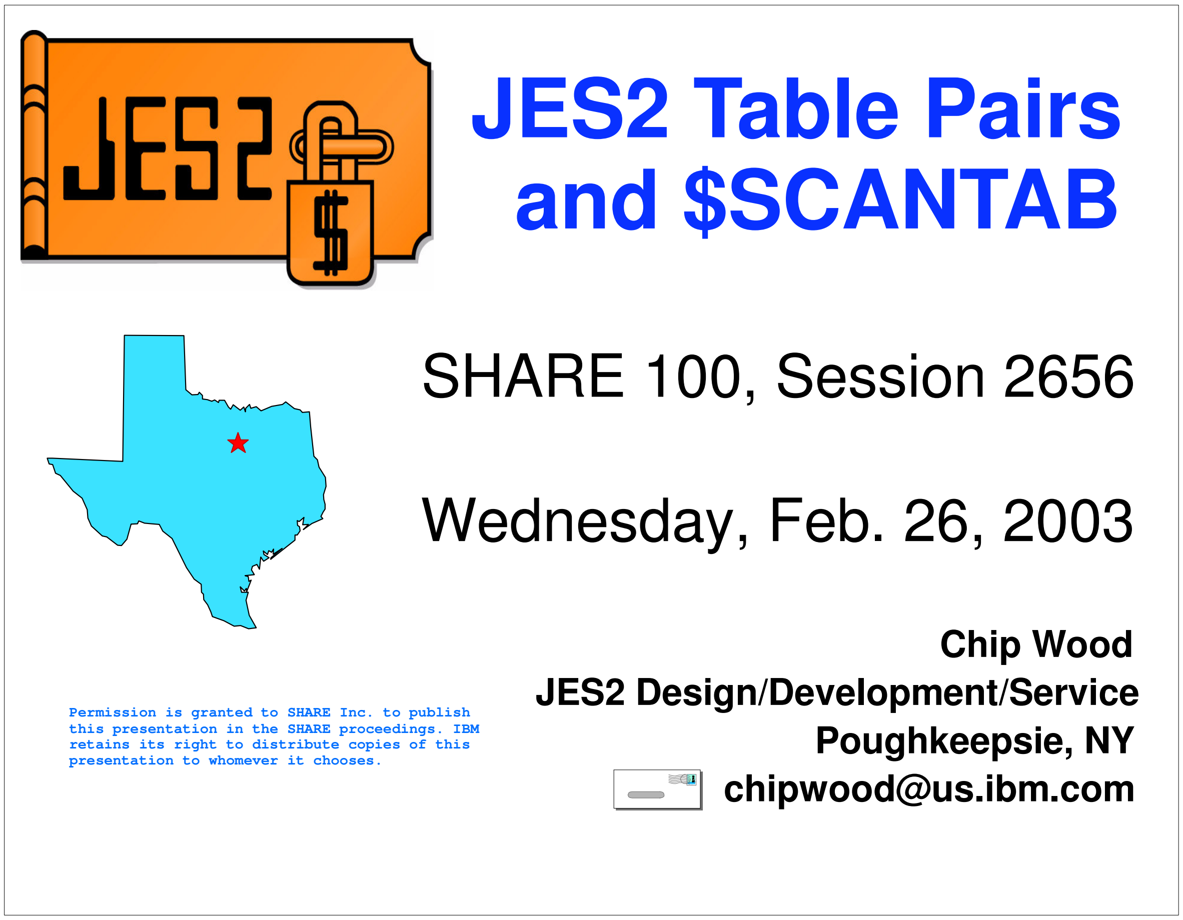


JES2 Table Pairs and $SCANTAB

SHARE 100, Session 2656

Wednesday, Feb. 26, 2003

Chip Wood
JES2 Design/Development/Service
Poughkeepsie, NY
chipwood@us.ibm.com

Agenda

- What are table pairs?
- Extending JES2 commands and initialization statements with $SCANTAB
  - Example tables
  - Extending job commands
  - Extending output commands
- Extending the $JQE using $BERTTAB

Disclaimers

- From *JES2 Installation Exits*, page 1:

Caution!

Defining exits and writing installation exit routines is intended to be accomplished by experienced system programmers; the reader is assumed to have knowledge of JES2.

