**LAB Assessment - 2**

1. ISPF

Name: UID.MEDICINE.PS1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MEDICINE\_CODE** | **QUANTITY** | **EXP\_STATUS** | **DOSAGE** | **RATE\_PER\_NO** |
| **X(03)** | **9(03)** | **X(07)** | **X(01)** | **9(02).99** |
| P01 | 080 | NOT-EXP | H | 10.15 |
| P03 | 040 | NOT-EXP | H | 05.50 |
| P02 | 030 | NOT-EXP | L | 28.36 |
| P04 | 080 | EXP | L | 09.52 |
| P06 | 150 | EXP | L | 06.25 |
| P03 | 040 | NOT-EXP | H | 05.50 |
| P02 | 035 | NOT-EXP | L | 28.36 |
| P05 | 260 | NOT-EXP | H | 13.84 |
| P09 | 079 | NOT-EXP | M | 08.56 |
| P07 | 060 | NOT-EXP | M | 18.00 |
| P08 | 045 | NOT-EXP | M | 15.00 |
| P09 | 010 | NOT-EXP | M | 08.56 |

1. JCL

JOB 1: Allocate a VSAM KSDS with the following specification:

Name: UID.MED.NEWORD.KSDS

RECORDSIZE (80 80)

KEY FIELD: MEDICINE CODE X(03)

JOB 2:

STEP 1) Using the Sort utility perform the below operations on PS1 and store the output in a new PS file UID.MEDICINE.PS2

1. SORT the records based on the medicine code
2. Eliminate the duplicates by adding the Quantity field
3. Remove the header and rearrange the positions of fields as below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Medicine\_code  X(03) | Dosage  X(01) | Quantity  9(03) | Expiry Status  X(07) | Rate\_per\_no  9(02).9(02) |

Step 2) Split the file into 2 new files based on the Expiry\_status, and the layout will be same as PS2

UID.MEDICINE.NOTEXP

UID.MEDICINE.EXP

COBOL:

INPUT FILE : UID.MEDICINE.PS2

DD NAME : INPMEDPS

OUTPUT FILE : UID.MED.NEWORD.KSDS

DD NAME : OUTMEDPS

OUTPUT FILE : UID.MEDICINE.SCH2

DD NAME : OUTMEDS2

OUTPUTFILE : UID.MEDICINE.ERR

DD NAME : OUTMEDER

Write a COBOL program for the following

* Read the records from the input PS2 dataset and validate input fields for each field
  + Check whether the Medicine code is not blank
  + Check whether the dosage fields contains values ‘L’, or ’M’, or ‘H’
  + Check whether the quantity is numeric
  + Check whether the expiry status contains ‘NOT-EXP’ or ‘EXP’
  + Validate Rate\_per\_no for numeric before and after decimal point
* If the input records does not pass through the validates, then write the record into ERROR file UID.MEDICINE.ERR and start processing the next record
* Business Processing:
  + If the quantity of medicine is less than 100 and Expiry\_status is ‘NOT-EXP’
    - Set new Quantity as 250 and Order\_Status as ‘REORDER’
  + If the quantity of medicine is less than 100 and Expiry\_status is ‘EXP’
    - Set new Quantity as 50 and Order\_Status as ‘REPLACE’
  + If the quantity of medicine is greater than 100 and Expiry\_status is ‘EXP’
    - Set new Quantity as 0 and Order\_Status as ‘STPSELL’
  + If none of the above conditions satisfied, skip that record from further processing
* Class Calculation:’
  + If Dosage = ‘L’ then Class is ‘SCH1’
  + If Dosage = ‘M’ then Class is ‘SCH2’
  + If Dosage = ‘H’ then Class is ‘SCH3’
* Discounted rate\_per\_no and tot\_price calculation has to be done in a SUB program
  + Calculate discounted rate\_per\_no and Tot\_price using the formulas
  + Tot\_price value should be rounded off to the two decimal values

DISC Rate\_per\_no = Rate\_per\_no – (Rate\_per\_no \* 20 /100)

TOT\_PRICE = New Quantity \* Disc Rate\_per\_no

* + Load the data into a COBOL one dimensional table. Format of the table is as follows:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Medicine\_code  X(03) | Quantity  9(03) | Expiry Status  X(07) | Order\_status  X(07) | CLASS  X(04) | Rate\_per\_no  99.00 | Disc\_Rate\_per\_no  99.99 | Tot\_price  9(05).99 |

* + From COBOL table, retrieve the records with CLASS=’SCH1’ or ‘SCH3’and write the same in to the KSDS file
  + From the COBOL table, retrieve the records with CLASS=’SCH2’and write the same into the PS file UID.MEDICINE.SCH2