£19.00
12/03/82	332391	11630	£26.25
12/03/82	1	11639	£45.00
12/03/82	380011	11640	£55.89
15/03/82	341011	11644	£4.50
23/03/82	635091	11646	£11.00
24/03/82	635601	11647	£2.50
25/03/82	635601	11648	£10.80
6/04/82	635081	11650	£105.00
6/04/82	635081	11651	£209.23
13/04/82	34011	11652	£223.00
19/04/82	766677	11659	£13.00
19/04/82	332391	11662	£50.33
19/04/82	138242	11653	£2,676.00
19/04/82	766677	11660	£7,500.00
20/04/82	138242	11665	£3,943.50
21/04/82	332191	11668	£1.25
21/04/82	332191	11675	£2.15
21/04/82	1	11669	£25.50
21/04/82	203542	11671	£86.25
21/04/82	111111	11674	£148.68
21/04/82	1	11676	£412.00
21/04/82	651011	11670	£6,667.50
22/04/82	298011	11680	£22.65
22/04/82	1	11681	£25.00
3/05/82	380011	11690	£14.00
3/05/82	332391	11689	£19.00
4/05/82	766677	11687	£14.16
6/05/82	332191	11692	£4.50
6/05/82	3081	11693	£152.50
7/05/82	766677	11695	£4.00
7/05/82	766677	11694	£11.60
7/05/82	332261	11698	£12.00
7/05/82	5021	11697	£26.00
7/05/82	766677	11696	£52.00

CUSTOMER_ORDER_TOTALS 18/05/82 17:51:26

TOTAL_ORDER_VALUE	TOTAL_NUMBER_OF_ORDERS
£22,631.59	38

DISKETTE DEVICE FILES

The Create Diskette File command is used to create another diskette file (QDKT is the default)

```
CRTDKTF     ORDERS DEV(QDKT) VOL(ORDER1 ORDER2)
            LABEL(MONDAYS) LOC(*S12)  EXPDATE(08/07/79)
            SPOOL(*NO)
```

If the diskette magazine drive is to be used in an RPG program, the device file name has to be given with an RPG device type of SEQ.

CARD DEVICE FILES

The Create Card File (CRTCRDF) command is used to create card-type device files.

TAPE DEVICE FILES

The Create Tape File (CRTTAPF) command is used to create tape-type device files.

Tape output can not be spooled.

DATA COMMUNICATIONS ON SYSTEM/38

Communicating with Remote Workstations

The device file used in this situation is the normal Display File.

From the Data Communications viewpoint, a Line Description and a Control Unit Description must have been created - these in addition to and created before the Device Description for the workstation(s).

```
CRTLIND PORT1 LINNBR(20) TYPE(*SDLCP) CNN(*PP) RATE(4800)

CRTCUD  BSTOKE TYPE(5251) MODEL(0012) CTLADR(0120) LINE(PORT1)
```

Communicating with another System or device

The device file used in this situation may depend, in part, on the line 'protocol' in use.
The Control Unit Description will specify either BSC or SDLC.
In the case of BSC, the device file may be a BSC File, which will specify one format for the 'Sending' operation and a second format for the 'Receiving' operation.
In the case of SDLC, the device file may be a Communications File.

Alternatively, a MIXED File may be created.
This is a device file which can allow access to one or more different types of Display, BSC or Communications device (up to 256 devices) from the same file in an application program (RPG III or Cobol only).
Input can be invited from one or more of the devices - the program can be doing other work while the invited devices are sending data, and can wait for input from any one of the invited devices.
DDS defines record formats and fields used with the program devices.
The program uses a Program Device Name (which can differ from the Device Name) to access the devices.

After creation of a Mixed File, other CL commands (ADDDSPDEVE, ADDBSCDEVE, ADDCMNDEVE) add entries which specify the devices to be used with the file. SHARE(*YES) can be specified for the file.
Each required program device in the mixed file must be 'acquired' (implicitly for one device, or explicitly) when the file is opened or during execution of the program.

Before a program can be executed which is to communicate with another system or device, the communications link must be established. Apart from making sure that all the appropriate devices are powered on and varied on, the other system may require some identification of the System/38, before it will allow a program to engage in conversation with it. This identification can be supplied by the BSC, Communications or Mixed File.
If this is so, the System/38 program can be called, like any other program.

Communicating with another system using a BSC file will involve ensuring that one program starts with a read, while the other starts with a write. The program which issues a read will wait until data actually arrives.

An SDLC session will begin with a request to the other system, which can be specified when the file is created.
As soon as the file is opened by the program which uses it, the request will be transmitted to the other system.
APPC (Advanced Program to Program Communication) allows System/38 or the other system to initiate programs without operator intervention.

FILE OVERIDES

Overriding data base files
Overriding display files
Overriding device files

Exercise 10

Overriding data base files
(Reference: Chapter 9 CPF Programmers Guide)

Some attributes of a data base file can be modified temporarily before a program is called. Overrides are, however, only in effect for a routing step, but can be deleted with the DLTOVR.

from R7, mbr can be *all (i.e all members processed sequentially)

```
OVRDBF  EMPLOYEES MBR(WEEKLY)
CALL    PAYPGM
```

selects the particular group of employee records in one member of the file for processing.

Where there are several overrides, they are applied from the innermost out

= ignore previous overrides

```
1.....OVRDBF EMPLOYEES SECURE(*YES)
      CALL PAYROL
'PAYROL' 2.....OVRDBF EMPLOYEES TOFILE(MANAGERS)
         CALL PAYPROC
```

The DSPOVR command shows the effective override (a merge of the overrides at all invocation levels) and contributing overrides.

Other parameters of the OVRDBF command:

R7, not if mbr(*all)

POSITION	positions the file to the start, end, or a particular record - by RRN or key
WAITFILE	time to wait for the file
WAITRCD	time to wait for any record
SEQONLY	allows control of input and output blocking for sequentially processed files. (unless NBRRCDS is specified)

NBRRCDS can shorten processing time when the records are physically in the required sequence (this can be done with the RGZPFM command). Specifies how many records are to be fetched into main storage each time records are requested by the program.

LVLCHK(*NO) do not check the record format identifier against that stored in the program.

EXPCHK(*NO) do not test to see if the file expiration date is past

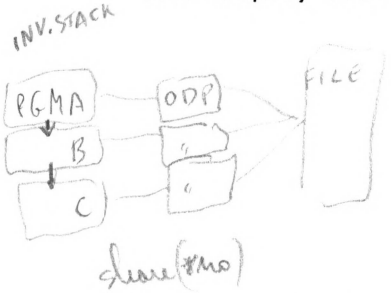
INHWRT(*YES) don't write or update records in the file - this might be useful in restarting a failed or aborted batch job stream

SHARE(*YES) share Open Data Path.

EOFDLY wait a specified time before trying to read additional records once EOF has been reached.

Display Files

Most parameters on the CRTDSPF can be overridden at execution time with the OVRDSPF command.
The display file can be permanently changed with the CHGDSPF command.



1st prog must open file for worst case. (1st prog in stack must have cosmetic programming for that.) also each program will have to reset pointer in ODP (eg SETLL)

Device Files

All programs communicate with the device through its device file. The parameters for these device files (including the IBM-supplied device files) can be overriden at execution time with the appropriate OVRxxxF command, or permanently changed with the appropriate CHGxxx command.

For example, characteristics of the print file can be altered permanently (CHGPRTF), or overridden at program execution time by the OVRPRTF command which then behaves rather like a // PRINTER statement on S/34.

Consider a program that specifies a printer file named REPORT, and that the run-time characteristics of that file need to be modified. The commands may look as follows:

```
OVRPRTF REPORT TOFILE(QSYSPRT) OUTQ(INVOICES) COPIES(3) +
          SCHEDULE(*FILEEND)
CALL OEINVCE
DLTOVR REPORT  (Not strictly necessary but good practice)
```

The parameter values could be supplied at execution time, so they may be coded as variables: eg by a user in a menu.

```
OVRPRTF REPORT TOFILE(QSYSPRT) OUTQ(&QNAME) COPIES(&COPIES) +
          SCHEDULE(*FILEEND)
etc.
```

Naturally the effect of this override does not affect any other user of the system - it is limited to the job in which the command is issued.

Exercise 10

The object of this exercise is to modify your menu and menu program to allow the operator to change the number of copies, and the sequence of the 'Dispatch Notes'.

1. Modify your menu format to include two new fields:

   COPIES (2 0)

   SEQNC (7A)

   The sequence field should only allow a value of ORDERL1 (orders by order no.) or ORDERL2 (orders by customer no.).

2. Modify your CL program to include file overrides for the logical (ORDERL1) and printer (QSYSPRT) file. Have the PRINT program run as an interactive job.

COPY FILE

File Copying

Exercise 11

The QUERY Utility

Exercise 12

File Copying

Reference: Chapter 10 of CPF Programmers Guide.

A powerful feature of S/38 file management is the Copy File function. As the diagram on the next page illustrates, it is possible to copy data from any input file to almost any output file.

You can copy records from and to both Data Base and Device files using the following commands:

Copy File	CPYF
Copy Source File	CPYSRCF
Copy From Diskette	CPYFRMDKT
Copy To Diskette	CPYTODKT
Copy From Tape	CPYFRMTAP
Copy To Tape	CPYTOTAP

create dupl. object
CRTDUPOBJ

The following illustrate the permutations of file copying:

From-File	To-File: Physical	Printer	Diskette	Card	Tape	*LIST[4]
Physical	X[1]	X	X	X	X	X
Logical	X[1]	X	X	X	X	X
Diskette	X	X	X[3]	X	X	X
Card	X	X	X	X	X	X
Tape	X	X	X	X	X	X
Spooled Inline[2]	X	X	X	X	X	X

[1]If the to-file does not exist before the copy operation and the from-file is a physical or logical file, the copy operation will create a physical file as the to-file if you specified CRTFILE(*YES) on the CPYF command.
[2]A spooled inline file (which is treated like a device file) is included as part of a batch job when the job is read by a reader program.
[3]If the from-file and the to-file are both diskette files, the to-file must be spooled and a diskette spool writer must not be active before the copy operation.
[4]If TOFILE(*LIST) is specified, the from-file records are copied to the IBM-supplied printer device file QSYSPRT and formatted according to the PRTFMT parameter.

Copy Function	Parameter	Data Base Files: Physical From	Physical To	Logical From	Logical To	Device Files: Spooled Input From	Spooled Input To	Diskette From	Diskette To	Tape From	Tape To	Card From	Card To	Printer From	Printer To
Select files	FROMFILE	X		X		X		X[5]		X		X			
	TOFILE		X						X[5]		X		X		X
Select members	FROMMBR	X		X				X		X					
	TOMBR		X						X		X				
Add to or replace existing records	MBROPT		X												
Create the to-file	CRTFILE[1]	X	X	X											
Print copied and/or excluded records	PRINT[2]	X	X	X		X		X	X	X	X	X	X		X
Select by record format	RCDFMT			X											
Select by relative record number	FROMRCD	X		X[3]		X		X		X		X			
	TORCD	X		X[3]		X		X		X		X			
Select by key field value	FROMKEY	X		X											
	TOKEY	X		X											
Specify number of records to copy	NBRRCDS	X		X		X		X		X		X			
Select by character content	INCCHAR	X		X		X		X		X		X			
Select by field value	INCREL	X		X											
Process different data base record formats	FMTOPT	X	X	X											
Update sequence number and/or date	SRCOPT	X	X	X											
Specify start value and increment	SRCSEQ	X	X	X											
Print character and/or hex format	PRTFMT[2]	X	X	X		X		X	X	X	X	X	X		X
Maximum recoverable errors allowed	ERRLVL	X	X	X						X					
Disregard or include deleted records	COMPRESS[4]	X	X												

[1]If the to-file does not exist before the copy operation and the from-file is a physical or logical file, the copy operation will create a physical file as the to-file if you specified CRTFILE(*YES) on the CPYF command.
[2]You can specify a program-described printer file so that the copy will produce a listing with no special formatting or page headings, or you can specify TOFILE(*LIST) to produce a formatted listing. You can specify PRINT(*COPIED) to produce a formatted listing of the copied records, and you can specify PRINT(*EXCLD) to produce a formatted listing of the records excluded by the INCCHAR or INCREL parameter. When you request a listing by specifying TOFILE(*LIST) or by the PRINT parameter, the PRTFMT parameter specifies whether the data is printed in character or in both character and hexadecimal form.
[3]You can specify the FROMRCD and TORCD parameter values for a logical file if it has an arrival sequence access path.
[4]You cannot specify COMPRESS(*NO) if:
- The to-file member or a logical file member based on the to-file member has a keyed access path with any of the following attributes:
  - Unique keys (UNIQUE keyword specified in the DDS)
  - Floating-point key field and not MAINT(*REBLD)
  - Select/omit specifications in the DDS and not MAINT(*REBLD)
- Field-level mapping or source/data conversion is required (FMTOPT parameter)

Note: To copy deleted records, the from-file must be processed in arrival sequence.
[5]If the from-file and to-file are diskette files, you must specify that the to-file be spooled (SPOOL(*YES)) on a CRTDKTF, CHGDKTF, or OVRDKTF command.

Record Selection

You can select records to be copied on given selection criteria. These criteria are (parameter keywords in parenthesis):

° Record Keys (FROMKEY and TOKEY)

Limits are set by these two parameters. You can specify the number of key fields (if there are composite keys) to be used in searching the record keys.