If you want to customize JES2, IBM recommends that you use JES2 installation exits to accomplish the task. **IBM does not recommend or support alteration of JES2 source code.**

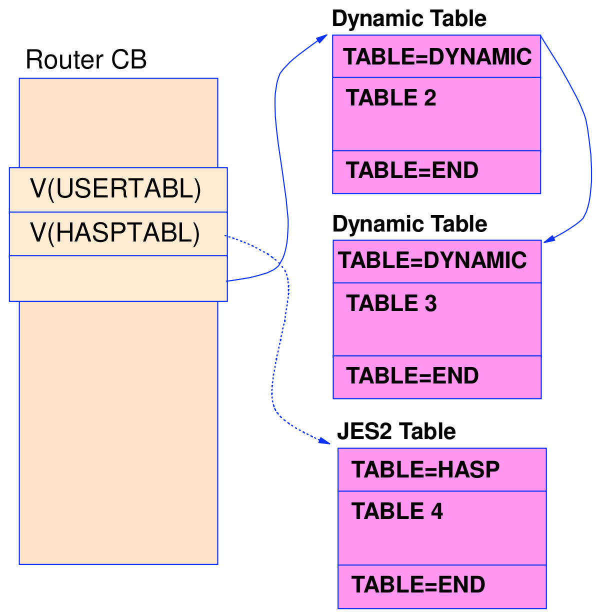
- From *JES2 Diagnosis*:

CAUTION: IBM does not recommend or support modifications to JES2 source code. If you assume the risk of modifying JES2, then also assure that your modifications do not impact JES2 serviceability using IPCS. Otherwise, LEVEL2 may not be able to read JES2 dumps for problems unrelated to the modifications.

What are table pairs?

- Provide a mechanism to:
  - Add installation processing/function
  - Modify JES2 processing/function
- Involves:
  - Installation written tables/routines
  - Less detailed knowledge of JES2 code and control blocks
- **DOES NOT** replace need for exits
- Main exploiter - $SCANTAB
  - Also $PCETAB, $DTETAB, $DCTTAB, $WSTAB, $TIDTAB, $BERTTAB

Extended table pairs



Dynamic table

- Information supplied by installation
- Multiple dynamic tables may be associated with one table pair
- Linked to table pair automatically when loaded (via LOAD init statement)
- Overrides JES2 table
- Overridden by Installation table

Table pair

- Consists of three pointers
- Defined by $PAIR macro

Extended table pairs

- $PCETAB TABLE=DYNAMIC
- $DCTTAB TABLE=DYNAMIC
- $DTETAB TABLE=DYNAMIC
- $TIDTAB TABLE=DYNAMIC
- $BERTTAB TABLE=DYNAMIC
- $SCANTAB TABLE=(DYNAMIC, pair-name, pair-name, ...)
- $WSTAB TABLE=(DYNAMIC, pair-name, pair-name, ...)

Extending JES2 commands and initialization statements with

$SCANTABs

What are $SCANTABs

- $SCANTABs:
  - Define initialization statements and keywords
  - Define commands and keywords
    - Syntax rules
    - Control block structures and fields to be displayed/modified/filtered
- Parsing command or statement handled automatically by $SCAN facility
- Data conversion handled automatically by $SCAN facility
  - EBCDIC to numeric, hexadecimal, or flag settings (SET, FILTER)
  - Numeric, hexadecimal, or flag settings to EBCDIC (DISPLAY)

Example 1 - USERDEF

- Define a new statement with the following syntax:

```
USERDEF   MYCHARS=cccccccc,
          MYFLAG=YES/NO,
          MYCOUNT=nnnnn
```

- Allow values to be set and displayed during initialization, and via **$D** and **$T** commands.

USERDEF table

- First, we need to define a table for the USERDEF statement (the highest level keyword).

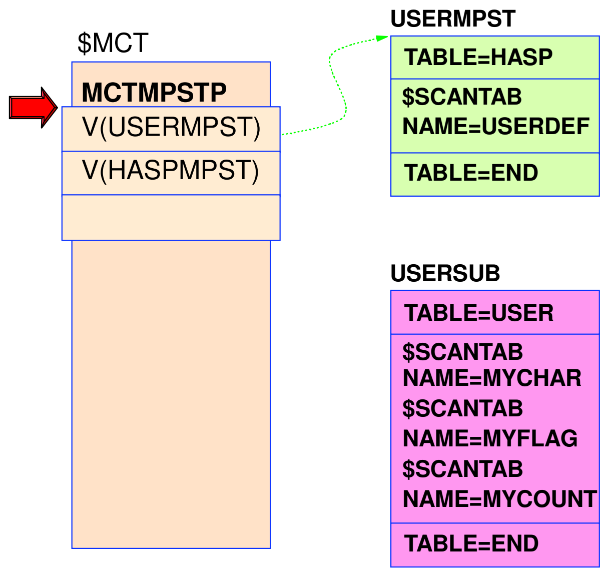
```
$SCANTAB   NAME=USERDEF,MINLEN=4,
           CB=UCT,CONV=SUBSCAN,
           SCANTAB=(USERSUB,ADDR)
           CALLERS=($SCIRPL,$SCIRPLC,$SCSCMDS,$SCDCMDS)
```

