Medical Severity Diagnosis Related Groups (MS-DRG) Software

Software Installation Guide for z/OS Batch

ICD-10 Pilot

v30.0 Pilot January 2013

PBL-036
January 2013

Table of Contents

Preface

This manual contains the information needed to use the Medicare Severity Diagnosis Related Groups (MS-DRG) Software (the grouper), version 30.0 Pilot in a mainframe environment. Two interface versions to the MS-DRG software are supplied. One, the standard version, assumes that the operating system is z/OS Batch. The second is re-entrant and uses no macros and so can be used in a variety of operating system environments, although it requires additional parameters from the calling program.

This manual provides technical personnel with the detail necessary to install, debug, and support the MS-DRG software. The first four chapters describe installing, testing, and running the grouper. Chapter 5 provides detailed information on the logic of the executor and the construction of the tables. An appendix provides grouping results for the test database.

Users already familiar with the MS-DRG software are encouraged to read this manual to ensure that installation, testing, and production runs perform without incident. If you have never used the software, we strongly recommend that you read the manual thoroughly to become familiar with it before installation.

The manual assumes that you are familiar with:

- IBM Basic Assembler Language (BAL)
- IBM MVS Job Control Language

Chapter 1: Introduction

This manual provides technical personnel with the detail necessary to install and understand the Medicare Severity Diagnosis Related Groups (MS-DRG) Software (the grouper) so they can install, interface with, and support it.

The MS-DRG software may be implemented either as a set of subroutines to be called from a program written in Assembler or a higher level language (e.g., COBOL) or as a utility program with all parameters passed through a job's SYSIN input stream.

For example purposes, the required datasets are copied to disk and cataloged under the Userid, GROUPER.

Data format requirements

The grouper executor is contained in three Basic Assembler Language (BAL) programs. The data formats required by the executor are shown in the table that follows.

If these data requirements are met, the grouper may be implemented by using a utility program (see "Using and testing the grouper utility" on page 37). Whenever these requirements are not met, the grouper must be implemented as a subroutine to a higher level language program that re-codes the information as necessary (see "Using the grouper with higher-level languages" on page 45).

Table 1. Required data formats

Name	Length in bytes	Description
Diagnosis	8	First 7 bytes represent the diagnosis code. Left-justified, blank-filled, up to 25 accepted. The eighth byte represents the POA indicator. Y - Yes, present at the time of inpatient admission N - No, not present at the time of inpatient admission U - Insufficient documentation to determine if present on admission W - Clinically unable to determine if present at time of admission 1 - Code is exempt from POA reporting (Used on 4010 form) Blank - Code is exempt from POA reporting (Used on 5010 form, effective 01/01/2011)
Procedure	7	Left-justified, blank-filled, up to 25 accepted

Name	Length in bytes	Description
Age	3	0 (zero) through 124, right-justified
Sex	1	1 or 2 (1-male, 2-female)
Discharge Status	2	01-Home, Self-Care 02-Short Term Hosp 03-SNF 04-Custodial/supportive care (effective 10/1/2009) 05-Canc/Child hosp 06-Home Health Service 07-Against Medical Advice 20-Died 21-Court/law enfrc 30-Still A Patient 43-FedHospital 50-Hospice-Home 51-Hospice-Medical Facility 61-Swing Bed 62-Rehab facility/rehab unit 63-Long term care hospital 64-Nursing facility - Medicaid certified 65-Psych hosp/unit 66-Critical Access Hospital 70-Oth institution
POA logic	1	Present On Admission (POA) logic indicator. X - Exempt from POA reporting Z - Requires POA reporting
Admit Date	8	Format = YYYYMMDD (for use with future POA logic)
Discharge Date	8	Format = YYYYMMDD (for use with future POA logic)
Procedure Dates	200	Date of each procedure code Format = YYYYMMDD (for use with future POA logic)

Information returned by the grouper

The information returned by the grouper is shown in the following tables.

The field DRG listed below represents the 3-digit MS-DRG number used by the Centers for Medicare and Medicaid Services (CMS) for DRG payment purposes. The 3-byte “initial DRG” field in the ancillary buffer represents the DRG prior to the application of the HAC logic. The ancillary buffer also contains 4-byte initial and final DRG numbers. These 4-byte DRG numbers are for statistical purposes only. Each 3-digit DRG concept is split on MCC, CC, and non-CC to create the 4-digit DRG.

For example, as a 3-digit DRG, Non-specific CVA & precerebral occlusion w/o infarction is split into 067 (w MCC) and 068 (w/o MCC). As a 4-digit DRG, Non-specific CVA & precerebral occlusion w/o infarction is split into 0671 (w MCC), 0672 (w CC), and 0673 (w/o CC/MCC). There are also “initial” and “final” flags in the diagnosis flag buffer. The flags indicating which secondary diagnoses and procedures affect DRG assignment apply to the 3-digit DRG, but could be different for the 4-digit DRG.

The grouper executor may be implemented as a subroutine to be called from Assembler or a higher-level language program. This chapter shows how this may be done for a COBOL programming environment. To create the subroutines, you must have copied file 1 (the grouper objlib) from the media to disk (see "Installing the MS-DRG Software" on page 19).

Table 2. Information returned by the MS-DRG software

Name	Length in bytes	Description
RTC	2	Grouper return code (page 13) 00 - Record grouped 01 - Diagnosis code cannot be used as principal dx 02 - Record does not meet criteria for any DRG in the MDC that is indicated by principal dx 03 - Invalid age 04 - Invalid sex 05 - Invalid discharge status 06 - Illogical principal diagnosis 07 - Invalid principal diagnosis 09 - POA logic indicator = Z and at least one HAC POA is invalid, missing, or 1 10 - POA logic indicator is invalid or missing and at least one HAC POA is N or U 11 - POA logic indicator is missing or invalid, and at least one HAC POA is invalid, missing, or 1 12 - (not valid effective 10/1/2010) POA logic indicator = Z and at least one HAC POA =1 13 - (not valid effective 10/1/2010) POA logic indicator is invalid or missing and at least one HAC POA = 1 14 - (not valid effective 10/1/2010) POA logic indicator = Z and there are multiple HACs that have different HAC POA values that are not Y, W, N, U 15 - POA logic indicator is missing or invalid, and there are multiple HACs that have different HAC POA values that are not Y or W
Final MDC	2	Major Diagnostic Category number (00 - 25) assigned to patient record
Final DRG	4	Diagnosis Related Group number (0001 - 0999) assigned to patient record (after HAC logic is applied)
GRFLGS	5	See table for "Grouper flags returned by the MS-DRG software" (page 14)
DXFLGS	150	See table for "Diagnosis flags returned by the grouper" (page 15)
PRFLGS	175	See table for "Procedure flags returned by the grouper" (page 16)

Grouper return code

The grouper return code (RTC) indicates whether or not the grouping process was successful for a given record. The following table describes the values for the Return Code.

Table 3. Return code descriptions

Return code	Description
1	The first listed diagnosis is a valid code but it can not be used as principal diagnosis. An example of this situation would be any one of the ICD-10-CM "V" codes, which are not indicative of the MDC into which this patient should be classified.
2	This code occurs when all of the DRG criteria for the MDC have been examined and the record does not match any of them.
3, 4 and 5	These codes occur only for those DRGs that are part of grouping criteria (i.e., the grouper does not perform an automatic edit check of age, sex, and discharge status).
6	The principal diagnosis is considered illogical, meaning that it is unlikely that there would be an occurrence. For example, diagnosis code 76509 (extreme immat 2500+g) is flagged as an illogical diagnosis whenever it is coded as the principal diagnosis.
7	The code used as principal diagnosis is not a valid ICD-10-CM code.
9, 10, 11, 15	These codes occur when there is at least one HAC on the record and there is an issue with either the POA logic indicator or the POA values assigned to the HAC.

Flags returned by the grouper

The information returned by the grouper regarding DRGs, diagnoses, and procedures.