° Relative record numbers (FROMRCD and TORCD)

These keywords can be specified for card, diskette, tape, physical (even a keyed sequence access path), and logical files that are in arrival sequence.

° Compression (COMPRESS)

COMPRESS(*YES) is the default. Deleted records are not copied. The relative record numbers will be changed to reflect this on an arrival sequence physical file.

° Record character selection (INCCHAR)

You can use this keyword to test for the presence or absence of a character in a certain position of a record, or in a field in a record to determine whether or not the record is to be copied. eg:

```
CPYF  FILEIN  FILEOUT  INCCHAR(*RCD 1 *NE D)
                    or
CPYF  FILEIN  FILEOUT  INCCHAR(FLDA 1 *NE D)
```

° Field content (INCREL)

You can only test for the value of a field when copying a data base file. Up to 50 AND and OR relationships can be specified, but be careful with your logic; this example says 'if FLDA is greater than 5 and FLDB is less than 6, select the record, OR if FLDB is equal to 9 (FLDA isn't tested), select the record':-

INCREL((*IF FLDA *GT 5) (*AND FLDB *LT 6) (*OR FLDB *EQ 9))

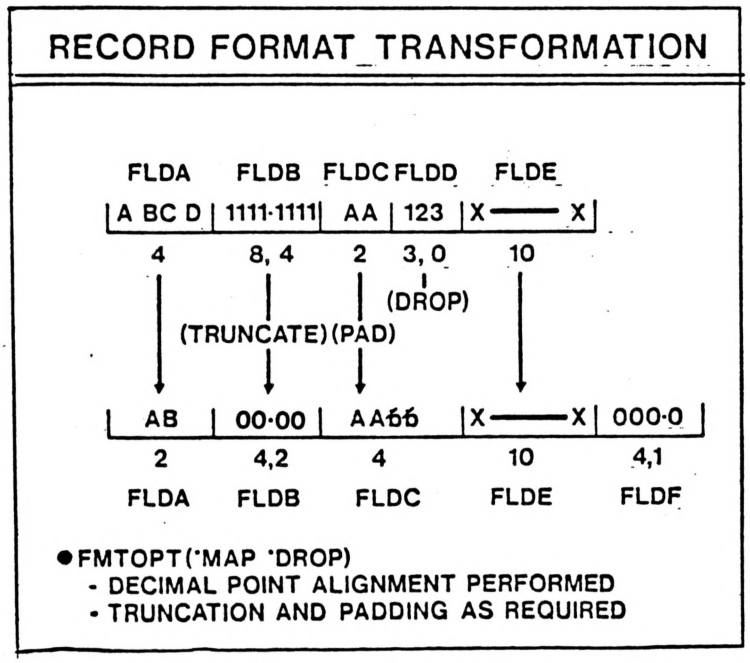
This is because the *OR operator is used to group the *ANDs and not the brackets. The constants 5, 6 and 9 could equally well have been field names.

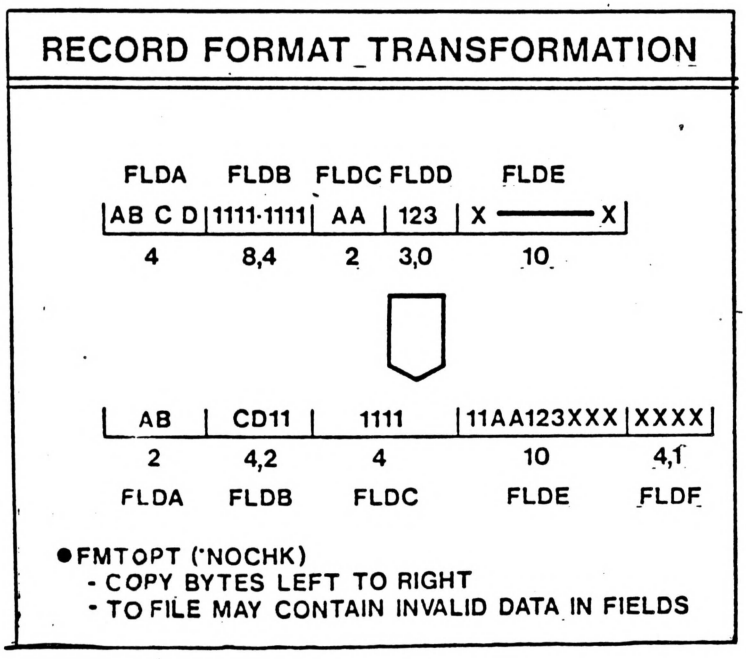
° Record formats (RCDFMT)

When you copy from a logical file you must specify which record format is to be copied. This is not the same as copying from the underlying physical file because the logical file may be applying selection and omission criteria. RCDFMT(*ALL) can be specified when copying to a device file such as a printer. but is not the default which is * only. If *ALL (multiformat LF), you cannot specify INCREL parameter.

Record Format Transformation

If the record formats are not identical in the from file and the to file, you can specify the dropping or the mapping of fields. Alternatively, you can copy byte-by-byte from the from-file to the to-file without regard to the record formats. No checking is done as to the validity of the data arriving in the to-file if this option (FMTOPT(*NOCHK)) is chosen.





Printing Records (PRINT)

The to-file may be specified as *LIST. The Copy File commands use the IBM-supplied printer file QSYSPRT. Character print, ~~Hexadecimal print or both can be chosen~~. with or without hex print can be chosen on the PRTFMT parameter

The PRINT keyword enables you to print those records that were selected for copying, or those records that were not selected thus providing an audit trail. CPYF FREDFILE TOFILE(*LIST) PRINT(*COPIED), whilst illogical, is an allowable combination of keywords simply to demonstrate that S/38 has a sense of humour.

Member Copying (FROMMBR,TOMBR,MBROPT)

CPYF allows you to choose which member to copy from and to. The to-file can either have records replaced or added as the result of the copy.

Undefined To-files (CRTFILE)

CPYF can create a file for you to hold the copied records. You need authority to the CRTPF command, the resulting file is created with PUBAUT(*NONE) and the owner of the created file is the owner of the from-file. You can only create a file in this manner when copying from a database file.

Source Copying

CPYSRCF allows the copying of source members between source files with the option of a formatted listing.

do not use CPYF interactively: use option 6 on pr. menu to SBMJOB (1 cmd only)

Exercise 11

1. Use CPYF to copy the records from CUSTP.SF38LIB to CUSTP.SF38TEAMXX.

2. Use CPYF to produce a listing of the new physical file into your own output queue. There is no reason to actually print the records. ↳ QPRINT2

3. Use the SEU Browse/Copy feature to create a copy of the statements in the member FLDREFFILE in QDDSSRC.SF38LIB in your QDDSSRC. Then create FLDREFFILE in your library.

QUERY

This Utility is designed to generate reports from information in the Data Base.

To request QUERY to build an 'application', use the DSNQRYAPP command or take option 2 on the Programmer's Menu, then respond to the prompts presented (note the existence of a 'Fast Path' option).

It is now necessary to create the application, using an option from a menu, which gives the choice of creating in either interactive mode or batch mode.

The application could now be executed, by a menu option, as often as required without repeating the build or create.
It is also possible to change/move/rename/delete an existing 'query' or to inquire about the status of one submitted for execution.

Accessing the Data Base

QUERY can only be used to access files (logical or physical) which are externally defined.

For each report, the following can be specified:

- a title
- page headings
- column headings
- field editing
- column spacing
- the left-to-right sequence of columns

The types of report from which a choice can be made are:

- detail, showing the information contained in each record
- summary, showing field totals or averages
- combinations of detail and summary.

Other optional features include:

- selection of those records containing fields which are less than, equal to, or greater than, the contents of other fields from the same format, numeric constants or computed result fields.

- sampling - for example, showing in a report the first 200 records of a file, or every 10th record.

- sorting - in the sequence of either existing fields or computed result fields.

- computation - accumulating field values for totals/subtotals.

  - averaging a field in the set of records processed.

  - multiply/divide/add/subtract values, or extract a remainder from a value. The values can be numeric constants, the contents of fields within a record format, or computed result fields. If fields are to be used which are not defined in the Data Base they must be added to the prompt screen for selected fields, together with a number representing their horizontal positions on the report.

- composition of tables, to assist identification of patterns, trends and exceptions. Fields are selected as 'table' fields, and tabulation values are specified - eg. 'balances' of less than £50, greater than £50 but less than £100, and so on. 'Table data' can be printed in addition to the 'report' information.

Exercise 12

Use the logical file CUSTL1 to print a report of Customer records.

The following fields from the format CUSTPR are required:

CUST
NAME
CRLIM
BALANC
YRTOTL
SCOPE (a result field)
APERCT (another result field)

The file should be sorted on the balance field in descending sequence.

Create the result field SCOPE by subtracting BALANC from CRLIM. The field APERCT represents SCOPE times 100 divided by CRLIM.

Only select those records where CRLIM is greater than ten.

Accumulate the BALANC and YRTOTL fields.

Set up a table to show the range of balances outstanding, with the average balance per Interval of £1000, evenly spaced.

FILE REFERENCE FACILITIES

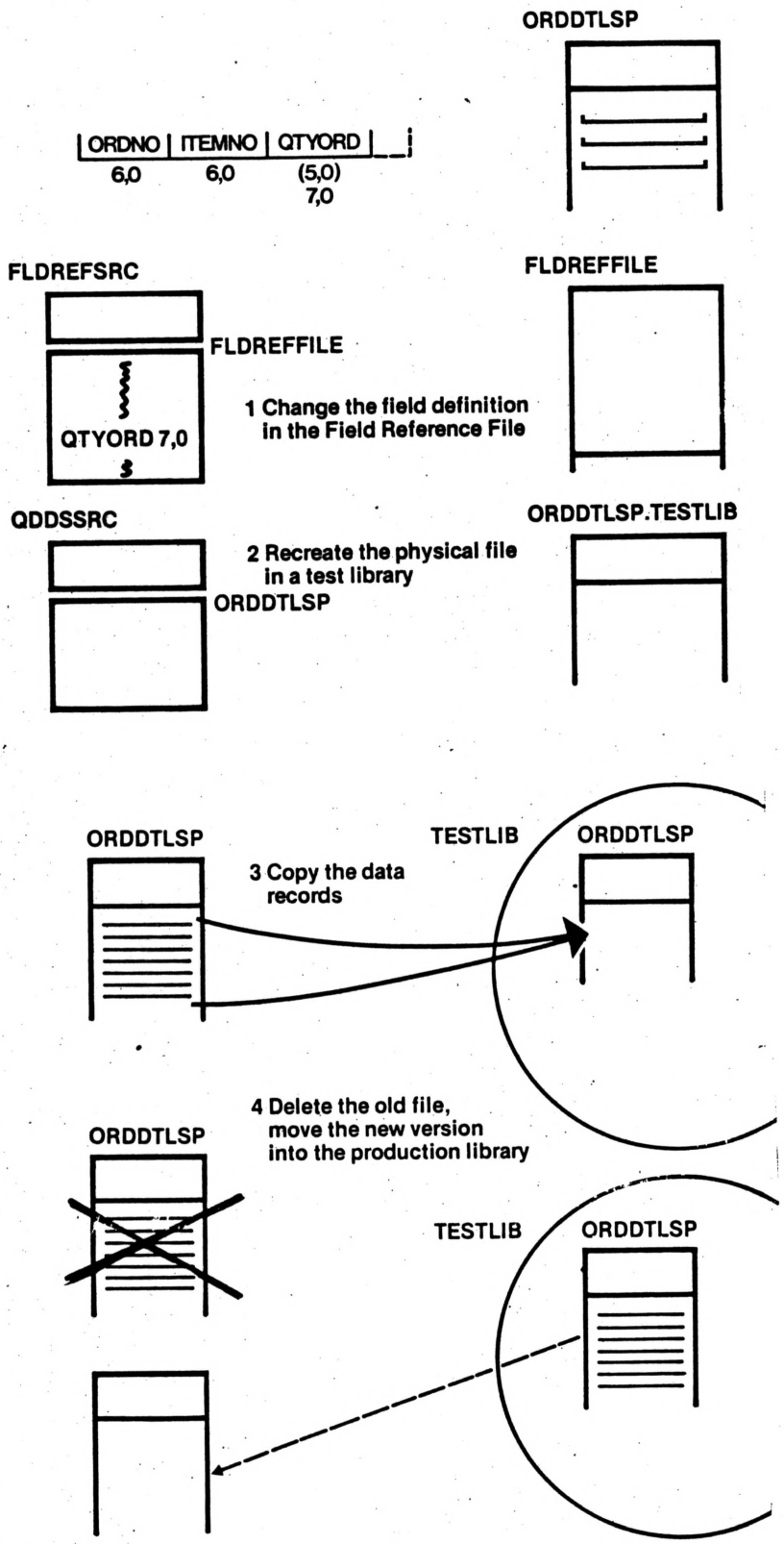
CHANGING THE FILES

Making changes to the physical files
Record format level checking
The file information commands

Exercise 13

Making changes to the physical file - maintenance

To add a new field or change the attributes of an existing field (its length for example) the physical file must be re-created.