  - **NAME=** - defines the parameter name
  - **MINLEN=** - specifies minimum length to be specified (USER)
  - **CB=** - Control block associated with keyword
  - **CONV=** - type of conversion to be done
    - **SUBSCAN** - indicates lower level keywords will be defined
  - **SCANTAB=** - Where to find the subscan parameters
  - **CALLERS=** - What command and statement types are allowed
    - Required on highest level $SCANTAB

CONV=

- Specifies conversion/verification to be done on input values or fields:
  - For **SET** requests, validates input, converts and stores value
  - For **DISPLAY** requests, converts and displays value
  - For **FILTER** requests, converts and compares value
- Valid values
  - **CONV=CHAR** - character input
    - Optionally can specify valid character set
  - **CONV=NUM** - Numeric input (stored in binary)
  - **CONV=HEX** - Hexadecimal input (stored in binary)
  - **CONV=FLAG** - Maps EBCDIC names to bits
  - **CONV=SUBSCAN** - Scan for subparameters

CONV=SUBSCAN



Commands and Init statements are all defined by tables pointed to by MCTMPSTP

- Tables for specific statements specify CONV=SUBSCAN to define subparameters
- SCANTAB= parameter points to table pair

To define new commands or init statements, associate USER or DYNAMIC tables with MCTMPSTP

Including the USERDEF table

- To include the table, associate the table with the main parameter statements

```
MYTABLE   $SCANTAB TABLE=(DYNAMIC,MCTMPSTP)
          $SCANTAB NAME=USERDEF,...
          $SCANTAB TABLE=END
```

- TABLE=(DYNAMIC,MCTMPSTP)
  - When the module containing the table is LOADed from the init deck, the table is automatically associated with main parameter statements
- TABLE=END
  - Marks the end of tables in this group

Including the subscan tables

- To include the table, associate the table with the main parameter statements

```
MYTABLE   $SCANTAB TABLE=(DYNAMIC,MCTMPSTP)
          $SCANTAB NAME=USERDEF,...,
                   CONV=SUBSCAN,
                   SCANTAB=(USERSUB,ADDR)
          $SCANTAB TABLE=END

USERSUB   $PAIR  ,
          $SCANTAB NAME=MYCHAR,...
          $SCANTAB NAME=MYFLAG,...
          $SCANTAB NAME=MYCOUNT,...
          $SCANTAB TABLE=END
```

- SCANTAB= - points to subscan table pair
  - ADDR - indicates table pair is inline, not in MCT
- $PAIR - table pair and opening $SCANTAB

$PAIR Macro

- Use the **$PAIR** macro to define your own table pairs (**CONV=SUBSCAN**)
  - IBM-supplied table pairs in the $MCT
    - **CONV=SUBSCAN,SCANTAB=MCT*xxx*TP**

```
 MCTxxxTP   $PAIR  TABLE=SCAN
+MCTxxxTP   DS     0H
+MCTxxxTU   DC     V(USERxxxT)     User table
+MCTxxxTH   DC     V(HASPxxxT)     JES2 table
+MCTxxxTD   DC     A(*-*)          Dynamic tables
```

  - Installation-defined table pairs in the $UCT
    - **CONV=SUBSCAN,SCANTAB=(UCT*xxx*TP,UCT)**

```
UCTxxxTP   $PAIR  TABLE=SCAN,HASPENT=NONE
+UCTxxxTP   DS     0H
+UCTxxxTU   DC     V(USERxxxT)     User table
+           DC     A(*-*)          No JES2 table
+UCTxxxTD   DC     A(*-*)          Dynamic tables
```

$PAIR Macro

- In inline code
  - When extendability is not important (e.g. HASPMSG or installation code)
  - **CONV=SUBSCAN,SCANTAB=(MYTABLE,ADDR)**

```
 MYTABLE   $PAIR   ,
+MYTABLE    DS     0H
+           DC     A(MYTABLE_U)    User table
+           DC     A(0)            No JES2 table
+           DC     A(0)            No Dynamic tables
+MYTABLE_U $SCANTAB TABLE=USER

        . . . <insert tables here>

          $SCANTAB TABLE=END
```

