INFO3105 Week 3 Class 2

Review

- Intro to Summary Report

Let’s review the process to get currency formatted data to the printer:

1. Read the data (this field layout is part of the given salesperson master file layout):

```
…
05 SALESPERSON-GROSS-SALES     PIC 9(5)V99.
05 SALESPERSON-RETURN-SALES    PIC 9(4)V99.
05 SALESPERSON-BRANCH-NO       PIC 9(3).
05 SALESPERSON-COMM-RATE       PIC V9999.
```

2. To make some calculation with the data, we need to introduce some additional fields to hold the results of our calculations, so in working storage you’d define a set of numeric fields (Notice they are signed, and larger than the input fields...):

```
01 WS-CALC-FIELDS.
   05 WS-BRANCH-TOTAL-GROSS      PIC S9(7)V99   VALUE ZERO.
   05 WS-BRANCH-TOTAL-COMMISS    PIC S9(7)V99   VALUE ZERO.
   05 WS-NET-SALES               PIC S9(7)V99   VALUE ZERO.
   05 WS-COMMISSION-EARNED       PIC S9(7)V99   VALUE ZERO.
   …
```

3. Do the actual calculations using a combination of both sets of fields in a PROCEDURE DIVISON paragraph:

```
U200-SALESPERSON-CALCULATIONS.
    COMPUTE
       WS-BRANCH-TOTAL-GROSS = WS-BRANCH-TOTAL-GROSS +
       SALESPERSON-GROSS-SALES
    END-COMPUTE.
   etc. …
```

Working storage field

Input file field

4. When we’ve completed all calculations on the field we need to print out the number as part of one of the lines on the report. To use currency formatting we need to move our calculated field to a numeric edited counterpart:

```
01 WS-BRANCH-TOTAL-LINE.
    05 FILLER                     PIC X(3)    VALUE SPACES.
    05 FILLER                     PIC X(6)    VALUE "TOTAL ".
    05 FILLER                     PIC X(7)    VALUE "BRANCH".
    05 WS-BT-BRANCHNO             PIC ZZ9.
    05 FILLER                     PIC X(12)   VALUE SPACES.
    05 WS-BT-SLSP-GROSS           PIC $$,$$$,$$$.99.
    05 FILLER                     PIC X(2)    VALUE SPACES.
    etc. …
```

5. Then in our logic (again in the procedure division) we move the calculated field to the report line and write it:

Working storage field

Formatted output field

```
W400-PRINT-BRANCH-TOTALS.
          MOVE WS-BRANCH-TOTAL-GROSS TO WS-BT-SLSP-GROSS.
          etc. …
          …
          MOVE WS-BRANCH-TOTAL-LINE TO PRINT-AREA.
          WRITE PRINT-AREA AFTER ADVANCING 2 LINES.
```

Control Break Notes

- Records must be in sequence by control field (we’ll use branch no - ie. Branch #)
- Records for 1st branch read in, printed, branch total accumulated
- Branch number will be stored in a control field (working storage)
- Read and Writes continue until record read in has a different branch number
- Change in branch indicates the **Control Break** therefore we need to:
    - Print control total for previous group
    - Accumulate branch total to grand total
    - Initialize branch total field to zero
    - Reinitialize control field with new branch value
    - Perform detail processing as in previous step
- Last group of records has no subsequent record to trigger printing therefore we need an end of job routine to move final group total to output record and print it and print any grand totals

Control Break Summary Report cont'd

Traditionally programmers hammered out logic like above in one of two ways, they either drew a flowchart or they coded what was deemed pseudocode. Below is a simple example of pseudocode for a typical summary report (control break) program, the textbook also shows an example of pseudocode on **pages 124-125**:

```
000-MAINLINE.
     open the files.
     do LOOP-PARAGRAPH until there are no more records
     close the files.

LOOP-PARAGRAPH.
     read a record.
     if it's not EOF
         if it's the first record
             save the value of the break field
             PRINT-THE-HEADERS
             Process the current record (calcs and print details)
         else
             If the saved break field changed
                perform the control break
                process the current record (calcs and print details)
             else
                process the current record (calcs and print details)
     else
         print out the totals for current branch
         print out the final totals

PRINT-THE-HEADERS.
     write the header lines.
     increment the page number.
     reset the detail line count back to zero

CONTROL-BREAK-PARAGRAPH.
    print totals for current branch
    print headings for next branch
    reinitialize any counters or sub totals
    save the value of the break field
```

Notes for coding from Pseudocode

For the "process the current record" step we need to do the calculation routine that you coded for homework to update the various fields. Then once we have calculated the fields and incremented the totals we would write the details.

The "control break" can be done in its own routine or if there isn't that much to do you can just include the statements in the contents of the if-else. In our case when the branch number changes we need to:

- dump out the branch totals for the previous branch
- print the headings for the next page (increment the page no)

- process the current record
- store the new number to the control field

With the main logic laid out, it is just a matter of transferring this pseudocode into real COBOL syntax.