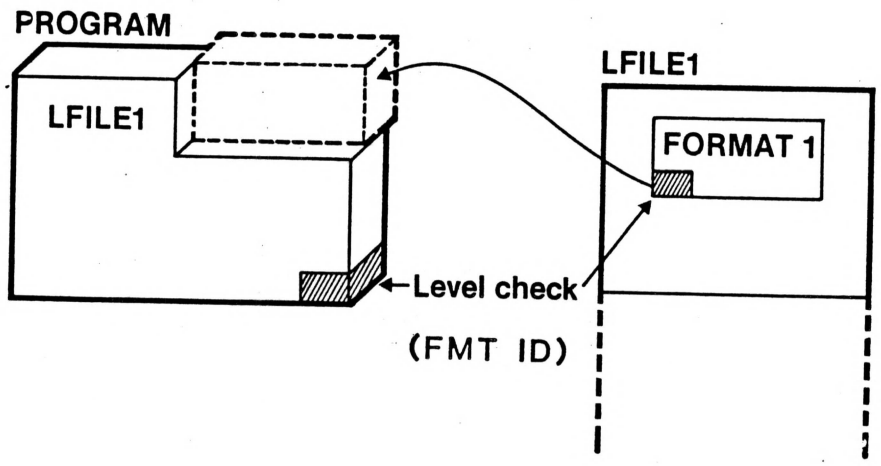


A physical file cannot be deleted unless all logical files based upon it are deleted first. So any change to a physical file means:

(1) all logical files which use it must be re-created
(2) all programs which use any changed field must be re-created

Record format level checking

To ensure integrity when changes are made to record formats, the system checks record formats whenever a file is opened, to ensure that the input and output buffer layouts have not changed since the program was created (unless LVLCHK(*NO) is specified).
Firstly, the format names are checked. If a format name is found which was not present in the file at the time the program was compiled, a level check will occur.
Next, if the format names are the same, the format identifiers are compared with those that were in the file at the time of compilation. If an identifier is different, the format layout will be checked to ensure that it has not changed. If it has, a level check will occur.



If a field is being added to a file, the original source statements for a logical file can be used to recreate it provided that it does not need the new field. Only when a field is changed in its attributes will it be necessary to recompile any programs.

Making changes to Physical file or Logical file attributes

It is possible to change permanently attributes of Physical or Logical files by using the commands CHGPF, CHGSRCPF and CHGLF. These require that as well as the necessary authority, the object to be changed has no other users.

Time for rebuilding AP : 20 byte key : 10,000 Rec/min

eg recreating a L.F.

File information commands

Various cross-reference and informational commands are available to help the system designer and programmer:

DSPFD — Display File Description - creation date, format names, access path specs etc. A particularly useful option is *MBRLIST, giving information for each member in the file (name, record count, size, creation date and text.

DSPFFD — Display File Field Description - all DDS information as held in the file description.

DSPDBR — Display Data Base Relations - files using named format, logical files based on named physical file, files sharing access path.

DSPPGMREF — Display Program References - all external objects referenced by the named program (eg. files, data areas, called programs).

DSPOBJD — Display Object Description - owner, creation date, save/restore data etc. Consider also specific uses:

```
DSPOBJD filename.*ALLUSR  *FILE
     (in which library is the file?)
DSPOBJD *ALL.libname  *FILE
     (list of files in library, with text etc)
DSPOBJD partname*.*ALLUSR  *FILE
     (all files beginning with xxx)
```

For all the above commands output can be to a data base file (see CPF Programmer's Guide, Chapter 15, for layouts), which may then be the subject of a QUERY application, or processed as required.
Other commands can achieve the same effect by copying the printer file to a data base file, using the CPYSPLF command.

Where changes are to be made to a physical file format, the changes should first be made to the field definitions in the appropriate Field Reference File (or Files!). Next, use DSPFFD across all production libraries with output to a DB file. A Query application can now be defined, giving the names of the changed fields as selection criteria to list out the affected files - some will be physical, some logical, and some display.

All the display and physical files will have to be re-created, but the list of logical files may not be complete.

DSPDBR should be used to establish the names of all logical files based on the physical files affected. Some logical files may not use the fields which have been changed, but still have to be created over the new versions of the physicals.

Lastly, establish which programs use the files (including display files) affected by the changes. Use DSPPGMREF, again across all production libraries. A Query application can list out the names of programs by file and format name.

We can now list out the programs which have to be re-compiled, after the new versions of the files have been created. The only exceptions will be those programs which are on the list because they use logical files based on one of the changed physical files, but do not use any of the changed fields. These logical files can be created over the new physical(s) using the current (old) logical file format, thereby retaining the old record format identifier. This will keep the time required to open files down to the minimum, since the level checking done will be limited to a check of the format names and identifiers.

Two other "cross-reference" commands which can be useful are:

LSTCMDUSG	List Command Usage
DSPPGM	Display Program (attributes)

Exercise 13

The object of this exercise is to modify a field in your Field Reference File and then change the relevent files and programs.

1. Change the field BALANC in your Field Reference File to (8 2).

2. Determine what needs to be changed and then create the new files and programs in your library.
   The program source for CUSINQ is supplied in QRPGSRC.SF38LIB.

3. Make certain that the objects to be created use <u>your</u> library for <u>any</u> <u>dependency</u>.

1- EDTSRC QDDSSRC
2- DLTF FLDREFFICE (or CF11)
3- CRTPF " MBR(*NONE)
4- " CUSTP2 SRCMBR(CUSTP)
5- CPYF CUSTP CUSTP2 FMTOPT(*MAP) MBROPT(*ADD)
6- DLTF CUSTL1 & all CUSTP based logical files
7- " CUSTP
8- RNMOBJ CUSTP2 *FILE CUSTP (RNMMBR is also valid too)
9- CRTLF CUSTL1 (after changing source for DDS ~~*SF38LIB~~ remove * SF38LIB)
10- DLTF CUSINQDF (or CF11)
11- CRTDSPF CUSINQDF
12- DLTPGM CUSINQ
13- ~~CRTCLPGM CUSINQ~~ (or CF11)
CRTRPGPGM CUSINQ

15/02/84 14:43:50 FILES USING THE BALANC FIELD. PAGE 1

FILE	LIBRARY	FILE CREATION DATE	TYPE OF FILE	NBR OF FMTS	RETRIEVAL DATE AND TIME	RECORD FORMAT	FORMAT LEVEL ID
CUSTP	SF38TEAMXX	0840215	P	1	0840215143852	CUSTPR	360E648880E41
PHYSICAL CUSTOMER MASTER							
FLDREFFILE	SF38TEAMXX	0840215	P	1	0840215143852	JRDREFFILE	46DBA7B4D2288
FIELD REFERENCE FILE							
CUSTL1	SF38TEAMXX	0840215	L	1	0840215143852	CUSTPR	360E648880E41
PHYSICAL CUSTOMER MASTER							
CUSINQDF	SF38TEAMXX	0840215	D	2	0840215143852	CUSFMT2	138313874E5D7
CUSTOMER DETAILS							

FORMAT TEXT DESCRIPTION	NBR OF FLDS AND INDS	FORMAT LENGTH	INTERNAL FIELD NAME	EXTERNAL FIELD NAME	OUTPUT BUFFER POS	INPUT BUFFER POS	FIELD LENGTH	DECIMAL DIGITS
CUSTP	14	133	BALANC	BALANC	115	115	4	6
FLDREFFILE	60	484	BALANC	BALANC	2	2	4	6
CUSTL1	14	133	BALANC	BALANC	115	115	4	6
CUSINQDF	11	129	BALANC	BALANC	124	0	5	6

TOTAL NUMBER OF RECORDS PROCESSED 4

15/02/84 15:09:52 FILES USED BY PROGRAMS. PAGE 1

OBJECT	LIBRARY	SOURCE FILE NAME	NBR OF FMTS	FILE USAGE	RECORD FORMAT	FORMAT LEVEL ID	NBR OF FIELDS	PROGRAM
CUSINQDF	SF38TEAMXX	CUSINQDF	2	3	CUSFMT1	133F9FF186818	0	CUSINQ
				2	0840215150838			
CUSINQDF	SF38TEAMXX	CUSINQDF	2	3	CUSFMT2	138313874E5D7	0	CUSINQ
				2	0840215150838			
CUSTL1	SF38TEAMXX	CUSTL1	1	1	CUSTPR	360E648880E41	0	CUSINQ
				2	0840215150838			

PROGRAM TEXT DESCRIPTION	OBJECT COUNT	RETRIEVAL DATE AND TIME

TOTAL NUMBER OF RECORDS PROCESSED 3

15/02/84 14:44:12 REPORT SHOWING PHYSICAL FILES AND DEPENDANT LOGICALS. PAGE 1

TYPE OF FILE	FILE	LIBRARY	MEMBER	RECORD FORMAT	DEPENDENT COUNT	RETRIEVAL DATE AND TIME	DEPENDENT FILE	DEPENDENT LIBRARY	DEPENDENT MEMBER	TYPE OF DEP
P	CUSTP	SF38TEAMXX	*NONE	*NONE	1	0840215143951	CUSTL1	SF38TEAMXX		D
P	FLDREFFILE	SF38TEAMXX	*NONE	*NONE	0	0840215143951				

TOTAL NUMBER OF RECORDS PROCESSED 2

DSPDBR
WALL WALL

SAVE/RESTORE AND JOURNALLING

1. THE SYSTEM/38 FACILITIES
   - Save/Restore
   - Journalling

1. THE SYSTEM/38 FACILITIES

System/38 offers various aids in designing an application for efficient recovery.

- o System/38 precautionary aids

  Forcing files can be effected by the FRCRATIO parameter (when data base logging is used, only log and transaction files need normally have FRCRATIO of 1).

  RPG III programs may use the FEOD operation code to force out data at strategic points.

  Data areas are always forced when changed.

  Message queues will be forced if created with the parameter FORCE(*YES).

- o System/38 facilities to aid recovery

  Abnormal termination message on IPL display.

  Automatic recovery of all out-of-synchronisation access paths (at IPL or file open)

  Automatic application of journal records to any physical files which were open at the time of an abnormal termination.

  Override Data Base File command - parameters INHWRT or POSITION.

What you can Save and Restore (Ref: CPF Programmers Guide Chapter 22.)

The Save/Restore facilities enable you to save backup copies of objects (except queues) on your system to help you recover from program or system failure.

The following list shows what you can save and restore:

You Can Save and Restore:

- A single object in a library
- A group of objects by generic name
- A group of objects by generic name and object type:
  - Command definitions
  - Message files
  - Tables
  - Programs
  - Print images
  - Classes
  - Edit descriptions
  - Subsystem descriptions
  - Files
  - Job descriptions
  - Data areas
- An entire library
- The system (QSYS, device configuration, and user profiles)

USRPRF etc cannot be saved except with SAVSYS contents of Q can't be saved.

Saving Objects

When an object is saved, it is not removed from the system. However, if you do not want to keep an object on the system, you can ask that an object's storage be freed. This makes the storage being occupied by the object available to the system. The object is then considered offline and must be restored before you can use it. This is not the same as deleting the object; references to the saved object that has its storage freed will result in an 'object unavailable' condition rather than a 'not found' condition. You can still examine the object's description - it will tell you the object is saved and where it is. (For a logical file, only the definition of the file is saved, not the Access Path.)

When Objects can be Saved

The system will require exclusive update control of an object in order to save it. If you are saving a library and the system cannot lock one of the objects in the library, the saving of that object is bypassed and a message issued. In Rel 5 it is possible to avoid this by specifying PRECHK(*YES) in the save command. When the following save operations are taking place, all subsystems must be terminated (except QCTL) and only one interactive job in QCTL may be executing:

- Saving the entire system
- Saving all user libraries with one command

The system will tell you how many objects were saved and if any could not be saved, why not, at the completion of the save operation.

MSGQ : do FORCE(*YES) to get diagnostics

Save Media

Diskette - 2D diskettes only - capacity 1.2 Mb
2 magazines can be loaded (ie. 24 Mb on-line)
The diskettes must be initialised in SAVRST format, but this can be carried out 'in flight'
The volume id on each diskette in a magazine should have as its last character (eg.) SAVE1, SAVE2....SAVE0.

Tape - The 3410/3411 Subsystem 2400' tape will hold about 42 Mb of data (1600 bpi)
The 3430 holds approximately 3.5 times more, and has Data Compression/Decompression capability for Save/Restore
The tape must be initialized as a standard label tape, and as with diskettes, can be carried out 'in flight'
For after a SAVE, you can specify rewind, left as it is, or rewind and unload

online save file, then ~~SAVSAVF~~ SAVSDTA

R7:

Transfer Speeds - Diskette: 50-100 Mb per hour
Tape: 100-900 Mb per hour (fastest model)

Note that the speeds will vary according to what is being saved - there is an overhead for each object saved, so one large file will be saved at a faster rate than many small objects.

multimember file: each member is treated as an object

SAVRST commands

SAVOBJ	one object or up to fifty named objects up to fifty named libraries generic name (eg PAY*) or *ALL all objects of a type (eg *FILE)
SAVCHGOBJ	objects (by type if wished) changed since the last save or since a specified time/date from up to fifty named libraries
SAVLIB	whole library or up to fifty libraries *NONSYS

If multiple libraries are specified on one of the above commands, processing which occurs at the beginning and end of each library is overlapped with the actual output to media to save another library.

SAVSYS	system library, configuration, user profiles
RSTLIB	equivalent of SAVLIB
RSTOBJ	can restore individual objects from a saved library

Restoring the System

If it is necessary to restore the system you should restore from the most recently saved version. You could also restore from the original installation material but that will not contain your user profiles, line descriptions, device descriptions etc. The required sequence is:

1. IPL from the saved system diskettes.
2. Restore User Profiles (RSTUSRPRF).
3. Restore user libraries and QGPL.
4. Restore to the user profiles, object authority for the user libraries (RSTAUT).