Lower level $SCANTABs

- **MYCHAR** keyword specifies a character string, stored in the UCT

```
$SCANTAB   NAME=MYCHAR,MINLEN=4,
           CB=PARENT,DSECT=UCT,FIELD=UCTCHAR,
           CONV=CHAR,RANGE=(1,L'UCTCHAR)
```

- **CB=PARENT** - use control block passed from higher level $SCANTAB
- **FIELD=** - the field in that control block to be used
- **DSECT=** - the DSECT in which the field resides
- **CONV=CHAR** - store as character string
- **RANGE=** - minimum and maximum allowed lengths of that string

Lower level $SCANTABs

- **MYCOUNT** keyword specifies a numeric field, stored in the UCT

```
$SCANTAB  NAME=MYCOUNT,
          CB=PARENT,DSECT=UCT,FIELD=UCTNUM,
          CONV=NUM,RANGE=(1,9999)
```

- **CONV=NUM** - store as numeric value
- **RANGE=** - minimum and maximum allowed values
- **CALLERS=** - defaults to all callers
  - Limited by callers at higher levels

Lower level $SCANTABs

- **MYFLAG** keyword specifies a single bit to be manipulated, stored in the UCT

```
$SCANTAB   NAME=MYFLAG,CB=PARENT,DSECT=UCT,FIELD=UCTFLAG,
           CONV=FLAG,
           VALUE=(YES,X'80',X'FF',NO,0,X'FF'-X'80')
```

- **CONV=FLAG** - store as numeric value
- **VALUE=** - a set of triplets, defining values and bit manipulations
  - first element - a character string associated with the bit values
  - second element - 'on' mask
    - bits on in mask must be on in flag byte
  - third element - 'off' mask
    - bits off in mask must be off in flag byte

VALUE= keyword

- **VALUE=(YES,X'80',X'FF',NO,0,X'FF'-X'80')**
- On a set request
  - If **'YES'** (or 'Y') is specified, then:
    - all bits on in **X'80'** mask will be turned on (or)
    - all bits off in **X'FF'** mask will be turned off (and)
  - If **'NO'** (or 'N') is specified, then:
    - all bits on in **0** mask will be turned on (or)
    - all bits off in **X'FF'-X'80'** mask will be turned off (and)
- On a display request
  - If all bits that are on in **X'80'** are on, and all bits that are off in **X'FF'** are off, then **'YES'** will be displayed
  - If all bits that are on in **0** are on, and all bits that are off in **X'FF'-X'80'** are off, then **'NO'** will be displayed

USERDEF tables

```
MYTABLE  $SCANTAB TABLE=(DYNAMIC,MCTMPSTP)
         $SCANTAB NAME=USERDEF,MINLEN=4,
                  CB=UCT,CONV=SUBSCAN,
                  SCANTAB=(USERSUB,ADDR)
                  CALLERS=($SCIRPL,$SCIRPLC,
                  $SCSCMDS,$SCDCMDS)
         $SCANTAB TABLE=END

USERSUB  $PAIR ,
         $SCANTAB NAME=MYCHAR,MINLEN=4,
                  CB=PARENT,DSECT=UCT,FIELD=UCTCHAR,
                  CONV=CHAR,RANGE=(1,L'UCTCHAR)
         $SCANTAB NAME=MYFLAG,CB=PARENT,DSECT=UCT,
                  FIELD=UCTFLAG,CONV=FLAG,
                  VALUE=(YES,X'80',X'FF',
                  NO,0,X'FF'-X'80')
         $SCANTAB NAME=MYCOUNT,
                  CB=PARENT,DSECT=UCT,FIELD=UCTNUM,
                  CONV=NUM,RANGE=(1,9999)
         $SCANTAB TABLE=END
```

Example 2

- Add a new filter, MINUTES, to job and output commands, similar to the existing HOURS and DAYS filters
- Start with HOURS and DAYS $SCANTABs and modify as necessary

