# Lecture 14

# Data Manipulation: Table Processing

Text:
4th Edition: Chapter 15
5th Edition: Chapter 14

A Table is a two-dimensional structure:

Each ROW represents one item on the table
Each COLUMN represents attributes of the item

Example: Major codes and titles

| | |
|---|---|
| 026 | Computer Science BA |
| 027 | Computer Science BS |
| | |
| | |
| | |
| | |

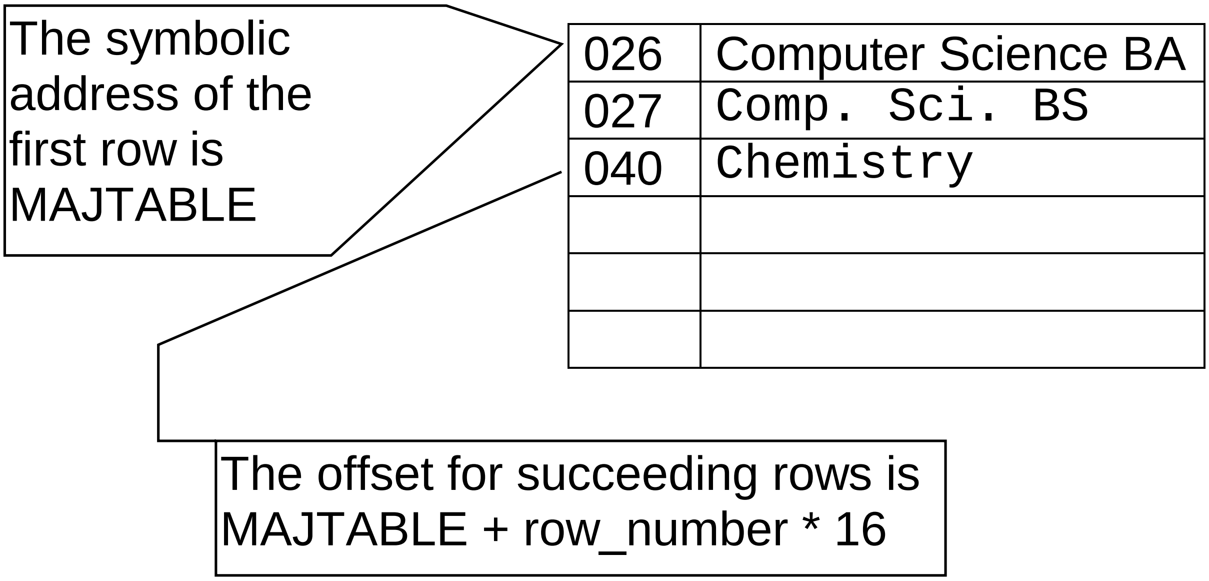
In order for the table to be used in a program, each row must be the same size. It could be defined as:

```
MAJTABLE  DB 026,’Computer Sci BA’
          DB 027,’Computer Sci BS’
          DB 040,’Chemistry      ’
          DB 050,’Physics        ’
```

In this case, the code is stored as a byte (as long as it is less than 256), and the major is a string of 15 characters. The length of a row is therefore 16.

Note that the table is stored SEQUENTIALLY in memory as a long stream of bytes.

To access a row of the table the address of the first byte of the row is needed.



The symbolic address of the first row is MAJTABLE

| | |
|---|---|
| 026 | Computer Science BA |
| 027 | Comp. Sci. BS |
| 040 | Chemistry |
| | |
| | |
| | |

The offset for succeeding rows is MAJTABLE + row_number * 16

The offset for the row with code 040 is MAJTABLE+32.

Note that the row addresses start with zero.

| | | |
|---|---|---|
| MAJTABLE+0 | 026 | Comp. Sci. BA |
| MAJTABLE+16 | 027 | Comp. Sci. BS |
| MAJTABLE+32 | 040 | Chemistry |
| MAJTABLE+48 | 050 | Physics |
| MAJTABLE+64 | | |
| MAJTABLE+80 | | |

To get the address of a row,

subtract one from the row number<br>
multiply the row number by the size of a row<br>
add the result to the address of the table