EXISTENCE & needed to SAV, + *SAVRST
MANAGEMENT (rename etc)
operational

Points to Note

For each object, the object description stores information as to the most recent save and/or restore for that object. Date, time and volume-id are stored; the system will then reject an invalid restore attempt. *JOBCTL authority

Certain conditions must be met before a save can be performed.

- user must have special authority for SAVRST noted in his user profile
- user must have existence rights to the object
- object must be exclusively locked against update

Contents of queues (JOBQ/MSGQ/OUTQ) cannot be saved.

Building Save/Restore into the machine schedule

It is obviously impracticable to consider saving the whole of auxiliary storage to diskette. (For a 2.6 Gb machine this could take about 30 hours and need some 2,200 diskettes.)

With System/38, the aim is to save the actual data, not the space, and is therefore more oriented to the application.

Because you may wish to save volatile files (transaction and log files, and journal receivers) whilst they are in use you can Copy the files involved, which only requires read only control to another library and save the files from that library. This could be built into a CL program that the operator runs last thing at night, the program being responsible for powering off the system. When the operator starts the system (auto start job?) the files could be saved offline from the save library.

Periodic save routines might be organised as follows:

Unattended batch job at end of day

```
PGM
CLRPFM  FILEA.SAVLIB
CPYF    FILEA.LIBA   FILEA.SAVLIB
CLRPFM  FILEB.SAVLIB
CPYF    FILEB.LIBB   FILEB.SAVLIB

... etc

PWRDWNSYS     *IMMED
ENDPGM
```

Attend job following morning (autostart job?)

```
PGM
SAVLIB  SAVLIB  CLEAR(*YES)
DSPDKT  *M12  DATA(*SAVRST)  OUTPUT(*LIST)
ENDPGM
```

Advantages:

- o live files copied out of normal hours to avoid problems of exclusive locking
- o off-line save run whilst operator on hand
- o faster disk-to-disk recovery available for minor disasters

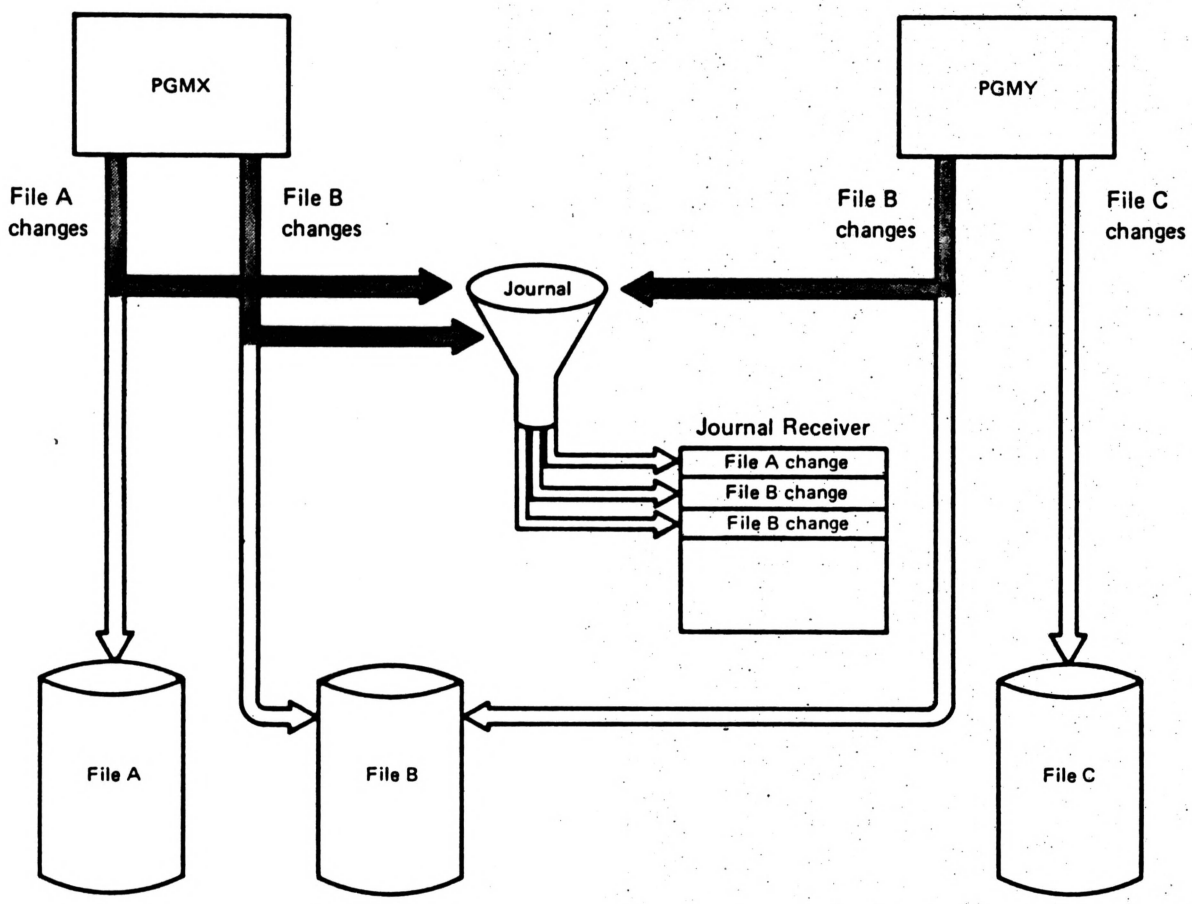
The following table suggests an approach to frequency of save.

OBJECT	DESCRIPTION	SAVE FREQUENCY
Machine Code	Microcode supplied on the system and also as backup on diskette	Never
QSYS	System library (CPF, user profiles, device descriptions, etc)	Before ~~At~~ a new release or major change
Application Libraries	User written programs, all file types except physical data files	At installation or re-installation
Master files	Basic data base physical files for the applications	Depends on volatility
The rest	Transaction files, log files, journal receivers, data areas and development libraries	Daily or twice daily

- Journalling

This function of the Control Program Facility effectively supplants data base logging. It is operable at the machine level, rather than at the user interface, and is therefore more efficient in operation. As with data base logging, the user defines the use to be made of the facility, but the definition is system wide,rather than job by job.

The user will create a new type of object called a journal receiver. This will contain journal entries. Having created a journal receiver, the user will create a journal, another new system object type, to which will be added a list of the physical files to be journalled through it, in the named journal receiver.



First, the journal receivers to be used are created: - The command

::CRTJRNRCV JRNRCV(RCVDST1.DSTJRN) THRESHOLD(1000)

- creates a journal receiver RCVDST1 in the library DSTJRN. When the size of the receiver exceeds 1000K the system sends a message (CPF7099) to the message queue specified on the CRTJRN or CHGJRN command. This message could then be used to automate the change of receivers.

The receiver record size will be about 75 bytes (system generated) plus the record size of the file being journaled. The file size will be determined by the number of changes per day and how many images are required.

Next, the journal for this group of receivers is created: - The command

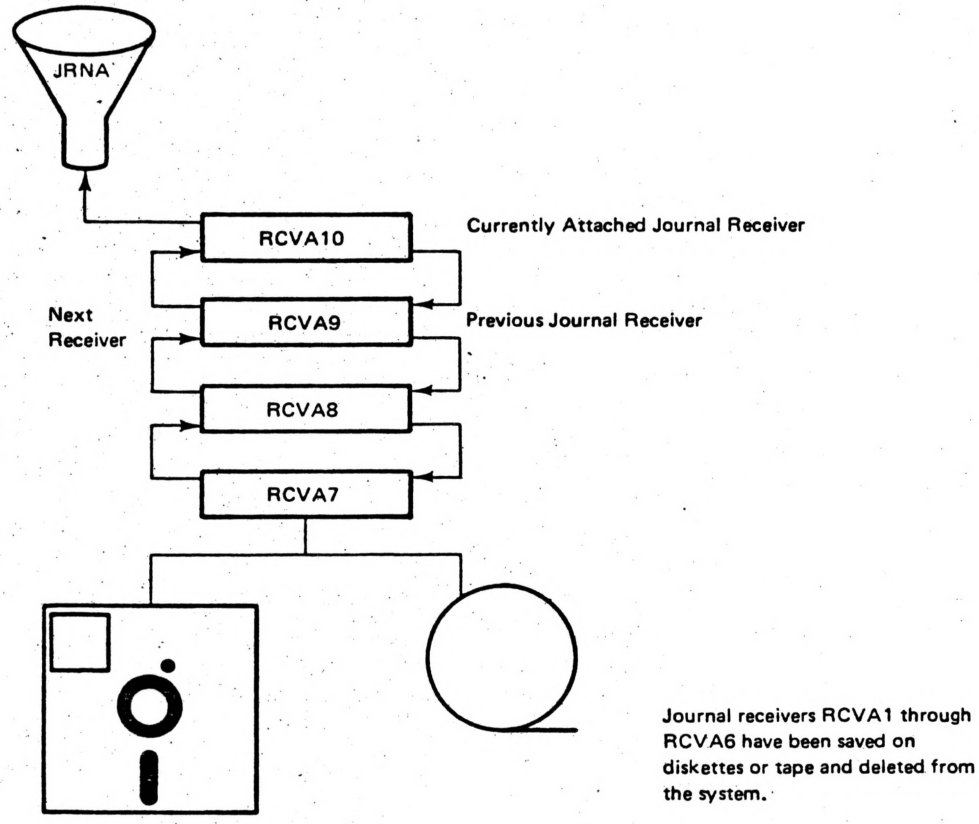
::CRTJRN JRNLA.DSTJRN JRNRCV(RCVDST1.DSTJRN)

- attaches the journal JRNLA to the receiver RCVDST1.Once a journal has been created, physical files are added to the list it will cover by using JRNPF command. The user will determine whether 'before' or 'after/both' images are to be recorded for each file. For example

::JRNPF FILE(ORDENTP.DSTPRODLIB) JRN(JRNLA.DSTJRN) IMAGES(*BOTH)

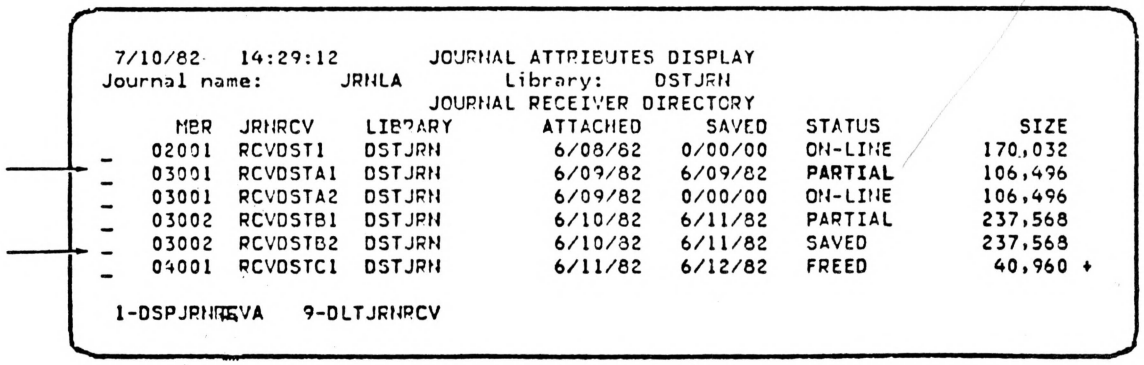
The journaling of a file can be stopped by using the ENDJRNPF command. It is possible, for added security, to specify that the journal will use duplicate journal receivers.

At any time, the user can use the CHGJRN command to replace the receiver or receivers in use by a particular journal, even if the files being journalled through the journal are open. As a new journal receiver is attached to the journal, the previously attached journal receiver is automatically detached and can then be saved. The new receiver will be linked to the old, and can be created by the system with the same generic name as the receiver being detached (*GEN option), so that user tracking of receivers can be more easily managed.



Once a receiver is detached from a journal, it cannot be re-attached to take more entries, but the system will have recorded its existence. If it is deleted after being saved, it can be restored when required, but there is obviously an advantage in keeping all receivers on line for as long as they are needed.

You can display the current status of the journal receiver by using the DSPJRNRCVA command. The journal receiver attributes display identifies the journal to which that receiver is, or was, attached. It can also identify the dual journal receiver and the journal receivers attached previously or subsequently to a given receiver.



```
 7/10/82   14:29:12        JOURNAL ATTRIBUTES DISPLAY
Journal name:       JRNLA          Library:     DSTJRN
                              JOURNAL RECEIVER DIRECTORY
      MBR   JRNRCV     LIBRARY       ATTACHED     SAVED     STATUS          SIZE
_   02001   RCVDST1    DSTJRN          6/08/82   0/00/00   ON-LINE      170,032
_   03001   RCVDSTA1   DSTJRN          6/09/82   6/09/82   PARTIAL      106,496
_   03001   RCVDSTA2   DSTJRN          6/09/82   0/00/00   ON-LINE      106,496
_   03002   RCVDSTB1   DSTJRN          6/10/82   6/11/82   PARTIAL      237,568
_   03002   RCVDSTB2   DSTJRN          6/10/82   6/11/82   SAVED        237,568
_   04001   RCVDSTC1   DSTJRN          6/11/82   6/12/82   FREED         40,960 +

 1-DSPJRNRCVA   9-DLTJRNRCV
```

When can receivers finally be destroyed (DLTJRNRCV command)? The answer to this question must be 'when the save of the group of files to which they apply is overwritten by a new save.