Existing tables

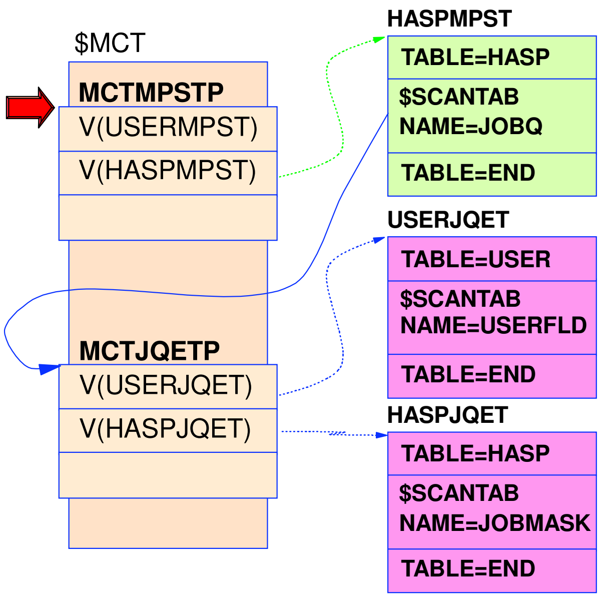
```
$SCANTAB NAME=HOURS,MINLEN=1,
         CB=PARENT,DSECT=JQE,
         FILTER=(YES,ALWAYS,GTLT),
         PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
         FIELD=JQXCRTME,CONV=(NUM,,60),
         CALLERS=($SCDCMDS,$SCSCMDS,$SCECMDS,$SCHCMDS,
         $SCRLCMD,$SCCCMDS,$SCPCMDS,$SCDOCMD),
         RANGE=(0,X'7FFFFFFF')

$SCANTAB NAME=DAYS,MINLEN=2,
         CB=PARENT,DSECT=JQE,
         FILTER=(YES,ALWAYS,GTLT),
         PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
         FIELD=JQXCRTME,CONV=(NUM,,60*24),
         CALLERS=($SCDCMDS,$SCSCMDS,$SCECMDS,$SCHCMDS,
         $SCRLCMD,$SCCCMDS,$SCPCMDS,$SCDOCMD),
         RANGE=(0,X'7FFFFFFF')
```

New table

- In HOURS/DAYS, **CONV=NUM** multiplier converts input value to minutes
  - **PREFOAGE** prescan routine converts input time to age in minutes and compares with input value (adjusted by multiplier)
    - Routine is called only for DISPLAYs and FILTERs
    - SET calls disallowed by other parameters
  - For MINUTES, use same table with no multiplier

```
$SCANTAB NAME=MINUTES,MINLEN=3,
       CB=PARENT,DSECT=JQE,
       FILTER=(YES,ALWAYS,GTLT),
       PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
       FIELD=JQXCRTME,CONV=NUM,
       CALLERS=($SCDCMDS,$SCSCMDS,$SCECMDS,$SCHCMDS,
       $SCRLCMD,$SCCCMDS,$SCPCMDS,$SCDOCMD),
       RANGE=(0,X'7FFFFFFF')
```

CONV=SUBSCAN



To define new keywords on job list commands, associate USER or DYNAMIC tables with MCTJQETP

CB, DSECT, FIELD

```
$SCANTAB NAME=MINUTES,MINLEN=3,
       CB=PARENT,DSECT=JQE,
       FILTER=(YES,ALWAYS,GTLT),
       PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
       FIELD=JQXCRTME,CONV=NUM,
       CALLERS=($SCDCMDS,$SCSCMDS,$SCECMDS,$SCHCMDS,
       $SCRLCMD,$SCCCMDS,$SCPCMDS,$SCDOCMD),
       RANGE=(0,X'7FFFFFFF')
```

- **CB=PARENT** indicates to use control block located at parent level of $SCAN
  - Read-mode JQA for display job commands
  - Update-mode JQA for set job commands

CALLERS

```
$SCANTAB NAME=MINUTES,MINLEN=3,
        CB=PARENT,DSECT=JQE,
        FILTER=(YES,ALWAYS,GTLT),
        PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
        FIELD=JQXCRTME,CONV=NUM,
        CALLERS=($SCDCMDS,$SCSCMDS,$SCECMDS,$SCHCMDS,
        $SCRLCMD,$SCCCMDS,$SCPCMDS,$SCDOCMD),
        RANGE=(0,X'7FFFFFFF')
```

- **CALLERS=** indicates which commands this table applies to:
  - **$SCDCMDS** - $D
  - **$SCSCMDS** - $T
  - **$SCECMDS** - $E
  - **$SCHCMDS** - $H
  - **$SCSTCMD** - $S
  - **$SCRLCMD** - $A
  - **$SCCCMDS** - $C
  - **$SCPCMDS** - $P
  - **$SCDOCMD** - Display after set

FILTERs

```
$SCANTAB NAME=MINUTES,MINLEN=3,
        CB=PARENT,DSECT=JQE,
        FILTER=(YES,ALWAYS,GTLT),
        PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
        FIELD=JQXCRTME,CONV=NUM,
        CALLERS=($SCDCMDS,$SCSCMDS,$SCECMDS,$SCHCMDS,
        $SCRLCMD,$SCCCMDS,$SCPCMDS,$SCDOCMD),
        RANGE=(0,X'7FFFFFFF')
```