Table 4. Grouper flags returned by the MS-DRG software

Position	Description
1 and 2	Number of unique Hospital Acquired Conditions (HAC) met
3	Final CC/MCC impact on DRG assignment: 0 = DRG assigned is not based on the presence of a CC or MCC 1 = DRG assigned is based on presence of MCC 2 = DRG assigned is based on presence of CC
4	Initial CC/MCC impact on DRG assignment: 0 = DRG assigned is not based on the presence of a CC or MCC 1 = DRG assigned is based on presence of MCC 2 = DRG assigned is based on presence of CC
5	HAC Status 0 = HAC Not Applicable; Hospital is exempt or HAC criteria not met 1 = Criteria for one or more HACs met, Final DRG did not change 2 = Criteria for one or more HACs met, Final DRG changed 3 = Criteria for one or more HACs met, Final DRG changed to 999

Table 5. Diagnosis flags returned by the grouper

Position	Description (6 characters per diagnosis)
1	0 = Diagnosis invalid 1 = Diagnosis valid
2	Diagnosis affects DRG 0 = Diagnosis not used to assign DRG 1 = Diagnosis affected the initial DRG only 2 = Diagnosis affected the final DRG only 3 = Diagnosis affected both initial and final DRG
3	CC/MCC Categorization 0 = Diagnosis is not considered a Major CC or CC for this patient 1 = Diagnosis is a Major CC for both initial and final DRG 2 = Diagnosis is a CC for both initial and final DRG 3 = Diagnosis is a MCC for initial DRG and a Non-CC for final DRG 4 = Diagnosis is a CC for initial DRG and a Non-CC for final DRG
4 and 5	Hospital Acquired Condition (HAC) assignment criteria 00 = Criteria to be assigned as an HAC not met 01 = Foreign Object Retained After Surgery 02 = Air Embolism 03 = Blood Incompatibility 04 = Pressure Ulcers 05 = Falls and Trauma 06 = Catheter Associated UTI 07 = Vascular Catheter-Associated Infection 08 = Infection after CABG 09 = Manifestations of poor glycemic control 10 = DVT/PE after knee or hip replacement 11 = Infection after bariatric surgery 12 = Infection after certain orthopedic procedures of spine, shoulder and elbow 13 = Surgical site infection following cardiac device procedures 14 = Iatrogenic pneumothorax w/ venous catheterization
6	Hospital Acquired Condition (HAC) Status 0 = HAC not applicable 1 = HAC criteria met 2 = HAC criteria not met

Table 6. Procedure flags returned by the grouper

Position	Description (7 characters per procedure)
1	0 = Procedure invalid 1 = Procedure valid
2	Procedure affects DRG* 0 = Procedure did not affect DRG assignment 1 = Procedure affected the initial DRG assignment only 2 = Procedure affected the final DRG assignment only 3 = Procedure affected both initial and final DRG assignment * When there are two or more procedures on the record that could impact either the initial, final or both DRG assignments: If one of these procedures is in the first procedure position, that procedure will be be flagged as 1, 2 or 3 with the following exceptions: a. If a single procedure designating a complete system is tied with a combination pair that also designated a complete system, the single procedure will be flagged regardless of position. b. If multiple combinations of lead/device pairs are tied then only one pair will be flagged regardless of position. c. If the two procedures tied are an OR and non-OR, the OR will be flagged regardless of position. If none of the tied procedures is in the first procedure position, then the procedure with the lowest ascii/index value will be flagged as 1, 2 or 3.
3	0= Procedure is not an OR procedure 1 = Procedure is an OR procedure
4 and 5	Hospital Acquired Condition (HAC) assignment criteria 00 = Criteria to be assigned as an HAC not met 08 = Infection after CABG 10 = DVT/PE after knee or hip replacement 11 = Infection after bariatric surgery 12 = Infection after certain orthopedic procedures of spine, shoulder and elbow 13 = Surgical site infection following cardiac device procedures 14 = Iatrogenic pneumothorax w/ venous catheterization
6	Not used
7	Not used

Ancillary buffer

The version number identifies the version of the grouper that is running.

Table 7. Additional flag information

Length in bytes	Description
5	1 byte reserved space (zero-filled) followed by 4-byte final DRG (after HAC logic applied)
1	Final DRG Medical/Surgical Indicator 0 = RTC is non-zero 1 = Medical DRG 2 = Surgical DRG
4	1 byte reserved space (zero-filled) followed by 3-byte initial DRG (prior to HAC logic)
5	1 byte reserved space (zero-filled) followed by 4-byte initial DRG (prior to HAC logic)
1	Initial DRG Medical/Surgical indicator 0 = RTC is non-zero 1 = Medical DRG 2 = Surgical DRG

8	Version ID returned by the grouper (PPPVVVUU)
	
	PPP = 001 (MS-DRG)
	VVV = T300 (Grouper version 30.0 Pilot)
	UU = 00 (update 00)

Chapter 2: Installing the MS-DRG Software

Downloading and installing the Medicare Severity Diagnosis Related Groups (MS-DRG) Software (the grouper) consists of three steps:

1. Downloading and unzipping the file to your local machine
2. Allocating and FTPing the files to the mainframe
3. Link-editing the Assembler subroutines and testing the grouper

The first two steps are discussed in this chapter.

Step 3, the procedure for link-editing and testing the programs, depends on the way in which the grouper is implemented at your installation. Chapter 3 (page 37) explains how to use the grouper utility and contains JCL for using it to run the test database. Chapter 4 (page 45) shows how to link-edit the grouper programs for use as subroutines for a higher-level language. Two COBOL programs using the test database are included on the media, and the JCL for using them to test the installation is included in chapter 4.

The content of the downloaded file folder is shown in the following table.

Table 8. MS-DRG media contents

File	File name	LRECL	BLKSIZE	Description
1	OBJLIB	80	27920	Object library
2	SRCLIB	80	27920	Source library
3	LOADLIB	0	6233	Load library

The content of the miscellaneous folder is shown in the following table.

Table 9. MS-DRG miscellaneous folder contents

File	File name	LRECL	BLKSIZE	Description
1	TESTDB	960	27840	Test database
2	MDCDSC	80	27920	MDC titles
3	DRGDSC3	80	27920	DRG titles (3-digit)
4	DRGDSC4	85	27965	DRG titles (4-digit)
5	DRGLOGIC	150	27900	DRG logic
6	DXATTLST	100	27900	Diagnosis attribute list
7	DXLIST	20	27980	Diagnosis list
8	DXPATERN	60	27960	Diagnosis pattern

File	File name	LRECL	BLKSIZE	Description
9	EXCLUSN	15	27990	Exclusion
10	PRATTLST	120	27960	Procedure attribute list
11	PRCLSTRS	25	27975	Procedure clusters
12	PRLIST	20	27980	Procedure list
13	PRPATERN	150	27900	Procedure pattern

eDownload instructions

This section contains instructions for downloading program files from the Internet or from a CD for the Medicare Severity Diagnosis Related Groups (MS-DRG) Software.

Grouper program installation

All required software for executing the MS-DRG grouper is contained in the folders in this directory.

This directory contains the following folders:

- Load library - MS-DRG grouper load modules
- Object library - MS-DRG grouper object modules
- Source library - MS-DRG grouper source programs
- Miscellaneous
  - Test database file
  - MS-DRG grouper tables
  - DRG and MDC descriptions

Load library

The load library is a sequential file, FTPLOAD.

The load library consists of the load modules for the MS-DRG Grouper. The entire load library is optional if you intend to use the object modules.

1. Pre-allocate a sequential dataset on your minframe to receive the file using the following file characteristics:

- DSN = [e.g. YOURID.GROUPER.FTPLOAD]
- RECFM = FB
- LRECL = 80
- BLKSIZE = 3120
- SPACE = (CYL(12,1),RLSE)

2. FTP in BINARY mode the FTPLOAD file into the sequential dataset you allocated above.

   Important! You must FTP the load module files in BINARY.

3. Pre-allocate a load library PDS on the mainframe using the following file characteristics:
   - DSN = [e.g. YOURID.GROUPER.LOADLIB]
   - RECFM = U
   - BLKSIZE = 6233
   - SPACE = (CYL(15,3,2),RLSE)
4. Create a BUILDPDS JCL member as follows:
   - Add your JOBCARD
   - Modify dataset names as necessary
     - INDATASET = sequential dataset that was FTP'd to the mainframe in the step above.
     - DATASET = pre-allocated load library PDS that was created in the step above.