Recovery Using Journalled Changes

After an abnormal termination of the system, the data base files will be automatically synchronized with the journal during IMPL. The system places a message (CPF7097) in the system history log to identify any files that could not be synchronized with the journal.

Having decided which files are to be recovered you can use the Display Journal (DSPJRN) command to identify the desired start point for normal processing. To make it easier to determine what has actually changed in the records the new command Compare Journal Image (CMPJRNIMG) can compare the before image with the after image, or the after image with the previous after image, and mark the difference character by character on the system printer.

The files are then recovered by either reloading the master files and applying the journal changes (APLYJRNCHG), or by simply removing the changes (RMVJRNCHG) from the current files. The amount of changes to be applied can be controlled from within the commands. For example

```
::RMVJRNCHG JRN(JRNA.JRNLIB) FILE(FILEA.DSTPRODLIB)
RCVRNG(*CURRENT)
```

removes the changes in journal JRNA from the member of FILEA. The system starts removing the changes beginning with the latest entry for that file on the currently attached journal receiver and continues to the earliest entry for that file on the currently attached journal receiver.

- Commitment Control

This is an additional facility which allows you to:

- **Ensure that all changes completed by a single transaction are completed for all files affected. If an unscheduled interruption occurs during the processing of the changes, you are assured that when the system resumes processing, all the changes are in the files or all the changes have been removed from the files.**
- **Design for application restart in case of a routing step or system failure.**
- **Remove any changes made during a transaction when the work station user determines that the transaction is in error.**

In addition, commitment control allows you to lock multiple records within a file, lock 'read-only' records, and wait for a lock on a 'read-only' request.

Using commitment control you can define and process a number of changes to data base files as a single unit of work (or transaction). For commitment control, a transaction is defined as a group of changes made to data base files that appear to be a single change to the work station user. For example, a work station user enters information to transfer funds from a savings account to a checking account, which to the user is a single change. However, to complete the transfer, the program must update both the savings record and the checking record.

If you place data base files under commitment control, you can ensure that *all* changes in a single transaction are completed before any of the changes are made permanently to the file. In addition, if the work station user detects an error in the transaction, the program can remove (rollback) all the changes made for that transaction.

All data base files under commitment control must be journaled to the same journal, and both before-images and after-images are journaled. (If you specify only after-images when you begin journaling your files, the system automatically journals the before-images while the files are under commitment control.) When changes are made to these data base files, entries are placed in the journal for each change generated by the transaction. When the transaction is completed, the program executes a commit operation (COMMIT command in CL, COMIT operation in RPG, and COMMIT verb in COBOL) that places a separate entry on the journal to identify that the changes generated by the transaction are committed–that is, that they are made permanently in the file.

SECURITY

User Profiles
Object Authority

Reference: CPF Programmers Guide Chapter 21.

The two principal concepts involved with security are:

LF can be used too to control access at rcd and fld level.

1). User Profiles

2). Object Authority

A User Profile is an object that represents a user to the System. A number of individuals can sign on under a common User Profile. Whenever any object is created on S/38, that object will contain the name of the User Profile under which the object was created - the initial "owner" of the object. It will also contain the "public authority" - authority that other User Profiles have - to the object. The User Profile under which the object was created will have an entry added to it to reflect ownership of that object.

When that object is subsequently referenced, the system will check to see if the object itself is secured (ie. NO public authority). If it is, the system will check to see if the User Profile referring to that object has the appropriate authority to it, either directly or via a Group Profile - another User Profile "adopted" for the job at sign on.

Note that the system does not perform this procedure time and again for further references to an object within the job. Once found, addressability to the object is stored in a system pointer to reduce the time taken to service subsequent requests to that object. In addition, the pointer contains the authority the user has to that object.

For example, if you issue the Delete File command, the System must check that you are authorised to delete the file, to use the command and to reference the libraries involved. The diagram over the page illustrates the components of the User Profile object.

The User Profile

USER NAME	Identification for the System - authorisations made to User Name
PASSWORD	Access at sign-on controlled by User Password
OWNED OBJECTS	Addressability
AUTHORISED OBJECTS	Addressability and right of use
SCHEDULING PRIORITY	Highest authorised
SPECIAL AUTHORITY	Authority to special functions - Job control and/or Save/Restore
STORAGE	Maximum auxiliary storage allocated
INITIAL PROGRAM	To be invoked automatically after sign-on
JOB DESCRIPTION	If specified, this Jobd will apply unless overridden in JOB/SBMJOB
ACCOUNTING CODE	If specified, this code will apply unless overridden in JOB/SBMJOB

IBM supplies six user profiles with the system. The chart below is reproduced from the CPF Programmer's Guide, chapter 21.

- *Security officer user profile (QSECOFR).* Controls the authorization of functions and data used in your installation. (See *Security Officer User Profile (QSECOFR)* later in this chapter for more information.)

- *Programmer user profile (QPGMR).* Contains the authority necessary for system and application programmers and the special authorities of save system rights and job control rights. QPGMR is specified as the user profile for the IBM-supplied batch job description QBATCH. The initial program for QPGMR is QPGMMENU (programmer menu), and the initial user password is PGMR. This profile can be changed or deleted. Note that IBM-supplied objects owned by QPGMR have to be changed to a different owner by using a Change Object Owner (CHGOBJOWN) command, before QPGMR is deleted.

- *Work station user profile (QUSER).* Contains the authority necessary for work station users. QUSER is specified as the user profile for the IBM-supplied interactive job description QINTER. The initial program for QUSER is QCALLMENU (program call menu), and the initial user password is USER. This profile can be changed or deleted.

- *System operator user profile (QSYSOPR).* Contains the authority necessary for the system operator, and the special authorities of save system rights and job control rights. The initial program for QSYSOPR is QOPRMENU (system operator menu), and the initial user password is SYSOPR. This profile can be changed or deleted.

- *Programming service representative user profile (QPSR).* Contains the authority necessary for the PSR to service the system programming, and the special authorities of save system rights and job control rights. The PSR should not be revoked from the system console and, like other users, can be authorized to display work stations. The initial user password is PSR. This profile cannot be deleted.

- *Customer engineer user profile (QCE).* Contains the authority necessary for the CE to perform diagnostics and service the machine and the special authority of job control rights. The initial program for QCE is QSMCSMSU (concurrent service monitor), which is used at the system console to diagnose system problems. The initial user password is CE. You cannot delete this profile, nor can you have more than one CE user profile. The 5250 test request is a special support operating under the QCE profile, which performs device servicing functions from a special menu; the CE cannot access the command entry display from this special menu. The CE should not be revoked from the system console. The CE can use the 5250 test request support if the work station was created with public authority (PUBAUT(*NORMAL)) or if the CE is authorized to use the work station.

Object Authority

The creator of an object can specify the authority the "public" has to the object he creates. The creator is the owner until such time as the object is "given" to somebody else (CHGOBJOWN command). Once the object is deleted, all reference to it is deleted by the System.

Every CREATE command has a PUBAUT keyword. Permitted values are:

*ALL - The object is public and all users have all rights to the object (see later).

*NORMAL (Default) - The public has limited authority to the object, ie. to perform operations upon that object which are regarded as normal for the object (eg. a program can be CALLED).
The public do not have any means of affecting the existence or description of the object.

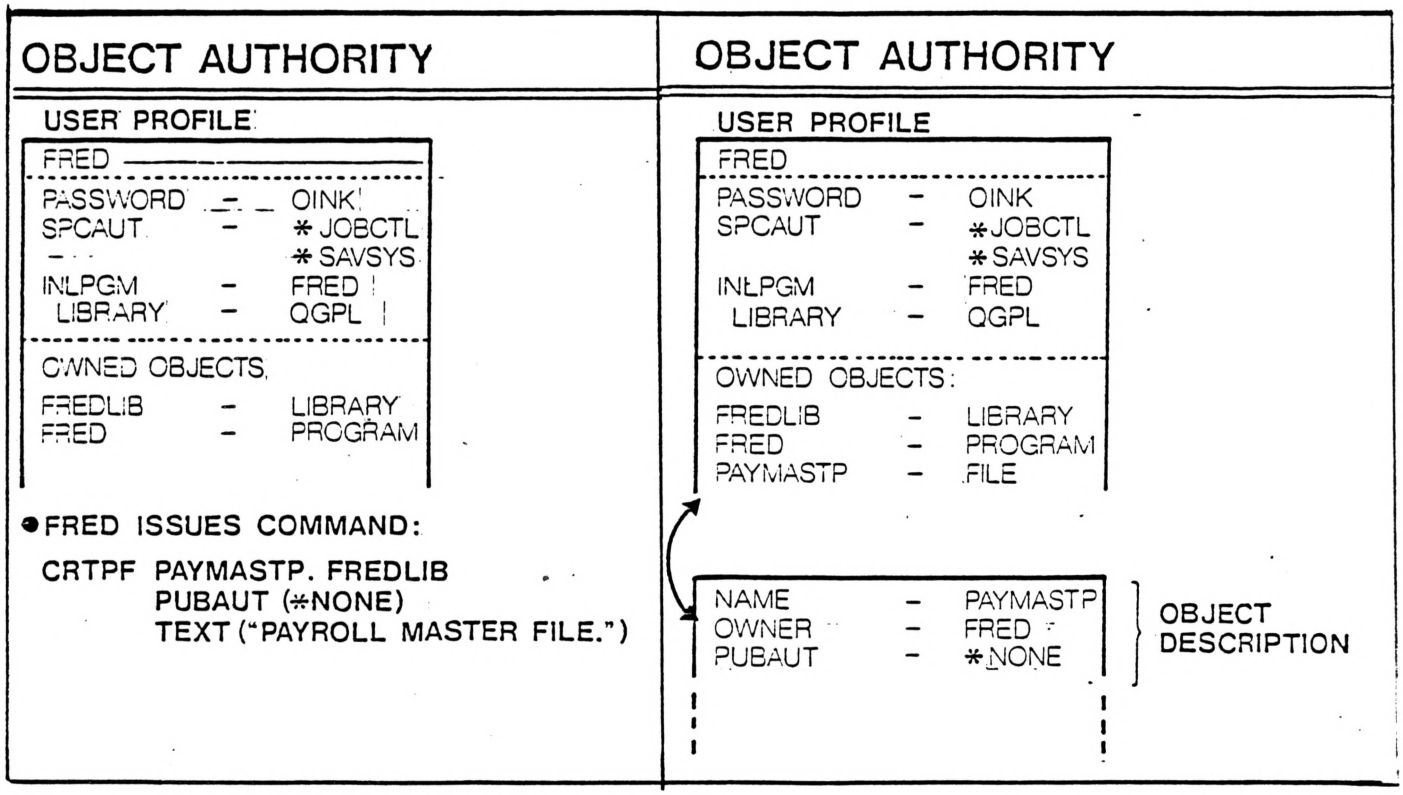
*NONE - The object is private - available only to the owner and the security officer (QSECOFR).

Owners of objects, and the security officer, can grant and revoke different authorities for objects to/from other named users - the commands are GRTOBJAUT/RVKOBJAUT, with wide-ranging parameters.

An object can only have one owner.
Be aware that if ownership is changed, the creator of that object retains full rights to the object. If this is not desirable, the new owner (or the security officer) must revoke rights to that object as appropriate.

DSPOBJAUT

The following diagrams illustrate the connection between the object and its owner:



Associated with overall object authority are the rights controlling the use of that object:

OBJECT RIGHTS — Those rights that affect the object's description and its very existence.

DATA RIGHTS — Those rights that affect the components that make up the object (ie. the data or functional part of the object).

Object rights fall into three catagories:

Existence rights
confer the right to delete, save, free the storage of, restore and transfer the ownership of an object.

Management rights
confer the right to move, rename, grant or revoke authority, and generally change the attributes of an object.

Operational rights
confer the right to use an object and look at its description.