- **FILTER=** specifies filtering options:
  - **YES** - indicates this keyword may be used as a filter
  - **ALWAYS** - indicates the keyword is always a filter on set commands (and '/' in syntax is optional)
  - **GTLT** - indicates that only > and < filtering is allowed

DISPALL=

```
$SCANTAB NAME=MINUTES,MINLEN=3,
        CB=PARENT,DSECT=JQE,
        FILTER=(YES,ALWAYS,GTLT),
        PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
        FIELD=JQXCRTME,CONV=NUM,
        CALLERS=($SCDCMDS,$SCSCMDS,$SCECMDS,$SCHCMDS,
        $SCRLCMD,$SCCCMDS,$SCPCMDS,$SCDOCMD),
        RANGE=(0,X'7FFFFFFF')
```

- **DISPALL=** specifies when keyword is to be displayed
  - **NO** - display only when requested explicitly ($DJQ,MINUTES)
  - **LONGONLY** -display only when requested explicitly or on long displays ($DJQ,LONG)
  - **YES** - display when requested explicitly, on long displays, and normal display-all situations (display after set, or when no keywords are explicitly requested) ($DJQ)

PRESCAN and PSTSCAN routines

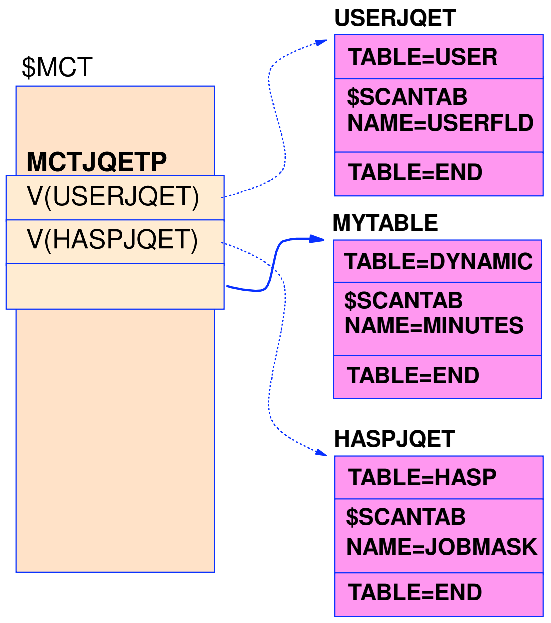
```
$SCANTAB NAME=MINUTES,MINLEN=3,
       CB=PARENT,DSECT=JQE,
       FILTER=(YES,ALWAYS,GTLT),
       PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
       FIELD=JQXCRTME,CONV=NUM,
       CALLERS=($SCDCMDS,$SCSCMDS,$SCECMDS,$SCHCMDS,
       $SCRLCMD,$SCCCMDS,$SCPCMDS,$SCDOCMD),
       RANGE=(0,X'7FFFFFFF')
```

- **PRESCAN=** - allows routines to supplement or override normal processing (before processing keyword)
- **PSTSCAN=** - allows routines to supplement normal processing (after processing keyword)
  - **DISPLAY**, **FILTER**, **SET**, **DELETE** - Indicate specific types of calls a particular routine is to be called for
    - **PSTSCAN=(***routine1***,SET,***routine2***,DISPLAY)**

PRESCAN and PSTSCAN services

- **$SCANB** (set requests)
  - Saves contents of field
  - Contents are automatically restored if command fails
- **$SCAND** (display requests)
  - Adds text to message area
- **SCWA** control block
  - Basic information about current keyword
    - Control block address (**SCWACBAD**)
    - $SCANTAB address (**SCWASTAB**)
    - Current input address and length
    - Address of "higher level" SCWAs

Job command extended table pair



Extending job list commands

- Link-edit USERJQET with HASJES20, or
- Set MCTJQETU to point to user table, or
- Specify TABLE=(DYNAMIC,MCTJQETP)
- Parent CB is a read-mode JQA for display commands
- Parent CB is an update-mode JQA for set commands

How to include the new table

```
MYTABLE   $SCANTAB   TABLE=(DYNAMIC,MCTJQETP)
         $SCANTAB NAME=MINUTES,MINLEN=3,
               CB=PARENT,DSECT=JQE,
               FILTER=(YES,ALWAYS,GTLT),
               PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
               FIELD=JQXCRTME,CONV=NUM,
               CALLERS=($SCDCMDS,$SCSCMDS,$SCECMDS,$SCHCMDS,
               $SCRLCMD,$SCCCMDS,$SCPCMDS,$SCDOCMD),
               RANGE=(0,X'7FFFFFFF')
         $SCANTAB TABLE=END
```