   Note: This JCL executes the utility, IKJEFT01, a terminal monitor program that executes the TSO commands via batch processing. This will populate the LOAD LIBRARY from the FTP'd load sequential file. A copy is shown below.

```
//JOB CARD FOR YOUR INSTALLATION
//             ************************************************************
//* *** RECEIVE FTP'D SEQUENTIAL FILES TO CREATE LOAD LIBRARY PDS ***
//             *************************************************************
//BDLOAD EXEC PGM=IKJEFT01
//SYSTSPRT            DD SYSOUT=*
//SYSTSIN             DD *
  RECEIVE             INDATASET ('YOURID.GROUPER.FTPLOAD')
                      DATASET ('YOURID.GROUPER.LOADLIB')
/*
```

5. After you modify the BUILDPDS, execute the JCL.

Table 10. Load library contents

Number	Name	Description
1	ALTTEST	Sample COBOL program (alternate interface)
2	COBTEST	Sample COBOL (standard interface) program

Number	Name	Description
3	DT300CA	Control program (alternate interface)
4	DT300CN	Control program (standard interface)

Object library

This information is for the object library. This directory contains an object module folder.

Table 11. Object library contents

Program	Description
DT300CN	The main control program (standard interface)
DT300GR	The grouper program
DT300RT	The grouper tables
DT300CA	The main control program (alternate interface)
DT300UT	The grouper utility interface

The first three programs (DT300CN, DT300GR, DT300RT) comprise the main grouper executor using the standard interface. Substitute DT300CA for DT300CN (that is, use DT300CA, DT300GR, and DT300RT) to compile the main grouper executor using the alternate (re-entrant, macro-free) interface. DT300UT is a utility program that can serve as an interface if your input data meets specific criteria. Chapter 3 (page 37) discusses this program.

All of the programs contained on the media were written in IBM Basic Assembler. There may be some reprogramming involved for those installations that do not have IBM equipment. The source code for each of the programs is provided.

Important! Object module files must be FTP'd in BINARY.

The following steps download the object library.

1. Allocate a PDS on your mainframe with the following characteristics:
   - DSN = [e.g. YOURID.GROUPER.OBJLIB]
   - RECFM = FB
   - LRECL = 80
   - BLKSIZE = 27920
   - SPACE = (CYL(10,1,2),RLSE)
2. FTP in **BINARY mode** all of the files in the object library folder into the PDS allocated in step 1.

Source library

There are several datasets included on the media that are not needed for the grouping process but may be useful to grouper users.

The folder contains the source library for all the grouper programs, tables, and the COBOL test programs. The library contains 7 members, as listed in the following table.

Table 12. Source library contents

Program	Description
DT300CN	The main control program (standard interface)
DT300GR	The grouper program
DT300RT	The grouper tables
DT300UT	The grouper utility interface program
DT300CA	The main control program (alternate interface)
COBTEST	The COBOL test interface program (standard interface)
ALTTEST	The COBOL test interface program (alternate interface)

Comments are also included in the source programs, DT300CN and DT300UT, describing the modifications needed to convert the programs to VSE.

The following steps are required to FTP the source library to the mainframe.

1. Allocate a PDS on your mainframe with the following characteristics:
   - DSN = [e.g. YOURID.GROUPER.SRCLIB]
   - RECFM = FB
   - LRECL = 80
   - BLKSIZE = 27920
   - SPACE = (CYL(21,1,4),RLSE)
2. FTP in ASCII mode all of the files in the source library folder into the PDS allocated in step 1.

Miscellaneous files installation

Test Database File

The following steps load the test database file to the mainframe.

1. Allocate a sequential file (PS) on your mainframe using the attributes below. It is also shown in the SYSUT2 DD card in JCL library member **DBLOAD**.
   - DSN=YOURID.GROUPER.**TESTDB**
   - RECFM=FB
   - LRECL=960
   - BLKSIZE=27840
   - SPACE=(CYL,(10,1),RLSE)
2. FTP the TESTDB file in ASCII mode from the miscellaneous folder to the mainframe, YOURID.GROUPER.**TESTDB**.

The following table provides a record description for the test database.

Table 13. Record layout for grouper test database

Field	Location	Name	Description
1	1-3	AGE	Age on admission, in years
2	4-4	SEX	Gender
3	5-6	DSP	Discharge status (disposition)
4	7-7	POALOG	POA logic indicator
5	8-15	ADATE	Admission date (YYYYMMDD)
6	16-23	DDATE	Discharge date (YYYYMMDD)
7	24-223	DX1-25	Diagnosis codes (DX1=Principal)
32	224-398	PROC1-25	Procedure codes
57	399-598	PRDATES (1-25)	Procedure dates (YYYYMMDD)
82	599-600	RTC	Return code from the grouper
83	601-602	MDC	MDC number returned by the grouper
84	603-606	DRG	Final DRG number returned by the grouper
85	607-611	GRFLGS	Output grouper flags
86	612-761	DXFLGS	Output diagnosis flags (25x6)
111	762-936	PRFLGS	Output procedure flags (25x7)
136	937-960	BUFF	Output ancillary buffer

Note: All diagnoses are left-justified in the first seven characters of an eight-character field, with each diagnosis' POA indicator occupying the eighth character of the field. Procedures are left-justified in seven-character fields. Unused characters in the diagnosis and procedure fields must be blank.

MDC Description File

The following steps load the MDC description file to the mainframe.

1. Allocate a sequential file (PS) on your mainframe using the attributes below.
   - DSN=YOURID.GROUPER.**MDCDSC**
   - RECFM=FB
   - LRECL=80
   - BLKSIZE=27920
   - SPACE=(TRK,(1,2),RLSE)
2. FTP the MDCDSC file in ASCII mode from the miscellaneous folder to the mainframe YOURID.GROUPER.**MDCDSC**.

Table 14. Record layout for MDCDSC

Column	Description
1-2	MDC number
3-3	Comma (,)
4-80	MDC title

DRG Description File

The following steps load the DRGDSC3 description file to the mainframe.

1. Allocate a sequential file (PS) on your mainframe using the attributes below.
   - DSN=YOURID.GROUPER.**DRGDSC3**
   - RECFM=FB
   - LRECL=80
   - BLKSIZE=27920
   - SPACE=(TRK,(1,2),RLSE)
2. FTP the DRGDSC3 file in ASCII mode from the miscellaneous folder to the mainframe YOURID.GROUPER.**DRGDSC3**.

Table 15. Record layout for DRGDSC3

Column	Description
1-3	DRG number

Column	Description
4-4	Comma (,)
5-7	Constant ‘MDC’
8-8	Blank
9-10	MDC number
11-11	’M’ (medical) or ‘P’ (surgical)
12-12	Comma (,)
13-80	DRG title

DRG Description File

The following steps load the DRGDSC4 description file to the mainframe.

1. Allocate a sequential file (PS) on your mainframe using the attributes below.
   - DSN=YOURID.GROUPER.**DRGDSC4**
   - RECFM=FB
   - LRECL=85
   - BLKSIZE=27965
   - SPACE=(TRK,(1,2),RLSE)
2. FTP the DRGDSC4 file in ASCII mode from the miscellaneous folder to the mainframe YOURID.GROUPER.**DRGDSC4**.

Table 16. Record layout for DRGDSC4

Column	Description
1-4	DRG number
5-5	Comma (,)
6-85	DRG title

ICD-10-CM/PCS MS-DRG Grouper Tables

Under ICD-9-CM, the MS-DRG Grouper Tables were included with the distribution of the mainframe MS-DRG grouper and were collectively referred to as the “EBCDIC Tables”. Though developed in ASCII on non-mainframe computers, the tables were necessarily converted to the EBCDIC character representation of mainframes in order to be included in the mainframe distribution.

Whether expressed in ASCII or EBCDIC, the ICD-10-CM/PCS MS-DRG Grouper Tables consist of nine files, as documented below. One of these, *drglogic*, specifies, via a non-platform-specific pseudo-language, the decisions the grouper makes to determine the MS-DRG from attributes of the patient and the codes on the patient record. All of the others are comma-delimited tables relating codes to attributes.

In the documentation of the comma-delimited files below, the information between successive commas is called a field. Field 1 is the information before the first comma, field 2 is that between the first and second comma, and so on. Information itself containing a comma will be enclosed in double quotation marks. No fields so enclosed are allowed themselves to contain double quotation marks.