Data rights control the use of the contents of the object:

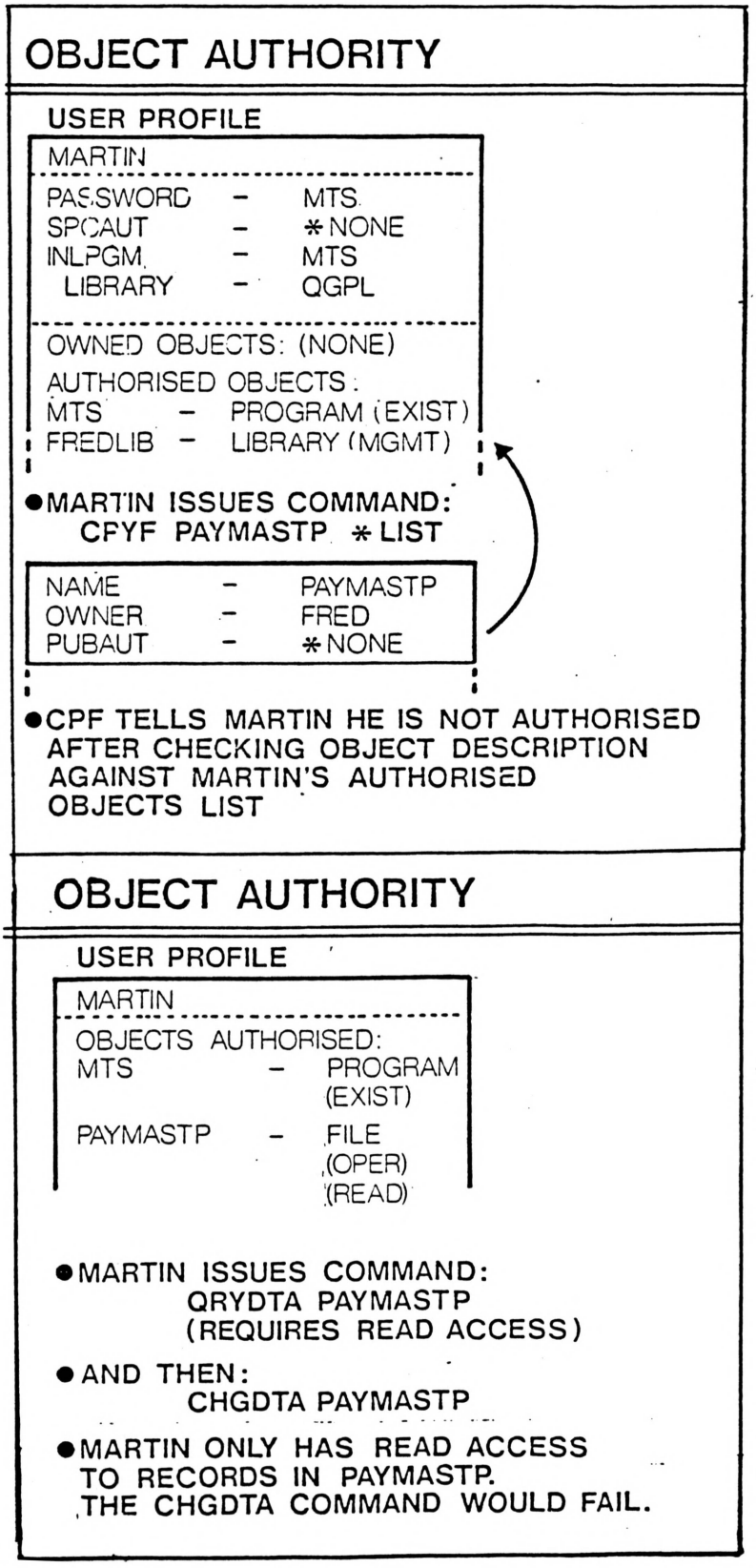
READ rights.

ADD rights.

UPDATE rights.

DELETE rights.

These two diagrams illustrate the manner in which object authority is checked when an object is referred to:



Using Appendix C of the CPF Programmer's Guide, what minimum levels of authority are required to execute the following command successfully ? (How many objects are involved !?)

MOVOBJ FRED.FREDLIB *PGM TOLIB(MASTLIB)

operational management operational + delete operational + add

Job Descriptions and Security

Earlier in the course, the importance of the Job Description in determining the library list to be used by a job was stressed. The same also applies to the security aspects of batch jobs.

For these, the user profile is identified by the USER parameter in the job description. Any user profile name can be specified, except QSECOFR, but *RQD is not acceptable as it is in an interactive job description.

Some other Security provisions

A limit on the number of invalid sign-on attempts can be applied through the System Value QMAXSIGN.
The System Serial Number can be accessed in the System Value QSRLNBR should control of "export/import" of objects between different machines be required.
Software developers can specify that CL program source code is not saved with the program object, effectively prohibiting the use of the RTVCLSRC command.

you can recreate source code from an object

Appendix A THE PROGRAMMER'S MENU

Programmer Menu

This is displayed if you sign on with QPGMR's password, or if you call a supplied program named QPGMMENU.

```
                              PROGRAMMER MENU
Select one of the following:
  1. Design/execute DFU app       (app), ,(options)
  2. Design/execute query app     (app), ,(options)
  3. Create object                object name, type, pgm for CMD, (text)
  4. Call program                 program name
  5. Execute command              command
  6. Submit job                   (job name), (command)
  7. Display submitted jobs
  8. Edit source                  (srcmbr), (type), (text)
  9. Design display format        (srcmbr)
 80. Display Menu                 (menu)
 90. Sign off                     (*NOLIST *LIST)
Types: BAS, BASP, BSCF, CBL, CL, CLP, CMD, CMNF, DFU, DSPF, LF, MXDF, PF, PRTF,
       QRY, RPG, RPT, TXT
Option:    Parm:            Type:       Parm 2:
Command:

Text:                                                  Log requests: *YES
Src file:           Src lib: *LIBL      Obj lib: SF38TEAM01 Jobd: SF38TEAM01
CF3-Command entry   CF4-Prompt (3,5 & 6 only)    CF6-DSPMSG
```

You will see there are eleven options on the menu, and with all of them, except option 7, an indication of what information is to be supplied when you select a particular option.

Below the menu is a list of object types. You must select one whenever this information is required to be entered in the input field 'Type'.

The input area covers five lines. You can complete the bottom line first - leave as they are the 'Src File' and 'Src Lib' fields but key your SF38TEAMxx in the fields 'Obj Lib' and 'Jobd'.
These four fields, plus 'Log Requests', can be changed (or protected from User change) via DSPPGMMNU - perhaps build into an 'Initial Program'.

Options 1 and 2

These two options take the place of the commands DSNDFUAPP and DSNQRYAPP which may be entered via the Command Entry Display, and initiate the Data File Utility and Query application design commands.

Optionally, you may enter the name of the application in the 'Parm' field and options, selected from the DFU or Query menus, in the 'Parm 2' field. For example, if you enter '2 3', this will take you from Option 2 on the first menu to Option 3 on the menu called by Option 2 on the first.

```
                              PROGRAMMER MENU
Select one of the following:
  1. Design/execute DFU app       (app), ,(options)
  2. Design/execute query app     (app), ,(options)
  3. Create object                object name, type, pgm for CMD, (text)
  4. Call program                 program name
  5. Execute command              command
  6. Submit job                   (job name), (command)
  7. Display submitted jobs
  8. Edit source                  (srcmbr), (type), (text)
  9. Design display format        (srcmbr)
 80. Display Menu                 (menu)
 90. Sign off                     (*NOLIST *LIST)
Types: BAS, BASP, BSCF, CBL, CL, CLP, CMD, CMNF, DFU, DSPF, LF, MXDF, PF, PRTF,
       QRY, RPG, RPT, TXT
Option: 2  Parm: SF38QRY    Type:       Parm 2:
Command:

Text:                                                  Log requests: *YES
Src file:           Src lib: *LIBL      Obj lib: SF38TEAM01 Jobd: SF38TEAM01
CF3-Command entry   CF4-Prompt (3,5 & 6 only)    CF6-DSPMSG
```

Option 3

Use this option to submit as a batch job the creation of an allowed object (NB - a SOURCE Physical File is not allowed). If the object already exists in your library, a message will invite the use of CMD 11 to delete it.

In this example, CUSEAR (a program) will be compiled. As "RPG" was specified in the 'Type' field, the system will submit the CRTRPGPGM command to be executed via your batch job queue and as no source file was specified, QRPGSRC will be assumed. CUSEAR will be created in the library named in the 'Obj lib' field.

```
                              PROGRAMMER MENU
Select one of the following:
  1. Design/execute DFU app       (app), ,(options)
  2. Design/execute query app     (app), ,(options)
  3. Create object                object name, type, pgm for CMD, (text)
  4. Call program                 program name
  5. Execute command              command
  6. Submit job                   (job name), (command)
  7. Display submitted jobs
  8. Edit source                  (srcmbr), (type), (text)
  9. Design display format        (srcmbr)
 80. Display Menu                 (menu)
 90. Sign off                     (*NOLIST *LIST)
Types: BAS, BASP, BSCF, CBL, CL, CLP, CMD, CMNF, DFU, DSPF, LF, MXDF, PF, PRTF,
       QRY, RPG, RPT, TXT
Option: 3  Parm: CUSEAR      Type: RPG  Parm 2:
Command:

Text: Customer Search program - Order Entry              Log requests: *YES
Src file:            Src lib: *LIBL      Obj lib: SF38TEAM01 Jobd: SF38TEAM01
CF3-Command entry    CF4-Prompt (3,5 & 6 only)    CF6-DSPMSG
```

A message will tell you that the job has been submitted. When the compilation is complete, a message will be sent to your workstation message queue - use CMD 6 to display it, then CMD 6 (or 8) to remove it. If it was 'bad news', Option 7 can be used to start Problem Determination!

As an alternative to the 'create' being submitted directly, a program named on the EXITPGM parameter of DSPPGMMNU could be called whenever Option 3 is chosen. (A DLTOPT parameter is also available.)

Option 4

This option allows you to call a program.

Notice, in the example, that it is not necessary to delete any of the input fields which are not required by a particular option.
The exception, in this case, may be if you have no CUSINQ program in your library, in which case you should change 'Obj lib' to "*LIBL" or "SF38LIB".

```
                              PROGRAMMER MENU
Select one of the following:
  1. Design/execute DFU app      (app), ,(options)
  2. Design/execute query app    (app), ,(options)
  3. Create object               object name, type, pgm for CMD, (text)
  4. Call program                program name
  5. Execute command             command
  6. Submit job                  (job name), (command)
  7. Display submitted jobs
  8. Edit source                 (srcmbr), (type), (text)
  9. Design display format       (srcmbr)
 80. Display Menu                (menu)
 90. Sign off                    (*NOLIST *LIST)
Types: BAS, BASP, BSCF, CBL, CL, CLP, CMD, CMNF, DFU, DSPF, LF, MXDF, PF, PRTF,
       QRY, RPG, RPT, TXT
Option: 4  Parm: CUSINQ      Type: RPG  Parm 2:
Command: dspjobd sf38team01

Text: Customer Search program - Order Entry              Log requests: *YES
Src file:           Src lib: *LIBL      Obj lib: SF38TEAM01 Jobd: SF38TEAM01
CF3-Command entry   CF4-Prompt (3,5 & 6 only)   CF6-DSPMSG
```

Option 5

This allows you to execute any command, the command being entered following 'Command'. With this option (and also options 3 and 6) you may use Cmd key 4 to request a prompt for the command.

```
                              PROGRAMMER MENU
Select one of the following:
  1. Design/execute DFU app       (app), ,(options)
  2. Design/execute query app     (app), ,(options)
  3. Create object                object name, type, pgm for CMD, (text)
  4. Call program                 program name
  5. Execute command              command
  6. Submit job                   (job name), (command)
  7. Display submitted jobs
  8. Edit source                  (srcmbr), (type), (text)
  9. Design display format        (srcmbr)
 80. Display Menu                 (menu)
 90. Sign off                     (*NOLIST *LIST)
Types: BAS, BASP, BSCF, CBL, CL, CLP, CMD, CMNF, DFU, DSPF, LF, MXDF, PF, PRTF,
       QRY, RPG, RPT, TXT
Option: 5  Parm:              Type:       Parm 2:
Command: dspjobd sf38team01

                                                              Log requests: *YES
Text:
Src file:             Src lib: *LIBL      Obj lib: SF38TEAM01 Jobd: SF38TEAM01
CF3-Command entry     CF4-Prompt (3,5 & 6 only)     CF6-DSPMSG
```

Option 6

Use this option to submit a job, that is not an object creation, to a batch job queue.

```
                              PROGRAMMER MENU
Select one of the following:
  1. Design/execute DFU app       (app), ,(options)
  2. Design/execute query app     (app), ,(options)
  3. Create object                object name, type, pgm for CMD, (text)
  4. Call program                 program name
  5. Execute command              command
  6. Submit job                   (job name), (command)
  7. Display submitted jobs
  8. Edit source                  (srcmbr), (type), (text)
  9. Design display format        (srcmbr)
 80. Display Menu                 (menu)
 90. Sign off                     (*NOLIST *LIST)
Types: BAS, BASP, BSCF, CBL, CL, CLP, CMD, CMNF, DFU, DSPF, LF, MXDF, PF, PRTF,
       QRY, RPG, RPT, TXT
Option: 6  Parm: TEAM1       Type:       Parm 2:       ,
Command: cpyf custl1 *list increl((*if balanc *gt 5500))

Text: Customer Search program - Order Entry              Log requests: *YES
Src file:            Src lib: *LIBL      Obj lib: SF38TEAM01 Jobd: SF38TEAM01
CF3-Command entry    CF4-Prompt (3,5 & 6 only)    CF6-DSPMSG
```

Option 7

Entering this option will display your submitted jobs which have not been cleared from the system.

```
17/11/83  12:02:50          SUBMITTED JOBS - *JOB
  JOB NAME    USER         NBR    TYPE    STATUS
  TEAM1       S038TEAM03 014142 BATCH   JOBQ     HELD
  TEAM3       S038TEAM03 014143 BATCH   JOBQ     HELD



1-DSPJOB    2-Spl files    4-HLDJOB    6-RLSJOB    9-CNLJOB    CF5-Redisplay
```

Once you see the display, you may enter one of the options shown against any job. Entering "2" shows you the printer files associated with the job.You may then display, release or cancel, just as you would from the DSPOUTQ command's output queue display.

```
16/11/83 10:45:12              SPOOLED OUTPUT FILES
Job:    BOBS           User: QSECOFR        Nbr: 013553
  FILE         NBR   OUTQ        LIBRARY    PTY  RCD/PAG  STATUS   CPY FORMTYPE
  QSYSPRT      0001  QPRINT2     QGPL         5       2P    HLD      1 *STD
  QPJOBLOG.    0002  QPRINT2     QGPL         5       1P    HLD      1 *STD



1-DSPSPLF   2-DSPSPLFA   4-HLDSPLF   6-RLSSPLF   9-CNLSPLF
CF5-Redisplay
```

If you cancel the files associated with the job after it has completed, you will be informed that the job is no longer in the system.

Option 8

This option initiates the Source Entry Utility via the EDTSRC command.