Given a row number in “TheRow”, move it to a variable called “TheMajor”:

```
 RowLen  DW   16
 TheRow  DW   3            ;The third row

         MOV  AX,TheRow    ;get row num
         DEC  AX           ; and adjust
         MUL  RowLen       ;size of row
         MOV  CX,16        ;move 16 bytes
         LEA  SI,MAJTABLE  ;get addr of 1st
         ADD  SI,AX        ; row plus offset
         LEA  DI,TheMajor  ;put row here
         REP  MOVSB        ;do the move
```

If you just wanted the major, and not the code:

```
RowLen   DW   16
TheRow   DW   3            ;The third row

         MOV  AX,TheRow    ;get row num
         DEC  AX           ; and adjust
         MUL  RowLen       ;size of row
         MOV  CX,15        ;move 15 bytes
         LEA  SI,MAJTABLE  ;set src to 1st
         ADD  SI,AX        ; row plus offset
         INC  SI           ;skip 1st column
         LEA  DI,TheMajor  ;put row here
         REP  MOVSB        ;do the move
```

## A More Interesting Problem

If you don’t know the row, you need to SEARCH the table.

A SEQUENTIAL SEARCH will start at the beginning of the table and compare each row until

the item is found, or<br>
the table is completely searched

```
TheCode  DB   ?         ;code to search for

         MOV  CX,10     ;table has 10 rows
         MOV  AL,TheCode
         LEA  SI,MajTable
SrchLoop:
         CMP  AL,[SI]   ;Is this it?
         JE   FoundIt   ;If so, exit
         ADD  SI,16     ;else next row
         LOOP SrchLoop  ;check again
; If you reach here, the item is not on
; the table.
FoundIt:
; If you reach here, the item is on
; the table
```

```
                page     60,132
TITLE   P15TABSR (COM)  Table search Using CMP
        .MODEL  SMALL
        .CODE
        ORG     100H
BEGIN:  JMP     SHORT MAIN
; -----------------------------------------------
STOKNIN DB      ’12’      ;Input stock no.
STOKTAB DB      ’05’,’Excavators’ ;Start of table
        DB      ’10’,’Lifters   ’      ;
        DB      ’12’,’Presses   ’      ;
        DB      ’15’,’Valves    ’      ;
        DB      ’23’,’Processors’      ;
        DB      ’27’,’Pumps     ’ ;End of table
DESCRN  DB      10 DUP(?),’$’     ;Save area
; -----------------------------------------------
MAIN    PROC    NEAR
        MOV     CX,06     ;Initialize compares
        LEA     SI,STOKTAB
A20:
        MOV     AL,STOKNIN
        CMP     AL,[SI]   ;Stock#(1) : table
        JNE     A30       ;Not equal - exit
        MOV     AL,STOKNIN+1  ;Equal:
        CMP     AL,[SI+1] ;  stock#(2) : table
        JE      A50       ;  equal -- found
A30:
        JB      A40       ;Low -- not in table
        ADD     SI,12     ;High -- get next entry
        LOOP   A20
A40:    ;Not in table
;       .       ;Display error message
        JMP     A90
```

*continued...*

```
A50:
        MOV     CX,10     ;Length of description
        LEA     DI,DESCRN ;Address of description
        INC     SI
        INC     SI        ;Extract description
        REP MOVSB         ;  from table
        MOV     AH,09H    ;Request display
        LEA     DX,DESCRN ;  stock description
        INT     21H
;       .
A90:    MOV     AX,4C00H  ;Exit to DOS
        INT     21H
MAIN    ENDP
        END     BEGIN
```

## Comparing Longer Items

In the previous examples, the items being compared were one byte big and could be compared using CMP.

If the item sought is a string, perhaps a NAME, then it is possible the use REPE (Repeat on Equal) with a CMPSB (Compare String Bytes).

Suppose we have a table containing nine-digit character string ID numbers and 20-character names:

| | |
|---|---|
| 123456789 | Beethoven, Ludwig |
| 235437543 | Bach, Johann |
| 874932847 | Handel, George |
| … | … |

```
TheID   DB  ’235437543’
IDTable DB  ’123456789Beethoven, Ludwig   ’
        DB  ’235437543Bach, Johann        ’
       (etc.) (presume there are 10 names)
LastRow DW  IDTable+290 ;(9+20)*10=290bytes
```

```
       LEA   DI,IDTable
SrchLoop:
       LEA   SI,TheID
       MOV   CX,9        ;if the 9-digit ID
       REPE  CMPSB       ;  matches the table
       JE    FoundIt     ;  stop searching
       ADD   DI,CX       ;else skip rem ID
       ADD   DI,20       ;begin. of next row
       CMP   DI,LastRow
       JL    SrchLoop
;  ID not on table
```

## XLAT - The Translate Instruction

Use the byte in the AL register as an offset to a table, and load the AL with the byte found in the table.