Attributes

Since there are approximately 140,000 ICD-10-CM/PCS codes, with new ones being added every year, yet only a few codes on any given patient record, efficiency dictates that we transform the codes on the record into a small set of attributes, and then base the DRG-assignment logic on those attributes. An attribute is a patient-level yes/no result – either the patient record has the attribute or it does not. Hence in most implementations of the grouper, the attributes are represented with individual bits.

The grouper maintains five sets of attributes:

- Patient characteristics based on age, sex and discharge status, computed once when the patient data is provided.
- Principal diagnosis attributes, obtained by looking up the first listed diagnosis and saving its attributes from the diagnosis tables.
- Secondary diagnosis attributes, obtained by looking up each diagnosis on the record after the first, and combining the attributes found in the diagnosis tables using a Boolean inclusive-OR.
- “Any” diagnosis attributes, obtained via a Boolean inclusive-OR of the principal diagnosis attributes and the secondary diagnosis attributes.
- Procedure attributes, obtained by looking up each procedure on the record, and combining the attributes found in the procedure tables using a Boolean inclusive-OR for non-restricted procedures. Procedure clusters (see below) must also be discovered and their attributes included. This must be done each time the working MDC changes since procedure restriction is MDC dependent.

Procedure clusters

When found on a patient record, some collections of ICD-10-PCS procedure codes have a different set of attributes, independent of those of the codes that make them up (their “components”). These are called procedure clusters. A routine in the grouper, upstream of the DRG assignment logic, searches the patient record for clusters. (Multiple clusters, and duplicates of the same cluster, are possible.) When a cluster is found, it is added to the list of procedures found on the record. Clusters may be “restricted” by MDC. A restricted cluster inhibits the use of its component attributes for the MDC’s DRG assignment logic.

For example, cluster 0SRT0JZ+0SPC0JZ may be recognized on the patient record if both codes appear (in any order and not necessarily together). This creates a new “procedure code” @0045. The cluster @0045 has a different set of attributes than either 0SRT0JZ or 0SPC0JZ, and is further “restricted” for MDC 08. If the grouper logic determines that the MDC is 08, it ignores the attributes of 0SRT0JZ and 0SPC0JZ and only uses those of @0045. These take the grouper to DRGs 466-468 rather than DRG 463-465. If the PDX were not for MDC 08, however, the cluster would not restrict the individual interpretation of the component codes and their own attributes could come into play as well as those of @0045.

Grouper logic (drglogic)

The decisions that the grouper makes to assign a patient discharge to a DRG are specified in the file *drglogic*. The specifications resemble a computer language like C, C# or Java, but with a few special conventions to keep the file small:

- Two slashes (//) introduce a comment, which continues to the end of the line.
- A key at the beginning of the file explains the attribute notation.
- Logic introduced by “{“ continues until the matching “}” is reached.
- A statement of the form “return MDC|DRG|BASEDRG=3|153|152” means to assign the values on the right of the equal sign respectively to the variables on the left, then return to the main loop. In this example, MDC would be set to 3, DRG to 153 and BASEDRG to 152 before returning.

Diagnosis attributes (dxattlst)

Each row of the table identifies a diagnosis attribute potentially used by the grouper logic.

Table 17. Diagnosis attributes

Field	Contents
1	Attribute number. Bits in mainframe implementation are stored left-to-right in this order.
2	Attribute name as used in *drglogic*
3	Attribute description.

Diagnosis table (dxlist)

Each valid ICD-10-CM diagnosis code is found in the table.

Table 18. Diagnosis table

Field	Contents
1	Diagnosis code. ICD-10-CM codes are represented without their decimal point.
2	0 = fields 3 and 4 relate to both sexes. 1 = fields 3 and 4 relate to male patients only. 2 = fields 3 and 4 relate to female patients only.
3	The "exclusion category" to be used to find CC/MCC exclusions when this diagnosis is used as a PDX (principal diagnosis). If 0, then no CC or MCC is excluded. If 1, then only the diagnosis itself is excluded. For other values, see *exclusn*.
4	The pattern number of the row in *diagnosis_patterns.txt* which gives the attributes of this diagnosis.

Diagnosis pattern table (dxpatern)

Each distinct combination of MDC, diagnosis category and attributes has its own row. The pattern number is arbitrary and is used only as a link between *dxlist* and this table.

Table 19. Diagnosis pattern table

Field	Contents
1	Pattern number
2	MDC
3	Diagnosis category (DXCAT in *drglogic*).
4	HAC group. 0 if diagnosis is not in a HAC group
5	List of attributes, separated by the vertical bar (|)

Procedure attributes (prattlst)

Each row of the table identifies a procedure attribute potentially used by the grouper logic.

Table 20. Procedure attributes

Field	Contents
1	Attribute number. Bits in mainframe implementation are stored left-to-right in this order.
2	Attribute name as used in *drglogic*
3	Attribute description.

Procedure table (prlist)

Each valid ICD-10-PCS procedure code occupies one row in the table.

Table 21. Procedure table

Field	Contents
1	Procedure code or cluster. Procedure clusters are identified by an "@" sign, followed by a 4-digit number. This is a "cluster ID", which will match the cluster ID field in *prclstrs*.
2	The pattern number of the row in *prpatern* which gives the attributes of this procedure or procedure cluster.

Procedure pattern table (prpatern)

Each distinct combination of procedure attributes has its own row. The pattern number is arbitrary and is used only as a link between *prlist* and this table.

Table 22. Procedure pattern table

Field	Contents
1	Pattern number
2	List of procedure attributes, separated by the vertical bar (|). The special attribute "incluster" indicates that the procedure is in at least one cluster.

Procedure cluster definitions (prclstrs)

Each component of each distinct procedure cluster will have a row in this table.

Table 23. Procedure cluster definitions

Field	Contents
1	Procedure code.
2	Cluster ID of one cluster the procedure is a component of.
3	The choices group for the cluster this procedure is a member of. For example, "1" means that this procedure is one of the first set of procedures listed for the cluster, a "2" means it is one of the second set of procedures listed for the cluster. A cluster may have only one procedure in a choices set. The procedures in the cluster (its components) may be recognized on the patient record in any order and need not be listed together.
4	The number of procedures required to recognize the cluster. For example, a "3" means that the cluster is recognized if there is at least one procedure on the record from its first set, one from its second and one from its third.
5	The list of MDCs for which the cluster is restricted, separated by the vertical bar (|). "ALL" signifies that it is restricted for all MDCs.

CC/MCC exclusions (exclusn)

A principal diagnosis (PDX) may exclude certain secondary diagnoses (SDX) from being considered complications/co-morbidities (CC) and/or major complications/co-morbidities (MCC). The diagnosis table in *dxlist* gives an "exclusion category" for each PDX. A zero indicates that the diagnosis has no CC/MCC exclusions. A one indicates that the diagnosis excludes only itself. All numbers 2 and over refer to the first field of this table. All SDX listed in this table with a given exclusion category are prevented from being considered CCs or MCCs when the PDX has the exclusion category listed.

Table 24. CC/MCC exclusions

Field	Contents
1	Exclusion category
2	Excluded secondary diagnosis. ICD-10-CM codes are listed without their decimal point.

Uploading grouper tables/files to the mainframe

DRG logic file

The following steps load the DRG logic file to the mainframe.

1. Allocate a sequential dataset using the following attributes:
   - DSN=YOURID.GROUPER.**DRGLOGIC**
   - LRECL=150
   - BLKSIZE=27900
   - RECFM=FB
   - SPACE=(CYL(1),RLSE)
2. FTP the DRGLOGIC file from the miscellaneous folder in ASCII mode into a mainframe sequential dataset, "YOURID.GROUPER.**DRGLOGIC**".

Diagnosis attribute list

The following steps load the Diagnoses attribute list to the mainframe.

1. Allocate a sequential dataset using the following attributes:
   - DSN=YOURID.GROUPER.**DXATTLST**
   - LRECL=100

    - BLKSIZE=27900
    - RECFM=FB
    - SPACE=(TRK(1),RLSE)
2. FTP the DXATTLIST file from the miscellaneous folder in ASCII mode into a mainframe sequential dataset, “YOURID.GROUPER.DXATTLST”.