The option number and the "type" of source member are required entries, but you may optionally enter the source member name and some text.

If you do not enter a source member name, SEU will display a list of the source members in the source file from which you may make a selection, or enter the name of the new member you wish to create.

```
                              PROGRAMMER MENU
Select one of the following:
  1. Design/execute DFU app       (app), ,(options)
  2. Design/execute query app     (app), ,(options)
  3. Create object                object name, type, pgm for CMD, (text)
  4. Call program                 program name
  5. Execute command              command
  6. Submit job                   (job name), (command)
  7. Display submitted jobs
  8. Edit source                  (srcmbr), (type), (text)
  9. Design display format        (srcmbr)
 80. Display Menu                 (menu)
 90. Sign off                     (*NOLIST *LIST)
Types: BAS, BASF, BSCF, CBL, CL, CLP, CMD, CMNF, DFU, DSPF, LF, MXDF, PF, PRTF,
       QRY, RPG, RPT, TXT
Option: 8  Parm:              Type: CLP  Parm 2:
Command:

Text:                                                    Log requests: *YES
Src file:           Src lib: *LIBL      Obj lib: SF38TEAM01 Jobd: SF38TEAM01
CF3-Command entry   CF4-Prompt (3,5 & 6 only)    CF6-DSPMSG
```

Option 9

This option initiates Screen Design Aid (SDA) via the DSNFMT command.
You may optionally enter the name of a Source Member against 'Parm'.

```
                              PROGRAMMER MENU
Select one of the following:
  1. Design/execute DFU app       (app), ,(options)
  2. Design/execute query app     (app), ,(options)
  3. Create object                object name, type, pgm for CMD, (text)
  4. Call program                 program name
  5. Execute command              command
  6. Submit job                   (job name), (command)
  7. Display submitted jobs
  8. Edit source                  (srcmbr), (type), (text)
  9. Design display format        (srcmbr)
 80. Display Menu                 (menu)
 90. Sign off                     (*NOLIST *LIST)
Types: BAS, BASP, BSCF, CBL, CL, CLP, CMD, CMNF, DFU, DSPF, LF, MXDF, PF, PRTF,
       QRY, RPG, RPT, TXT
Option: 9  Parm: CUSINQDF    Type:       Parm 2:
Command:

Text:                                                  Log requests: *YES
Src file:             Src lib: *LIBL       Obj lib: SF38TEAM01 Jobd: SF38TEAM01
CF3-Command entry     CF4-Prompt (3,5 & 6 only)     CF6-DSPMSG
```

Option 80

Use this option without any parameter to obtain a Command Grouping Menu. However, in the 'Parm' field you can enter the name of a menu available via the Command Grouping Menu to get to it directly.

```
                              PROGRAMMER MENU
Select one of the following:
  1. Design/execute DFU app       (app), ,(options)
  2. Design/execute query app     (app), ,(options)
  3. Create object                object name, type, pgm for CMD, (text)
  4. Call program                 program name
  5. Execute command              command
  6. Submit job                   (job name), (command)
  7. Display submitted jobs
  8. Edit source                  (srcmbr), (type), (text)
  9. Design display format        (srcmbr)
 80. Display Menu                 (menu)
 90. Sign off                     (*NOLIST *LIST)
Types: BAS, BASP, BSCF, CBL, CL, CLP, CMD, CMNF, DFU, DSPF, LF, MXDF, PF, PRTF,
       QRY, RPG, RPT, TXT
Option: 80 Parm: CRT         Type:       Parm 2:
Command:

Text:                                                  Log requests: *YES
Src file:             Src lib: *LIBL       Obj lib: SF38TEAM01 Jobd: SF38TEAM01
CF3-Command entry     CF4-Prompt (3,5 & 6 only)     CF6-DSPMSG
```

Option 90

Use this option to SIGNOFF, optionally specifying "*LIST" to cause your job log to be sent to the output queue.
*NOLIST is the default.

```
                              PROGRAMMER MENU
Select one of the following:
  1. Design/execute DFU app      (app), ,(options)
  2. Design/execute query app    (app), ,(options)
  3. Create object               object name, type, pgm for CMD, (text)
  4. Call program                program name
  5. Execute command             command
  6. Submit job                  (job name), (command)
  7. Display submitted jobs
  8. Edit source                 (srcmbr), (type), (text)
  9. Design display format       (srcmbr)
 80. Display Menu                (menu)
 90. Sign off                    (*NOLIST *LIST)
Types: BAS, BASF, BSCF, CBL, CL, CLP, CMD, CMNF, DFU, DSPF, LF, MXDF, PF, PRTF,
       QRY, RPG, RPT, TXT
Option: 90 Parm: *LIST       Type:       Parm 2:
Command:

Text:                                                  Log requests: *YES
Src file:            Src lib: *LIBL      Obj lib: SF38TEAM01 Jobd: SF38TEAM01
CF3-Command entry    CF4-Prompt (3,5 & 6 only)    CF6-DSPMSG
```

Conclusion

At any time, you may change to the Command Entry Display by pressing Cmd key 3, and then return to the Programmer Menu by using Cmd key 1.
To return to the Command Entry Display, if you called the Programmer Menu program, use Cmd key 1.

Appendix B THE SCREEN DESIGN AID

Screen Design Aid (SDA)

SDA is an interactive utility which can be used to design and maintain :

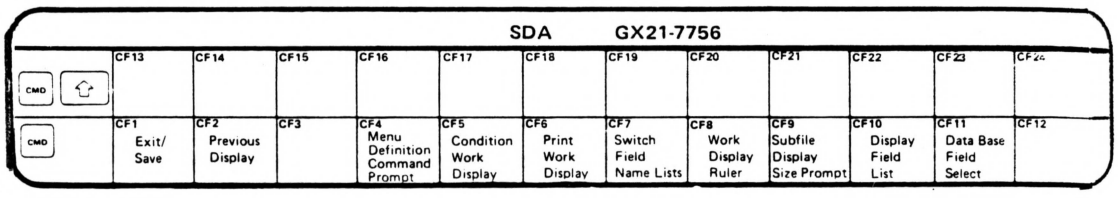
- Display record formats, including subfiles
- Application menus

It can be invoked from the Programmer Menu, optionally specifying the source member name (see Appendix A).

Alternatively, you can enter the DSNFMT command.
(Parameters can be overridden from within SDA)

```
                        Design Format (DSNFMT) Prompt
Enter the following:
  Source file name:                  SRCFILE     P   QDDSSRC
    Library name:                                      *LIBL
  Source member name:                SRCMBR      P   *SELECT
  Created file library name:         OBJLIB      P   QGPL
  Job description name:              JOBD            QBATCH
    Library name:                                      *LIBL
```

There is an array of command keys available, as shown on the SDA template.



Extensive HELP text is also available.

```
HELP                          SDA HELP TEXT MENU                    Next Option: __
OPTION                                                              CF1 to exit help
  1.  Introduction
  2.  Primary SDA Options
  3.    1-  DESIGN RECORD FORMAT MENU
  4.                Member List
  5.                Record Format List
  6.          1-    File Definition
  7.          2-    Record Format Definition
  8.          3-    Field Definition
  9.                  Work Display
 10.                    Work Display Conditioning          -CF5
 11.                    Field List                         -CF10
 12.                    Data Base File Displays and List   -CF11
 13.                    Extended Field Definition          -* before field
 14.          4-    Subfile Control Record Definition
 15.          5-    Subfile Record Format Definition
 16.          6-    Save and create options
 17.    2-  DESIGNING A MENU
 18.          Initial Menu Definition
 19.          Menu Definition
 20.    3-  TESTING AN EXISTING RECORD FORMAT
```

SDA uses work files, just like SEU, and does not affect your source files until you perform the "save" at end of job.

There is the option to automatically delete old versions before saving the new.

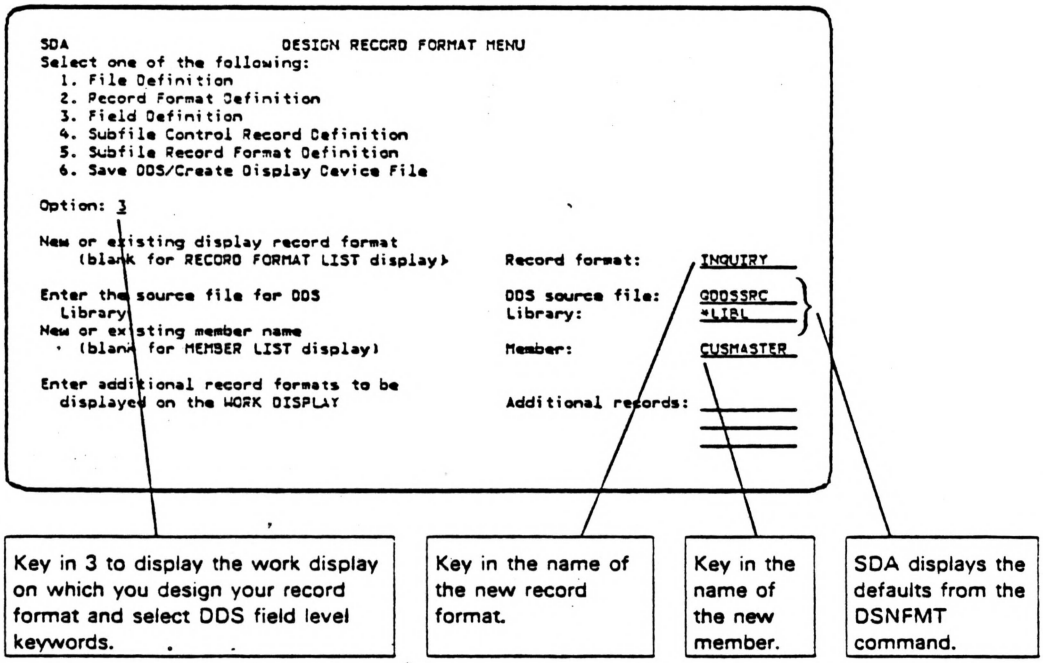
Compiles may be submitted to the batch job queue or run interactively.

The initial menu offers three options :

```
SDA                                OPTION MENU

Select one of the following:
  1. Design display record formats
  2. Design a menu
  3. Test an existing display record format

Option: _

 .                    .

 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

Within SDA:
  Press HELP key to display help text for the current display.
  Press CF1 key to exit any option and allow saving the changes.
  Press CF2 key to back up to the previous display in a series.
  Press ENTER/REC ADV to advance to the next display.
```

Designing a display format

The steps involved in designing display formats are centred around this menu.



The steps are as follows:

1. Define any File Level attributes.
   - General file keywords. (eg. DSPSIZ, REF, PRINT)
   - Command keys and indicators (HELP, CA01 etc.)

   CMD 2 returns you to the menu.

2. Define any Record Level details, initially entering the name of the format.

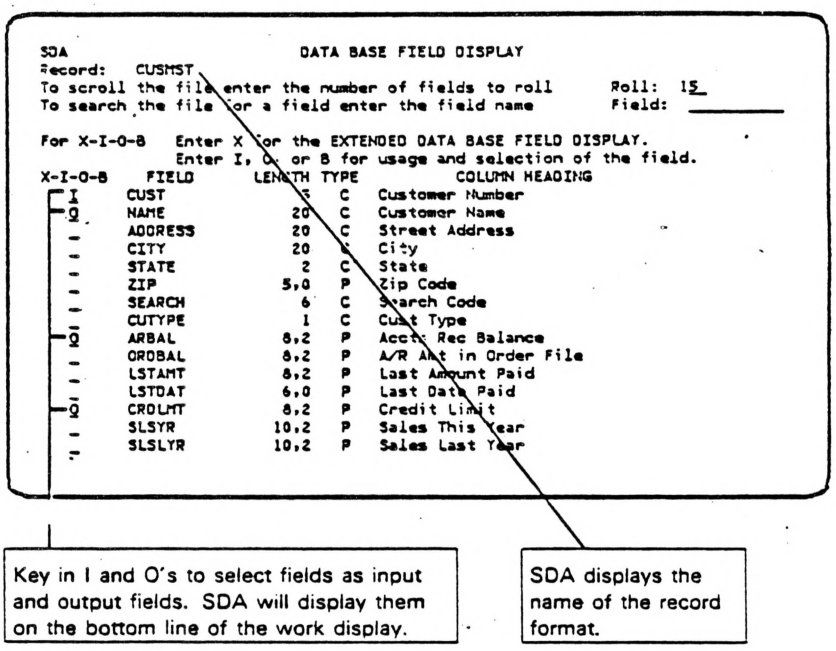
   Record level details are entered via a series of six screen displays.
   - Command keys and indicators.
   - Output keywords. (eg. BLINK)
   - Input keywords. (eg. UNLOCK)
   - General record keywords. (eg. CHGINPDFT)
   - Overlay keywords.
   - Routing feedback.(for writing information to the Data Management feedback area)

   CMD 2 returns you to the menu.