Opening and closing the files is similar to what we've already done in the previous labs, it's just a matter of getting the correct input file for the salesperson data:

```
FILE-CONTROL.

    SELECT SALESMAST ASSIGN TO SLINPUT
        FILE STATUS IS WS-IN-STATUS.

    SELECT SALESRPT ASSIGN TO PRNT.

DATA DIVISION.

     FILE SECTION.

     FD  SALESMAST
         RECORDING MODE IS F.
     01 SALESPERSON-MASTER-YTD-SALES.
        05 SALESPERSON-NO                PIC 9(5).
        05 SALESPERSON-LAST-NAME         PIC X(15).
           ...
```

Filename found in COBOL code

Logical names found in JCL

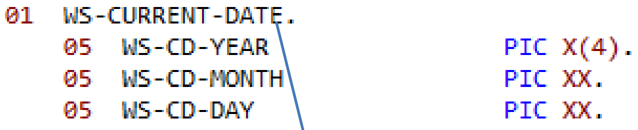
You'll need to add a working storage field called **WS-IN-STATUS** that has a picture clause of XX. Then if there is a problem with the file you can look up this status from the following link: http://www.simotime.com/vsmfsk01.htm . Also you should put the READ in its own routine, and remove the INTO clause or you will get an overlapping storage compile error (Cobol is fussy - the INTO clause is only allowed to read into Working storage defined fields, not FILE SECTION defined fields).

```
R200-READ-SALESPERSON-RECORD.

    READ SALESMAST
        AT END
         SET WS-SALESMAST-EOF TO TRUE.
```

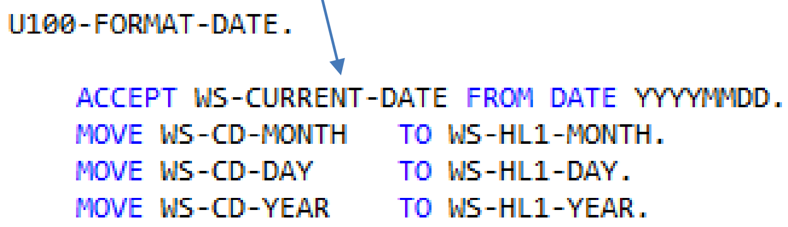
Obtaining System Date

If you look at the report from last class the current date is on the report. We can get the current date from the OS by laying out the date in working storage like this:



```
01  WS-CURRENT-DATE.
    05  WS-CD-YEAR                  PIC X(4).
    05  WS-CD-MONTH                 PIC XX.
    05  WS-CD-DAY                   PIC XX.
```

And then writing a utility routine like this:

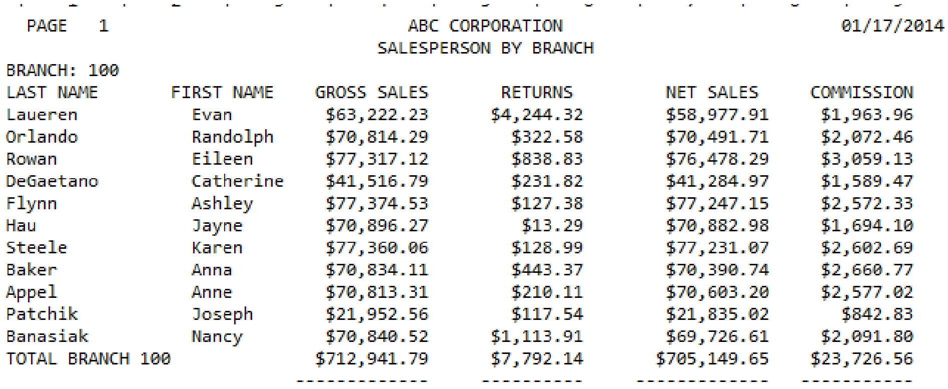


```
 U100-FORMAT-DATE.

      ACCEPT WS-CURRENT-DATE FROM DATE YYYYMMDD.
      MOVE WS-CD-MONTH    TO WS-HL1-MONTH.
      MOVE WS-CD-DAY      TO WS-HL1-DAY.
      MOVE WS-CD-YEAR     TO WS-HL1-YEAR.
```

Lab 6 - (4%)

- Edit the Sales Master data and change the Your name fields to your actual name (first salesperson of the first branch), use the same #'s so make sure you just change the name:
  02153Yourlast   Yourfirst   6322223424432100033
- Convert the pseudocode to actual syntax for the remainder of the program. If you want to work your logic different from the author that's fine, for instance you may not see the need for a separate control break headings print routine)
- See last class' .pdf for the totals you should be trying to get.
- For this lab get the program to compile cleanly and minimally print the headings and first branch details including totals.



```
  PAGE   1                          ABC CORPORATION                         01/17/2014
                                  SALESPERSON BY BRANCH
BRANCH: 100
LAST NAME       FIRST NAME     GROSS SALES        RETURNS        NET SALES     COMMISSION
Laueren           Evan          $63,222.23      $4,244.32       $58,977.91      $1,963.96
Orlando           Randolph      $70,814.29        $322.58       $70,491.71      $2,072.46
Rowan             Eileen        $77,317.12        $838.83       $76,478.29      $3,059.13
DeGaetano         Catherine     $41,516.79        $231.82       $41,284.97      $1,589.47
Flynn             Ashley        $77,374.53        $127.38       $77,247.15      $2,572.33
Hau               Jayne         $70,896.27         $13.29       $70,882.98      $1,694.10
Steele            Karen         $77,360.06        $128.99       $77,231.07      $2,602.69
Baker             Anna          $70,834.11        $443.37       $70,390.74      $2,660.77
Appel             Anne          $70,813.31        $210.11       $70,603.20      $2,577.02
Patchik           Joseph        $21,952.56        $117.54       $21,835.02        $842.83
Banasiak          Nancy         $70,840.52      $1,113.91       $69,726.61      $2,091.80
TOTAL BRANCH 100               $712,941.79      $7,792.14      $705,149.65     $23,726.56
                              -------------    ----------    -------------    -----------
```

SUBMIT the entire SPOOL file (all parts of the job) from JES for your job to the dropbox

- Next week we'll add a few more things to the report and finish getting the control break logic cleaned up and final totals working.
- Read pages **276-285** (working with tables) from the text