Diagnosis list file

1. The following steps load the Diagnosis list file to the mainframe.
2. Allocate a sequential dataset using the following attributes:
    - DSN=YOURID.GROUPER.**DXLIST**
    - LRECL=20
    - BLKSIZE=27980
    - RECFM=FB
    - SPACE=(CYL(2),RLSE)
3. FTP the DXLIST file from the miscellaneous folder in ASCII mode into a mainframe sequential dataset, “YOURID.GROUPER.DXLIST”.

Diagnosis pattern list

The following steps load the Diagnosis pattern list to the mainframe.

1. Allocate a sequential dataset using the following attributes:
    - DSN=YOURID.GROUPER.**DXPATERN**
    - LRECL=60
    - BLKSIZE=27960
    - RECFM=FB
    - SPACE=(TRK(1),RLSE)
2. FTP the DXPATERN file from the miscellaneous folder in ASCII mode into a mainframe sequential dataset, “YOURID.GROUPER.DXPATERN”.

CC/MCC exclusions file

The following steps load the CC/MCC exclusions file to the mainframe.

1. Allocate a sequential dataset using the following attributes:
    - DSN=YOURID.GROUPER.**EXCLUSN**

- LRECL=15
- BLKSIZE=27990
- RECFM=FB
- SPACE=(TRK(50),RLSE)

2. FTP the EXCLUSN file from the miscellaneous folder in ASCII mode into a mainframe sequential dataset, “YOURID.GROUPER.EXCLUSN”.

Procedure attribute list

The following steps load the Procedure attribute list to the mainframe.

1. Allocate a sequential dataset using the following attributes:
   - DSN=YOURID.GROUPER.**PRATTLST**
   - LRECL=120
   - BLKSIZE=27960
   - RECFM=FB
   - SPACE=(TRK(1),RLSE)
2. FTP the PRATTLIST file from the miscellaneous folder in ASCII mode into a mainframe sequential dataset, “YOURID.GROUPER.PRATTLST”.

Procedure list file

The following steps load the Procedure list file to the mainframe.

1. Allocate a sequential dataset using the following attributes:
   - DSN=YOURID.GROUPER.**PRLIST**
   - LRECL=20
   - BLKSIZE=27980
   - RECFM=FB
   - SPACE=(CYL(2),RLSE)
2. FTP the PRLIST file from the miscellaneous folder in ASCII mode into a mainframe sequential dataset, “YOURID.GROUPER.PRLIST”.

Procedure pattern list

The following steps load the Procedure pattern list to the mainframe.

1. Allocate a sequential dataset using the following attributes:

- DSN=YOURID.GROUPER.**PRPATERN**
- LRECL=150
- BLKSIZE=27900
- RECFM=FB
- SPACE=(TRK(5),RLSE)

2. FTP the PRPATERN file from the miscellaneous folder in ASCII mode into a mainframe sequential dataset, “YOURID.GROUPER.PRPATERN”.

Procedure clusters file

The following steps load the Procedure clusters file to the mainframe.

1. Allocate a sequential dataset using the following attributes:
   - DSN=YOURID.GROUPER.**PRCLSTRS**
   - LRECL=25
   - BLKSIZE=27975
   - RECFM=FB
   - SPACE=(TRK(1),RLSE)
2. FTP the PRCLSTRS file from the miscellaneous folder in ASCII mode into a mainframe sequential dataset, “YOURID.GROUPER.PRCLSTRS”.

Chapter 3: Using and testing the grouper utility

Installations with data that conforms to the grouper requirements provided in chapter 1 (see "Data format requirements" on page 9) and whose output record length does not exceed 2992 bytes, may implement the grouper as a utility program that receives all information pertaining to the input record layout from the job's SYSIN stream. To use the grouper utility, you must have copied file 1 (the grouper object library) from the media to disk (see "Installing the MS-DRG Software" on page 19).

Link-editing the grouper utility

The sample JCL for creating a load module for the grouper utility is shown in the following figure.

```
//JOB CARD FOR YOUR INSTALLATION
//* *******************************************************
//* THIS JOB CREATES A GROUPER UTILITY LOAD MODULE   *
//* *******************************************************
//LKED   EXEC  PGM=HEWL,PARM='LIST,MAP,AMODE=31,RMODE=ANY',
//    REGION=1024K
//SYSLMOD  DD  DSN=GROUPER.UTIL.LOAD,DISP=OLD
//SYSUT1   DD  UNIT=DISK,SPACE=(CYL,(1,1))
//SYSPRINT DD  SYSOUT=*
//OBJECT   DD  DSN=GROUPER.OBJLIB,DISP=OLD
//SYSLIN DD *
  INCLUDE OBJECT(DT300UT,DT300CN,DT300GR,DT300RT)
  ENTRY DT300UT
  NAME DT300UT
/*
```

Using the grouper utility

As previously mentioned, the grouper utility receives all information pertaining to the input and output record layouts from the job's SYSIN stream. When using the grouper utility, you must provide 16 SYSIN control statements shown in the following table. These statements must be present in the order shown. Each control statement consists of a 3-character keyword followed by at least one 4-digit field, right-justified and zero-filled, indicating the starting position of the variable.

Table 25. Control statements required by the grouper utility

Control statement	Keyword	Identifies the starting position(s) of...
1	DDX	Each 8-byte diagnosis code

Control statement	Keyword	Identifies the starting position(s) of...
2	SRG	Each 7-byte procedure code
3	AGE	The age field
4	SEX	The sex field
5	DSP	The discharge status field
6	POA	Present on admission logic
7	ADT	Admission date
8	DDT	Discharge date
9	SDT	Procedure dates
10	RTC	The grouper return code
11	MDC	The MDC number returned by the grouper
12	DRG	The DRG number returned by the grouper
13	GFL	Grouper flags
14	DFL	Diagnosis flags
15	SFL	Procedure flags
16	BUF	Grouper buffer

Control statement examples

The following examples of the control statements use the 960-byte record from the test database as input. The first 598 bytes contain the data that must be passed to the grouper, and the next 362 bytes contain the information filled in by the previous grouper. The output record is 362 bytes larger, with those 362 bytes containing the data returned by the new grouper when you run the test.

The discharge diagnosis control statement (DDX)

The DDX control statement specifies the starting position of each discharge diagnosis code in the patient record to be used in the grouping process. Blanks must be inserted between each position specified. The grouper assumes that the first specified diagnosis is the principal discharge diagnosis. You may specify up to 24 secondary diagnoses to be considered in the grouping process so there may be at most 25 diagnosis positions specified on the control statement. For example, the DDX control statement shown below indicates that the principal diagnosis started at position 24 and that there were 24 secondary diagnoses to be used by the grouper, which began at position 32.

The grouper assumes that each diagnosis code specified is left-justified in a ***8-byte field***. All codes must be blank-filled. Zero-filled codes are not allowed. The 8th byte in each field is the POA indicator.

```
Contents  DDX 0024 0032 0040 0048 0056 0064 0072 0080 0088 0096 0104 0112 0120 0128 0136 *
          DDX 0144 0152 0160 0168 0176 0184 0192 0200 0208 0216
```

When there are more than 15 diagnoses, the asterisk (*) must be placed in column 80.

The procedure control statement (SRG)

The SRG control statement specifies the starting position of each procedure code in the patient record to be used in the grouping process. As with the diagnosis control statement, you specify each starting position as a 4-digit number. Blanks must be inserted between each position specified. You may provide up to 25 procedures for use by the grouper. For example, the SRG control statement shown below indicates that there were 25 procedure codes to be used in the grouping process, with the first procedure beginning at position 224, the second procedure beginning at position 231, and so on.

The grouper assumes that each procedure code specified is left-justified in a ***7-byte field***. Short codes must be blank-filled. Zero-filled codes are not allowed.

```
Contents  SRG 0224 0231 0238 0245 0252 0259 0266 0273 0280 0287 0294 0301 0308 0315 0322 *
          SRG 0329 0336 0343 0350 0357 0364 0371 0378 0385 0392
```

When there are more than 15 procedures, the asterisk (*) must be placed in column 80.