3. Define the fields that are to appear on the display format via the Work Display

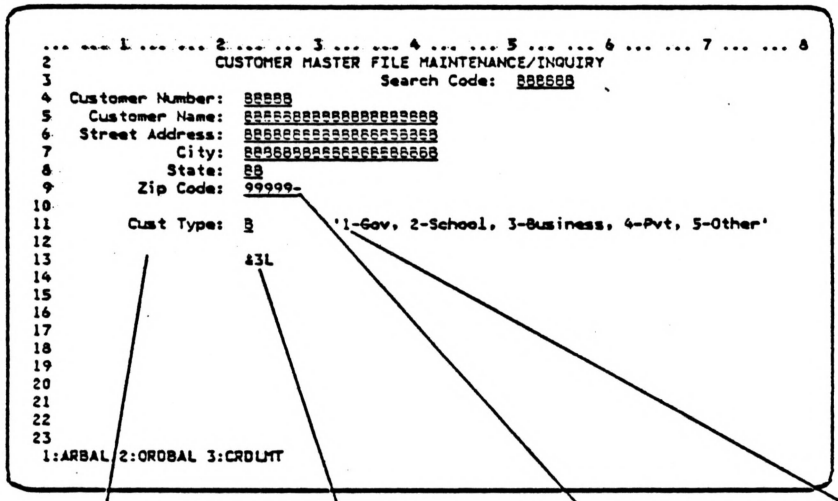
CMD 8 causes the SDA Work Display ruler to be displayed. To ensure correct positioning of the ruler, position the cursor at "line 1/position 1" (although any line no. will do) of the screen before pressing CMD 8.
Pressing CMD 8 again will remove the ruler.

CMD 11 will give access to any Data Base fields which you need to reference.



Having selected the fields that you require and specified their use, whether Input Output or Both, press CMD 2 (twice) to get back to the Work Display.

4. Now use the fields you selected and which are displayed to the bottom of the Work Display.
They are positioned by means of an ampersand(&) followed by the reference no. of the field. Optionally you may specify where the column headings should be displayed.



SDA displays the fields that you specified.

Key in &3L on line 13, position 23 to place the field CRDLMT on the work display with its column heading to the left.

The default for output numeric fields is signed numeric.

Key in this constant on line 11, position 32. These are the valid customer types.

Additional fields may be added to the format as necessary, either as constants or as extra variables. Any variables must be defined at the time of entry to the format.

Note : See Chapter 7 of the SDA manual for details of the required entries.
The additional fields are given a default name (eg. FLD001). CMD 10 will give you the Field List Display which allows you to change the field names, the sequence in which fields appear on the format and to add hidden fields.

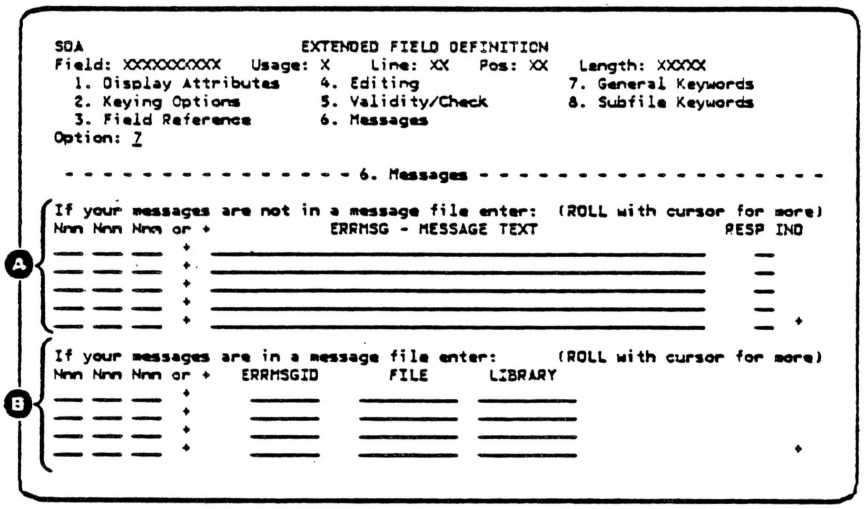
Fields, or blocks of fields may be moved left or right, or moved/copied to other areas on the format.

5. Display attributes may be set by entering the relevant code immediately prior to the field displayed on the Work Display.

eg.

B or b	-	Blink	C or c	-	Column separators
H or h	-	Highlight	R or r	-	Reverse image
U or u	-	Underline	N or n	-	Nondisplay

If an asterisk(*) is the code entered, then extended field definitions may be specified via a series of prompt screens. These prompts allow overrides to standard defaults, display attributes, editing, and assignment of error messages.



6. Finally, when you have completed the design of the display format, CMD 2 returns you to the Design Record Format menu. Option 6 prompts you for Save DDS/Create display device file. One option allows the prompt of the CRTDSPF command so that you may override defaults.

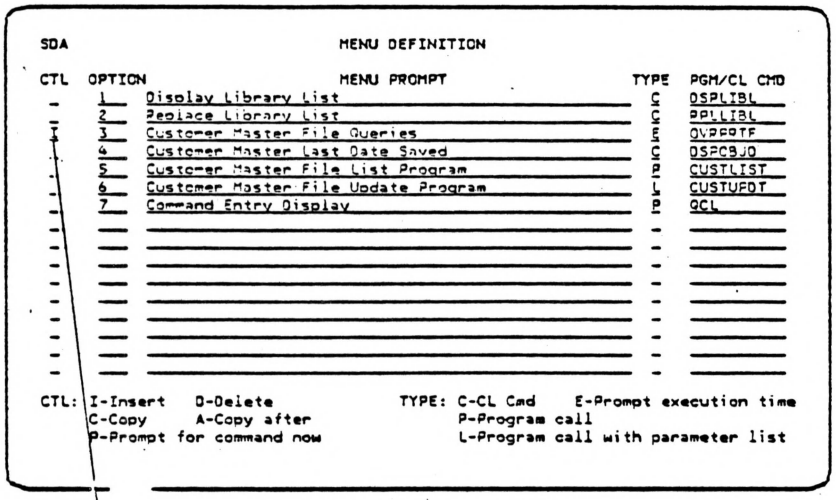
Testing a Display Device File

Once Display files have been created, they may be tested via SDA. Data may be entered for both Output and Input fields and indicators set. This facility is useful for checking that the appropriate Error Message is displayed for a given setting of indicators.

Designing and Creating a Menu

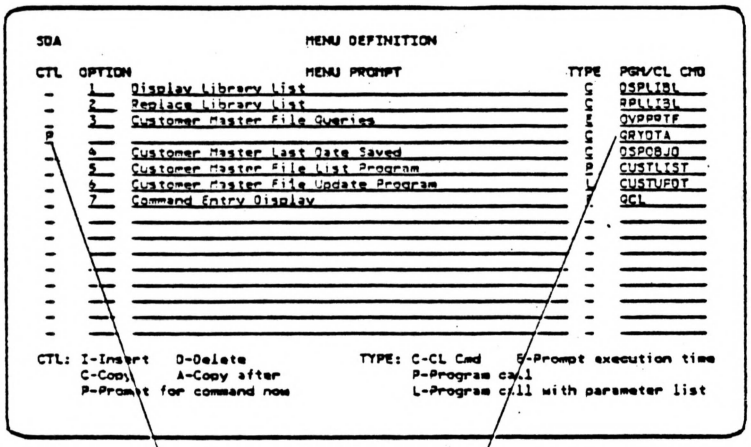
The major steps are as follows :

1. Define the options and prompt text for the menu.
2. Specify the commands to be associated with each prompt.



Key in the control character I to insert a line.

3. Using the Control characters, prompt for the command parameters or copy/insert commands.



Key in the control character P to specify parameters for this command.

QRYDTA is highlighted because parameters have not yet been specified for this command.

4. CMD 1 to move on to the Save DDS/Create Display Device File screen prompts.
   SDA will also initiate compilation of the Control Language program.

Appendix C

SF38 FILE LOAD / DFU EXERCISE

RUN INSTRUCTIONS

```
                              PROGRAMMER MENU
Select one of the following:
  1. Design/execute DFU app      (app), ,(options)
  2. Design/execute query app    (app), ,(options)
  3. Create object               object name, type, pgm for CMD, (text)
  4. Call program                program name
  5. Execute command             command
  6. Submit job                  (job name), (command)
  7. Display submitted jobs
  8. Edit source                 (srcmbr), (type), (text)
  9. Design display format       (srcmbr)
 80. Display Menu                (menu)
 90. Sign off                    (*NOLIST *LIST)
Types: BAS, BASP, BSCF, CBL, CL, CLP, CMD, CMNF, DFU, DSPF, LF, MXDF, PF, PRTF,
       QRY, RPG, RPT, TXT
Option: 1  Parm: CUSMNT     Type:      Parm 2:
Command:
Text:                                                Log requests: *YES
Src file:            Src lib: *LIBL      Obj lib: SF38TEAM01 Jobd: SF38TEAM01
CF3-Command entry    CF4-Prompt (3,5 & 6 only)    CF6-DSPMSG
```

> Take option 1 from the Programmer Menu. Enter a meaningful name for your DFU application in the 'Parm' field. Remember to fill in the Object Library and Job Description fields.

DFU MENU

Select one of the following:
1. Create, change, or delete an application
2. Execute an application
3. Manage existing applications

Option: 1

Press HELP for instructions. Press CF1 to exit.

> Take option 1.

DFU CREATE/CHANGE/DELETE MENU

Select one of the following and enter values below:
1. Display information about an application
2. Create a new application
3. Change an existing application
4. Delete an existing application
5. Display status of applications submitted

Option: 2

Application name: CUSMNT (*-Application selection list)
Library name: SF38TEAM01

HELP-Help CF2-Previous display

> Take option 2.

```
                              DFU CREATE PROMPT
Application name:     CUSMNT              Library:     SF38TEAM01

Enter information for new application:
  Description:           Maintain team 01 customer file
  File name:             CUSTLT01         (*-File selection list)
    Library name:          SF38TEAM01

HELP-Help   CF2-Previous display   CF3-File information
```

> Enter a meaningful description for your DFU application, and the name and library of your customer logical file. Press ENTER.

```
                          APPLICATION CONTROL PROMPT
Application name:     CUSMNT                        Library:     SF38TEAM01
File name:            CUSTLT01                      Library:     SF38TEAM01

Place an X next to display size:
  Primary display size:              X  24x80         12x80         16x64

Take defaults after field selection:                  N  (Y-Yes   N-No)


HELP-Help   CF2-Previous display
```

> Press ENTER.

```
                RECORD FORMAT REVIEW PROMPT FOR FILE CUSTLT01
For each record format to be used, enter R to review the fields, A to select all
fields, or E to enter the fields later:
     RECORD FMT  DESCRIPTION
  A  CUSTPR      Customer phys. Format
```

> Enter 'A' to select all fields in the record format.

ENTRY FORMAT DEFINITION PROMPT

```
For record format name CUSTPR     enter:
  Entry format identifier:        PM
  Entry format description:       Customer phys. Format
  Display one field per line:     *YES
  Display multiple records:       *NO
  Redisplay changes:              *NO
  Chain to entry format ID:
```

> Enter 'PM' for the Entry format identifier. Change 'Display one field per line' to *YES.

BASIC FIELD DEFINITION PROMPT FOR RECORD CUSTPR

Define format ID PM - Customer phys. Format

FIELD NAME	ORDER	INPUT	DISPLAY	VERIFY	XDEF	XVAL
CUST	1.0	X	X			D
NAME	2.0	X	X			
CUSAD1	3.0	X	X			
CUSAD2	4.0	X	X			
CUSAD3	5.0	X	X			X
SVIA	6.0	X	X			D
PAYM	7.0	X	X			D
DISC	8.0	X	X			D
CRLIM	9.0	X	X			
BALANC	10.0	X	X			
YRTOTL	11.0	X	X			
LORDER	12.0	X	X			
UPDATC	13.0	X	X			
NAMARG	14.0	X	X			

+

> Key X for CUSAD3 under XVAL – to allow removal of CHECK(MF).
> Press ENTER.

VALIDITY CHECK PROMPT FOR FIELD CUSAD3

Select validity checks by inserting X, or cancel with blanks:

Mandatory entry	Mandatory fill
MOD10 check	MOD11 check
Name check	Allow blanks
Relational operator:	List of values:

> Press ENTER. This cancels the field reference file edit keywords for the field.

ENTRY FORMAT DEFINITION PROMPT

```
For record format name CUSTPR     enter:
  Entry format identifier:
  Entry format description:      Customer phys. Format
  Display one field per line:    *NO
  Display multiple records:      *NO
  Redisplay changes:             *NO
  Chain to entry format ID:
```

> Press ENTER.

AUDIT CONTROL PROMPT

```
Enter the following:
  Add/delete records allowed:    *YES
  Change records allowed:        *YES
  Key changes allowed:           *YES
  Print additions:               *NO
  Print changes:                 *NO
  Print deletions:               *NO
  Data error option
   (*NOTIFY *DISPLAY *CHANGE):   *NOTIFY
```

> Press ENTER.

EXIT APPLICATION DEFINITION MENU

Select one of the following:
1. Restart definition
2. Modify definition
3. Delete definition
4. Save definition
5. Create application interactively
6. Create application in batch

Option: 6

Display fmt 'PM' rcd fmt CUSTPR too large for 12x80 screen.

> Press ENTER - Accept the default option 6.

APPLICATION CREATION PROMPT

```
Enter the following:
  Application name:                CUSMNT
    Library name:                    SF38TEAM01
  Public authority:                1    (1-Normal  2-None  3-All)
  Adopt owner's user profile:      N    (Y-Yes  N-No)
  Source listing:                  N    (Y-Yes  N-No)
  Dump internal data areas:        N    (Y-Yes  N-No)
  Generated code listing:          N    (Y-Yes  N-No)

  Job description name:            SF38TEAM01 (*-Job description list)
    Library name:                    SF38TEAM01
```

Display fmt 'PM' rcd fmt CUSTPR too large for 12x80 screen.

> Press ENTER.