- Link-edit in separate load module
  - LOAD of module automatically links table with table pair **MCTJQETP**
  - Multiple tables may be added to table pair this way

Existing output tables

```
$SCANTAB NAME=HOURS,MINLEN=1,
       CB=PCE,DSECT=JOE,CBIND=(COJWORK,PCE,LA),
       FILTER=(YES,ALWAYS,GTLT),
       PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
       FIELD=JOECRTME,CONV=(NUM,,60),
       RANGE=(0,X'7FFFFFFF')


$SCANTAB NAME=DAYS,MINLEN=2,
       CB=PCE,DSECT=JOE,CBIND=(COJWORK,PCE,LA),
       FILTER=(YES,ALWAYS,GTLT),
       PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
       FIELD=JOECRTME,CONV=(NUM,,60*24),
       RANGE=(0,X'7FFFFFFF')
```

New output table

```
$SCANTAB NAME=MINUTES,MINLEN=3,
       CB=PCE,DSECT=JOE,CBIND=(COJWORK,PCE,LA),
       FILTER=(YES,ALWAYS,GTLT),
       PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
       FIELD=JOECRTME,CONV=NUM,
       RANGE=(0,X'7FFFFFFF')
```

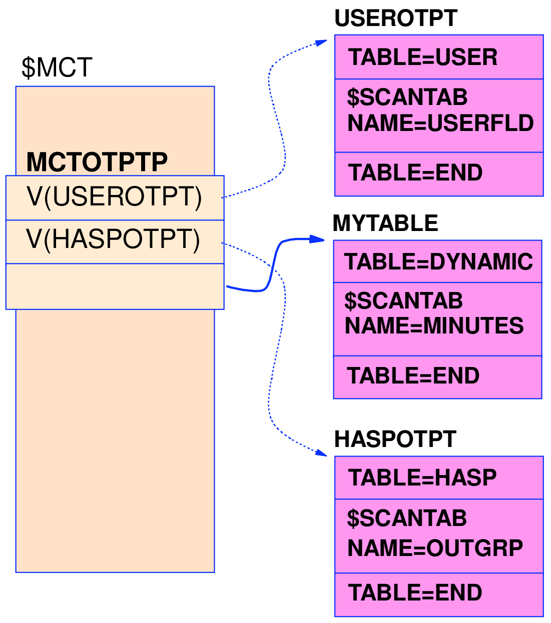
- CB=PARENT is a read-mode JQA
- Copies of work and characteristics JOEs are in PCE
  - CB=PCE,CBIND=(COJWORK,PCE,LA) for work JOE
  - CB=PCE,CBIND=(COCHAR,PCE,LA) for char JOE

CALLERS

```
$SCANTAB NAME=MINUTES,MINLEN=3,
       CB=PCE,DSECT=JOE,CBIND=(COJWORK,PCE,LA),
       FILTER=(YES,ALWAYS,GTLT),
       PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
       FIELD=JOECRTME,CONV=NUM,
       RANGE=(0,X'7FFFFFFF')
```

- CALLERS= is unspecified, indicating all callers from parent $SCANTAB are allowed:
  - $SCLTCMD - $DO
  - $SCTOCMD - $TO
  - $SCPOCMD - $PO
  - $SCCOCMD - $CO
  - $SCOCMDS - $O
  - $SCLOCMD - Display after set

Output command extended table pair



Extending job list commands

- Link-edit USEROTPT with HASJES20, or
- Set MCTOTPTU to point to user table, or
- Specify TABLE=(DYNAMIC,MCTOTPTP)
- Parent CB is a read-mode JQA
- Locate work JOE via CB=PCE, CBIND=(COJWORK,PCE,LA)
- Locate char JOE via CB=PCE, CBIND=(COCHAR,PCE,LA)

How to include the new table

```
MYTABLE   $SCANTAB   TABLE=(DYNAMIC,MCTOTPTP)
      $SCANTAB NAME=MINUTES,MINLEN=3,
             CB=PCE,DSECT=JQE,CBIND=(COJWORK,PCE,LA),
             FILTER=(YES,ALWAYS,GTLT),
             PRESCAN=(PREFOAGE,DISPLAY,FILTER),DISPALL=NO,
             FIELD=JOECRTME,CONV=NUM,
             RANGE=(0,X'7FFFFFFF')
      $SCANTAB TABLE=END
```