The age control statement (AGE)

The AGE control statement specifies the starting position of the field containing the patient age. Only ages between 0 and 124 are considered valid for grouping. The age field is assumed to be three bytes in length, containing right-justified numerics, and may be either zero- or blank-filled. For example, the AGE control statement displayed below indicates that the 3-byte age field appears on the patient record starting at position 1.

```
Column         123456789
Contents       AGE 0001
```

The sex control statement (SEX)

The SEX control statement specifies the starting position of the field containing the patient’s sex. The grouper assumes that the sex field is one byte in length, containing the values 0 through 2 (un-known/male/female respectively). The test database SEX control statement is:

```
Column        123456789
Contents      SEX 0004
```

The discharge status control statement (DSP)

The DSP control statement specifies the position of the discharge status on the patient’s record. The grouper assumes this is a 2-byte, right-justified field, with values as specified in the "Required data formats" table (page 9). Short codes (i.e., codes with fewer than two digits) may be either blank- or zero-filled. The test database DSP control statement is:

```
Column        123456789
Contents      DSP 0005
```

The present on admission control statement (POA)

The POA control statement specifies the starting position of the field containing the Present on Admission logic flag. The grouper assumes the POA flag is one byte in length, containing the values specified in the "Required data formats" table (page 9). The test database control statement is:

```
Column        1234567890
Contents      POA 0007
```

The admission date control statement (ADT)

The ADT control statement specifies the starting position of the field containing the patient's admission date. The grouper assumes the admission date is 8 bytes in length, formatted as YYYYMMDD. The test database control statement is:

```
Column        1234567890
Contents      ADT 0008
```

The discharge date control statement (DDT)

The DDT control statement specifies the starting position of the field containing the patient's discharge date. The grouper assumes the discharge date is 8 bytes in length, formatted as YYYYMMDD. The test database control statement is:

```
Column        1234567890
Contents      DDT 0016
```

The procedure dates control statement (SDT)

The SDT control statement specifies the starting position of a 200-byte buffer containing the date of each procedure coded on the patient record. The grouper assumes each procedure date is 8 bytes in length, formatted as YYYYMMDD. The test database control statement is:

```
Column        1234567890
Contents      SDT 0399
```

Grouper output control statements

It is important to note that none of the data returned by the grouper needs be written to the output record, although presumably you would want at least DRG and MDC numbers and the grouper return code (RTC). Regardless of whether you choose to output the data or not, a control statement with an output position must be supplied for each of the elements specified below (RTC, MDC, DRG, GFL, DFL, SFL, BUF).

You must ensure that the storage for all fields returned by the grouper can be contained on the output record. The utility program determines the output record length from the JCL DCB specifications for the output dataset. If the position specified is beyond the end of the output record but within the maximum record length allowed, the field is dropped when the output record is written.

The return code control statement (RTC)

The RTC control statement specifies the location of a 2-byte field, which is used to store the grouper return code. The test database return code control statement is:

```
Column        123456789
Contents      RTC 0961
```

The MDC control statement (MDC)

The MDC control statement specifies the starting position for the storage of the MDC number returned by the grouper. The MDC number is a 2-byte, right-justified numeric value. The test database MDC control statement is:

```
Column        123456789
Contents      MDC 0963
```

The DRG control statement (DRG)

The DRG control statement specifies where on the output record the grouper should store the Final DRG number. The Final DRG number returned by the grouper is a 4-byte, right-justified numeric value. The test database DRG control statement is:

```
Column        123456789
Contents      DRG 0965
```

The grouper flags control statement (GFL)

The GFL control statement specifies the starting position of the grouper flags. The grouper assumes this field to be 5 bytes in length. The test database control statement is:

```
Column        1234567890
Contents      GFL 0969
```

The diagnosis flags control statement (DFL)

The DFL control statement specifies the starting position of the diagnosis flags. The grouper assumes this field to be 150 bytes in length. There are 6 diagnosis flags for each diagnosis on the record, up to a total of 25 diagnosis codes. The test database control statement is:

```
Column        1234567890
Contents      DFL 0974
```

The procedure flags control statement (SFL)

The SFL control statement specifies the starting position of the procedure flags. The grouper assumes this field to be 175 bytes in length. There are 7 procedure flags for each procedure on the record, up to a total of 25 procedure codes. The test database control statement is:

```
Column        1234567890
Contents      SFL 1124
```

The buffer control statement (BUF)

The BUF control statement specifies the starting position of the buffer of additional DRG information. The grouper assumes this field to be 24 bytes in length. The test database control statement is:

```
Column        1234567890
Contents      BUF 1299
```

Running the grouper utility program

The following table shows the ABENDs (abnormal end of jobs) possible from the grouper utility program.

Table 26. ABEND codes

Code	Description
80A	Insufficient region size
001	Control statements missing or out of order
002	Non numeric data in position field on control statement
003	Missing control statement
004	Unsuccessful open of input database
005	Unsuccessful open of output database
006	Continuation character (*) found with less than 15 codes

The following sample JCL is for executing the grouper utility program.

```
//GO  EXEC  PGM=DT300UT
//STEPLIB  DD  DSN=GROUPER.UTIL.LOAD,DISP=SHR
//IN  DD  DSN=GROUPER.TEST.DATA,DISP=SHR
//OUT  DD  DSN=GROUPER.OUTTEST.DATA,
//    DISP=(NEW,CATLG,DELETE),UNIT=SYSDA,
//    DCB=(LRECL=1322,BLKSIZE=27762,RECFM=FB),
//    SPACE=(CYL,(10,1),RLSE)
//SYSPRINT DD  SYSOUT=*,DCB=(RECFM=FA,BLKSIZE=133,BUFNO=1)
//SYSIN    DD  *
DDX 0024 0032 0040 0048 0056 0064 0072 0080 0088 0096 0104 0112 0120 0128 0136 *
DDX 0144 0152 0160 0168 0176 0184 0192 0200 0208 0216
SRG 0224 0231 0238 0245 0252 0259 0266 0273 0280 0287 0294 0301 0308 0315 0322 *
SRG 0329 0336 0343 0350 0357 0364 0371 0378 0385 0392
AGE 0001
SEX 0004
DSP 0005
POA 0007
ADT 0008
DDT 0016
SDT 0399
RTC 0961
MDC 0963
DRG 0965
GFL 0969
DFL 0974
SFL 1124
BUF 1299
```

Note: The SYSIN control statements must not contain line numbers, as the entire 80 bytes is considered input. Failure to do this causes User ABEND 001.

Chapter 4: Using the grouper with higher-level languages

The grouper executor may be implemented as a subroutine to be called from Assembler or a higher-level language program. This chapter shows how this may be done for a COBOL programming environment. To create the subroutines, you must have copied file 1 (the grouper objlib) from the media to disk (see "Installing the MS-DRG Software" on page 19).

General strategy for COBOL driving program

A typical COBOL grouping utility might operate as follows:

- Opens the input and output datasets
- Reads records from the input dataset
- Reformats and recodes the input data to a form acceptable to the grouper
- Calls the grouper
- Stores the grouper return information on the output record

Writes a new dataset containing the original data and the grouping information A COBOL program (COBTEST) using the sample database is included on the media.