Example:  ASCII to EBCDIC numbers...

| ASCII | 2D | 2E | 2F | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Char | - | . | / | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | @ |
| EBCDIC | 60 | 4B | 40 | F0 | F1 | F2 | F3 | F4 | F5 | F6 | F7 | F8 | F9 | 40 |

The number -31.5 in

```
ASCII:    2D 33 31 2E 35
EBCDIC:   60 F3 F1 4B F5
```

```
ASCNO  DB     ’-31.5 ’  ;ASCII item to convert
EBCNO  DB     6 DUP(’ ’) ;Converted EBCDIC item
XLTAB  DB     45 DUP(40H); first 2Dh bytes = 40h
       DB     60H, 4BH   ; 2E(’-’)=60, 2F(’.’)=4B
       DB     40H        ; skip 30h
       DB     0F0H,0F1H,0F2H,0F3H,0F4H ;31h to
       DB     0F5H,0F6H,0F7H,0F8H,0F9H ;39h
       DB     198 DUP(40H) ;rem of table is 40h
;--------------------------------------------
MAIN   PROC   NEAR
       LEA    SI,ASCNO  ;Address of ASCNO
       LEA    DI,EBCNO  ;Address of EBCNO
       MOV    CX,06     ;Length of items
       LEA    BX,XLTAB  ;Address of table
A20:
       LODSB  ;Get ASCII char in AL
       XLAT   ;Translate character
       STOSB  ;Store AL in EBCNO
       LOOP   A20       ;Repeat 6 times
```

P15XLATE.ASM

## Example: Change all Upper case to Lower case

```
Upper   DB     ’THIS IS ALL UPPER CASE’
Lower   DB     22 DUP(’ ’)
;  ’A’ is ASCII 41h (Decimal 65)
;  ’Z’ is ASCII 5Ah (Decimal 90)
XLTAB   DB     44 DUP(20h);00h-2Bh become 20h
        DB     2CH      ; leave "," alone
        DB     20h      ; "-" becomes 20h
        DB     2Eh      ; leave "." alone
        DB     20h      ; "/" becomes 20h
        DB     30H,31H,32H,33H,34H ;leave nums
        DB     35H,36H,37H,38H,39H ;   alone
        DB     20h      ;"@" becomes 20h
;The upper case positions are filled with the
;lower case ASCII codes:
        DB     61H,62H,63H,64H,65H,66H,
        DB     67H,68H,69H...
        ...
        DB     56h,57h,58h,59h,5Ah
        DB     6 DUP (20h) ; 5Bh-60h
;The lower case ASCII codes are the same:
        DB     61H,62H,63H,64H,65H,66H,
        DB     67H,68H,69H...
        ...
        DB     76H,77H,78H,79H,7AH
        DB     144 DUP(20h)
;--------------------------------------------
MAIN    PROC   NEAR
        LEA    SI,Upper ;Upper case to convert
        LEA    DI,Lower ;  to Lower case
        MOV    CX,SIZE Upper
        LEA    BX,XLTAB ;Address of table
ReplLoop:
        LODSB  ;Get ASCII char in AL
        XLAT   ;Translate character
        STOSB  ;Store AL in Lower
        LOOP   ReplLoop ;Do the whole line
```

# Exercises - Lecture 14

1. Write a program that will

    1. define a table which will contain a 20 character name, 3-digit area code, 7-digit phone number (no hyphen), and a byte which indicates if this is a home (’H’) or office (’O’) number.
    2. print to the screen the names and phone numbers of all people in the table who are in the 718 area code. Include parenthesis around the area code, and a hyphen in the phone number.

2. Suppose that whoever typed in the above table used a lower-case "l" instead of a digit one ("1"). Include an XLAT table and instruction in your program to change the l’s to 1’s before you print the name and phone number.