- Link-edit in separate load module
  - LOAD of module automatically links table with table pair **MCTOTPTP**
  - Multiple tables may be added to table pair this way

Extending the $JQE control block with

$BERTTABs

Extending the JQE

- Method 1 - Using BERTs to extend the JQA
  - Define your field (or fields) in the JQA

```
JQANOTIF DS      CL8
```

  - Define a $BERTTAB to represent the field(s)

```
$BERTTAB CBTYPE=JQE,
         NAME=NOTIFY,
         CBOFF=JQANOTIF-JQE,
         LEN=L'JQANOTIF,
         PAD=C' '
```

  - Number of $BERTTABs for each CBTYPE is limited (253), so group fields if possible

Extending the JQE

```
$BERTTAB CBTYPE=JQE,
          NAME=NOTIFY,
         CBOFF=JQANOTIF-JQE,
         LEN=L'JQANOTIF,
         PAD=C' '
```

- **CBTYPE=** - defines the control block type
- **NAME=** - defines a symbolic name for the field in the BERTs
- **CBOFF=** - defines the offset of the field in the control block
- **LEN=** - defines the lengh of the field
- **PAD=** - defines the pad character

Extending the JQE

- Include the $BERTTAB in the table pair

```
MYTABLE   $BERTTAB TABLE=DYNAMIC
          $BERTTAB CBTYPE=JQE,
                   NAME=NOTIFY,
                   CBOFF=JQANOTIF-JQE,
                   LEN=L'JQANOTIF,
                   PAD=C' '
          $BERTTAB TABLE=END
```

- No table pair name required, as with $SCANTABs, since there is only one table pair for $BERTTABs

Extending the JQE

- **Method 2** - This can be done without a $JQE modification:

```
$BERTTAB CBTYPE=JQE,
         NAME=NOTIFY,
         CBOFF=*,
         LENGTH=8,
         PAD=C' '
```

- **CBOFF=*** - indicates offset to be assigned dynamically

Extending the JQE

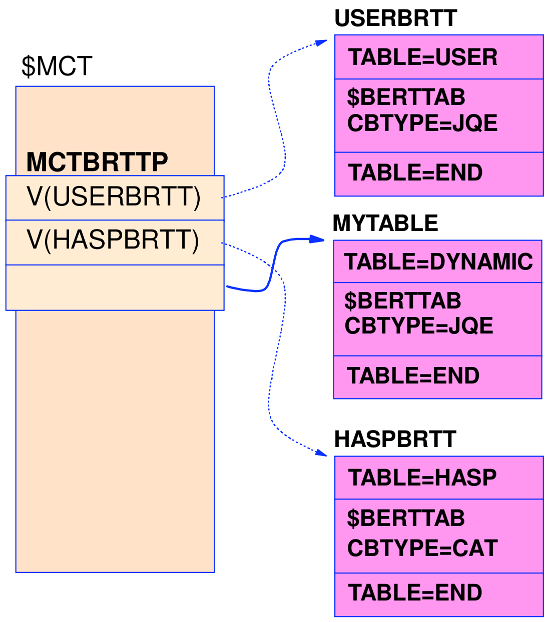
- Obtain offset, when needed, using $DOGBERT:

```
$DOGBERT ACTION=GETOFFSET,
         CBTYPE=JQE,
         NAME=NOTIFY
```

  - **CBTYPE=** matches **CBTYPE=** on $BERTTAB
  - **NAME=** matches **NAME=** on $BERTTAB
  - Offset is returned in register 1
  - Length is returned in register 0

- Can use $DOGBERT call to obtain address of field in PRESCAN routine

$BERTTAB extended table pair



Extending the JQE with BERTs

- Link-edit USERBRTT with HASJES20, or
- Set MCTBRTTU to point to user table, or
- Specify TABLE=DYNAMIC
- Use CBOFF=* in $BERTTAB to add fields independently of control block mapping
- Use $DOGBERT ACTION=GETOFFSET to obtain the offset of fields defined with CBOFF=*

HASXDYNT

- Shipped as sample part in **SHASSAMP**
- Contains simple examples of various table types
  - **$SCANTABs** for **MINUTES=**
  - **$SCANTABs** for **NOTIFY=** (notify userid)
  - **$WSTABs** for **WS=(NOTIFY)**
  - **$BERTTABs** for notify userid associated with $JQE
    - Also $DOGBERT calls to locate section, in PRESCAN routine for NOTIFY= keyword
  - **$EXIT 20** to copy notify userid from $JCT to $JQE