The following sample JCL is for grouping test database in the COBOL environment.

```
//JOB CARD FOR YOUR INSTALLATION
//* *******************************************************
//* SAMPLE JCL FOR GROUPING TEST DATABASE IN THE COBOL    *
//* ENVIRONMENT.    *
//*    *
//* BOTH OBJECT AND LOAD MODULES ARE TEMPORARY.  *
//* *******************************************************
//COBUCLG PROC
//*   COBOL FOR MVS COMPILE AND LE370 LINK
//COB    EXEC  PGM=IGYCRCTL,PARM='RENT,NODYNAM'
//STEPLIB  DD  DSN=IGYV3R4.SIGYCOMP,DISP=SHR
//SYSLIB   DD  DSN=GROUPER.SRCLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  DSN=GROUPER.SRCLIB(COBTEST),DISP=SHR
//SYSUT1   DD  UNIT=DISK,SPACE=(CYL,(1,1))
//SYSUT2   DD  UNIT=DISK,SPACE=(CYL,(1,1))
//SYSUT3   DD  UNIT=DISK,SPACE=(CYL,(1,1))
//SYSUT4   DD  UNIT=DISK,SPACE=(CYL,(1,1))
//SYSUT5   DD  UNIT=DISK,SPACE=(CYL,(1,1))
//SYSUT6   DD  UNIT=DISK,SPACE=(CYL,(1,1))
//SYSUT7   DD  UNIT=DISK,SPACE=(CYL,(1,1))
//SYSLIN   DD  DSN=&&LOADSET,UNIT=DISK,DISP=(MOD,PASS),
//    SPACE=(TRK,(3,3)),DCB=BLKSIZE=800
//*
//LKED   EXEC  PGM=IEWL,PARM='LIST,MAP,AMODE=31,RMODE=ANY',
//    COND=(5,LT,COB)
//SYSLIB   DD  DSN=CEE.SCEELKED,DISP=SHR
//SYSLMOD  DD  DSN=&&GOSET(GO),DISP=(,PASS),UNIT=DISK,
//    SPACE=(CYL,(5,1,5))
//SYSUT1   DD  UNIT=DISK,SPACE=(CYL,(1,1))
//SYSPRINT DD  SYSOUT=*
//SYSLIN   DD  DSN=&&LOADSET,DISP=(OLD,DELETE)
//    DD  DDNAME=SYSIN
//OBJECT   DD  DSN=GROUPER.OBJLIB,DISP=OLD
//*
//GO  EXEC  PGM=COBTEST,COND=((5,LT,COB),(5,LT,LKED))
//STEPLIB  DD DISP=SHR,DSN=CEE.SCEERUN
//    DD DISP=SHR,DSN=&&GOSET
//SYSPRINT DD SYSOUT=*
//INFILE   DD DSN=GROUPER.TEST.DATA,DISP=SHR
//    PEND
//*
//PROG1    EXEC COBUCLG,PROD=DRGT300
//LKED.SYSIN DD *
  INCLUDE OBJECT(DT300CN,DT300GR,DT300RT)
  ENTRY COBTEST
  NAME COBTEST
/*
```

Input to the grouper subroutines

The grouper control program (DT300CN) assumes that general purpose register 1 is pointing to a list of addresses with the structure shown in the following table.

Table 27. MS-DRG software address list

Offset	Fullword pointer to...
0	The buffer containing the ICD-10-CM diagnosis codes for the record to be grouped. The first code is assumed to be principal diagnosis.
4	4-byte binary (PIC 9(8) COMP) field indicating the number of diagnoses contained in the buffer discussed above. This can be the actual number of codes in the buffer, or the maximum number of codes that the buffer can hold. This number cannot be less than 1 nor greater than 25. If greater than 25, the software uses only the first 25 fields in the buffer and ignores the rest.
8	The buffer containing the procedure codes for the record to be grouped.
12	4-byte binary (PIC 9(8) COMP) field indicating the number of procedures present. This field has the same rules as for diagnoses, except that it may be zero.
16	3-byte numeric field containing the patient's age in years.
20	1-byte numeric field containing the patient's sex.
24	2-byte numeric field containing the patient's discharge status.
28	1-byte field containing the POA logic indicator
32	8-byte numeric field containing the patient's admission date (YYYYMMDD)
36	8-byte numeric containing the patient's discharge date (YYYYMMDD)
40	200-byte buffer containing the dates of the procedure codes. The buffer can hold up to a maximum of 25 dates, 8-bytes each (YYYYMMDD).
44	2-byte numeric field to hold the grouper return code.
48	2-byte numeric field to hold the MDC number.
52	4-byte numeric field to hold the DRG number.
56	5-byte field to hold the grouper flags.
60	150-byte field to hold the diagnosis flags.
64	175-byte field to hold the procedure flags.
68	24-byte field to hold the buffer of additional DRG information.

Note: COBOL applications programmers need not concern themselves with implementing this structure since COBOL automatically creates it when a CALL USING statement is issued.

You must ensure that each diagnosis code is left-justified in a 8-byte field and that all of the diagnoses are in contiguous locations in the buffer whose address is in the first pointer described above. Empty fields may be interspersed throughout the buffer. A detailed discussion of the way in which fields in the buffer are processed is located at the end of this chapter.

Similarly, each procedure code must be left-justified in a 7-byte field, and all of the procedure codes must be in contiguous locations in the buffer whose address is in the third pointer described above.

Each diagnosis and procedure code must be blank-filled if it is shorter than the maximum field length. *Zero filling is not allowed.*

The patient's age must be right-justified in a 3-byte field. Valid ages for grouping are between 0 and 124. The age may be either zero- or blank-filled.

The patient's sex must be contained in a 1-byte field, in the range 0 through 2 (Unknown/Male/Female, respectively).

The discharge status must be contained in a 2-byte field which is coded according to the conventions shown in the "Required data formats" table (page 9). The code must be right-justified and may be either zero- or blank-filled.

Output from the grouper subroutines

On return from the grouper executor, the DRG, MDC, return code, and the grouper, diagnosis, and procedure flags fields are filled in, along with the buffer of additional DRG information. The DRG and MDC numbers are right-justified. The grouper return code is filled in according to the conventions detailed in chapter 1 (page 9).

Using the alternate interface

The alternate grouper control program, (DT300CA) operates the same as the standard grouper control program (DT300CN) except that it does not contain any macros and is written to be re-entrant, so it should run in a wider variety of mainframe environments. Whereas the standard interface uses GETMAIN to obtain a 20,000 byte work area, the alternate interface requires that the calling program provide the work area. It must do so by providing two additional addresses in the list pointed to by general register 1. For details see the "MS-DRG software address list" table (page 46).

The following table gives the additional work area parameters required by the alternate interface.

Table 28. Work area parameters

Offset	Full word pointer to...
72	A buffer of at least 20,000 bytes.
76	4-byte binary (PIC 9(8) comp) field containing the actual length in bytes of the work area. The value of this field should not be less than 20,000 bytes, though larger values are acceptable.

To use the alternate interface, substitute DT300CA for DT300CN and provide these two extra parameters. See the COBOL program ALTTEST, provided in the source library, for an example of how to set up a work area and pass it to DT300CA.

Assembler programmers should note that the length of the work area is *not* given in the full word at the offset 76 from R1 but rather a *pointer* to the full word containing the length is given at offset 76.

Sample JCL for running ALTTEST may be created by modifying the JCL for grouping the test database in the COBOL environment (page 45). To modify the JCL, change all occurrences of COBTEST to ALTTEST and change DT300CN to DT300CA.

Executor processing of the diagnosis and procedure buffers

The way in which the grouper retrieves diagnosis and procedure codes for processing is to loop through the related buffers using the counts addressed by the second and fourth pointers. If any diagnosis or procedure field is all zeroes or all blanks, then that field is considered empty and the code is flagged as invalid and is ignored. Codes are saved in an internal work area that is subsequently used for construction of the record mask (see "The MS-DRG grouper executor" on page 51). Because processing is done this way, it is possible to pass a buffer that contains both valid and empty fields.

For example, assume there is a record containing a maximum of five diagnosis codes, three of which are coded for this abstract. The number of diagnoses passed would be five, and the buffer could look like any of the following:

```
3310   Y40210  Y5601   N
3310   Y       40210   Y5601   N
3310   Y00000  40210   Y5601   N00000
```

The principal diagnoses must be in the first field of the buffer. If the field is empty or invalid, the record is assigned DRG 999 (ungroupable) with a return code of 7 (invalid principal diagnosis).

Chapter 5: The MS-DRG grouper executor

To use the information in this chapter, you should have:

- A working knowledge of IBM Basic Assembler Language
- At least a rudimentary understanding of the underlying logic on which all DRG decisions are based
- Access to the *Medicare Severity Diagnosis Related Groups Definitions Manual*, which explains the principles on which all decisions are made

The executor essentially makes its decisions by comparing indicators for each DRG within an MDC. Indicators are set by the elements found on the patient record. These sets of indicators are referred to as masks. The content of the masks is listed in the MS-DRG grouper tables (page 27).

The tables are represented as hexadecimal constants in the module DT300RT and are present in memory when the grouper is loaded for execution. All table lookups are in-memory binary searches.

The executor begins its basic task by creating masks that are indicative of the conditions found on the patient record. These are called the record masks.

Once the record masks have been constructed, the corresponding DRG masks for the MDC indicated by the principal diagnosis are compared to them, until a match is found or the DRG masks for the MDC are exhausted.

Because the internal format of the grouper tables is optimized in DT300RT for fast lookups and is therefore difficult to read, the nine principal tables included in DT300RT are provided as flat files on the media. See chapter 2 (page 19) for table layout details.

Construction of the record mask

The following list describes how the executor constructs the record masks.

1. Sex is tested for validity (1-2).
   - An error indicator is turned on if sex is out of that range.
   - If not, the appropriate indicator is set in the record mask.
2. Discharge status is tested for validity (01-07, 20, 21, 30, 43, 50, 51, 61-66, 70)
   - An error indicator is turned on if discharge status is out of range.
   - Otherwise, the appropriate indicators are set in the record mask.
3. The first listed diagnosis (assumed principal) is looked up in the Diagnosis Table.
   - If no entry is found, the record is assigned DRG 999, RTC 7 and no further processing occurs.

- If an entry is found, but the MDC number is 0, the record is assigned DRG 999, RTC 7 and no further processing occurs.
- Otherwise, the MDC and DXCAT are saved and the indicators for this diagnosis code are moved to the mask where principal diagnosis indicators are positioned.

4. All secondary diagnoses are looked up in the Diagnosis Table and their bit indicators “OR’d” together in the mask reserved for secondary diagnosis indicators. Additionally, if any of the secondaries is a complication or comorbidity, the CC exclusion subroutine is called to determine if the CC flag in the record mask should be set. A complete discussion of the CC exclusion subroutine appears later in this chapter (page 53).

   Any secondary diagnosis for which there is no Diagnosis Table entry does not cause an error, but is instead ignored. MDC and DXCAT numbers are of no importance for secondaries.
5. Once all diagnoses have been processed, the indicators for principal and secondary are “OR’d” together in yet another indicator section mask for ALLDX criteria.
6. All procedure codes are looked up in Procedure Table and their bit indicators “OR’d” together in the mask reserved for procedure indicators. As with secondary diagnoses, invalid procedure codes do not generate errors, but are ignored.

DRG determination

Once the record masks have been constructed, the executor loops through the DRG masks for the MDC indicated by the principal diagnosis, comparing them with the record masks.

1. The comparison is done by moving the record mask to a work area and ANDing it with the current DRG mask.
2. The result of the ANDed work mask is then compared with the DRG mask.
   - If the results are identical, the associated DRG number is assigned and the processing to find and return the diagnosis and procedure flags is executed.
   - Otherwise, looping continues until a match is found or the DRG list is exhausted, at which time DRG 999 is assigned.

The rest of this section discusses some special conditions in the grouper logic.

Testing for the ONLY surgery condition

When the DRG mask indicates that ONLY specific surgeries can be present, the executor loops through the saved O.R. surgeries from the record, making decisions as follows:

1. The O.R. portion of the DRG mask is moved to a work area.
2. The work mask is ANDed with the mask of the saved O.R. surgery.

- If the result of the ANDing is zero, this indicates that the surgery found on the record is other than the ONLY surgery allowed. The executor ceases looping and gets the next DRG mask.
- Otherwise, the process continues until all saved O.R. surgeries have been tested.

Testing for the ONLY DX condition

The testing for this condition is virtually identical to that done for the ONLY surgery condition, except that the comparison is done on saved diagnoses against the ALLDX portion of the DRG mask.

Testing for the OWISE condition

This condition exists for DRGs 794, 963-965 and 997. This is essentially the “fall through” DRG for the MDC and is assigned when no other DRG criteria have been met. The “anydx” bit in the DRG mask is turned on, leaving a mask with only that bit on, thereby guaranteeing a match.

Testing for the ANYCOMB condition

This condition exists only for DRG 461-462 in MDC 8. The test is done by comparing all coded O.R. procedures with the procedure portion of the DRG mask and adding one to an accumulator for each procedure that has a matching mask. If the resulting count is less than two, this record does not meet the "anycomb" condition, and the next DRG mask is retrieved.

CC exclusion subroutine

A large subset of the diagnosis codes are flagged as complication/comorbidity codes (CC) or major complication/comorbidity codes (MCC). Many of these codes are not really CC/MCC codes at all times because there are many conditions for which the secondary diagnosis is a natural side effect of the principal diagnosis. The CC/MCC exclusion table is organized to reflect a direct relationship between a principal diagnosis and selected secondaries.

Because the ICD-10-CM codes are non-contiguous and do not lend themselves well to defining ranges of codes, an index number is associated with each diagnosis and the CC/MCC exclusion table is constructed entirely from those index numbers.

To determine whether a secondary should be considered a CC/MCC, the executor accesses the CC/MCC table, using the principal diagnosis CC/MCC exclusion category as the key each time a secondary flagged as CC/MCC is encountered.

- If no entry is found for the exclusion category, that means that there are no exclusions and the secondary is considered a CC/MCC code.
- If an entry is found, then the secondary is excluded as a CC/MCC.

Testing for the OTHOR condition

This test is similar in logic to the test for the ONLY conditions, except that it tests for procedures in addition to the O.R. criteria in the DRG mask. When the DRG mask indicates that other O.R. procedures must be present, the executor loops through the O.R. procedures from the record, making decisions as follows:

1. The O.R. portion of the DRG mask is moved to a work area.
2. The work mask is ANDed with the mask of the saved O.R. procedure.
   - If the result of the ANDing is zero, this indicates that the procedure is other than the specific procedure required (e.g., T&A) and therefore satisfies the other O.R. criteria. When that occurs, looping ceases and processing continues for the DRG.
   - Otherwise, the loop continues until a procedure satisfies the other condition. If all saved procedures are exhausted without finding one that satisfies the other condition, then processing for that DRG is ended.

Testing for illogical principal diagnosis

When a DRG has been matched, and the DRG number is 999, the cause is an illogical principal diagnosis. To indicate this, the return code is changed to 6.

Testing for multiple significant trauma

The principal diagnosis is tested to see if it is a trauma code. If it is, processing continues to test for multiple significant trauma. Otherwise, no further trauma testing is done.

To qualify as multiple significant trauma, two significant trauma codes from *different* body sites must be present. The diagnosis mask contains special trauma indicators, with each body site trauma represented by a different flag.

The mask of the first diagnosis (either principal or secondary) that is flagged as a significant trauma is saved. The mask of each subsequent diagnosis that is also flagged is compared with the initial saved mask. If they are not the same, the record is flagged as a multiple significant trauma episode. If they are the same, the next diagnosis is tested until the multiple condition is satisfied or the diagnoses are exhausted.

Finding codes that affect Initial DRG assignment

After the DRG has been determined, the grouper executor analyzes the saved diagnosis and procedure masks, comparing them against the masks which were used to determine MDC and DRG. Codes which were necessary for the determination of the MDC/DRG are flagged with an "affect flag."

Final DRG

If no Hospital Acquired Conditions (HACs) are found on the record, then the initial DRG becomes the final DRG. Otherwise, the record is re-grouped demoting the HAC secondary diagnosis which may or may not change the DRG assignment based on what DRG it was initially assigned to, and/or the presence of other codes that are CCs or MCCs.

Executor ABEND codes

There is one ABEND (abnormal end of job) code that can be generated by the executor, standard version only.

Table 29. ABEND codes generated by the executor–standard version

Code	Description
108	Not able to GETMAIN a work area of sufficient size.

The alternate interface does not contain any ABEND macros.

Appendix A. Grouping results for the test database

The following is a partial listing of the output produced by the grouper utility program (DT300UT). The program's printout is a distribution of record counts by final DRG, MDC, and return code (RTC), respectively. The test database used a POA indicator of Z. There were no POAs assigned to the diagnosis codes. The printout of counts from your test run may differ in appearance from what is shown in the appendix, but the content should be the same if the test is successful. Some editing was done in order to fit the text into this manual.

The test, when performed on an IBM Z10 2097-E26 (703) computer, used 192K of virtual storage, and took less than 1 CPU second.

```
COUNTS BY DRG
   1    5 |   51    0 |  101    5 |  151    5 |  201    5 |  251    5 |  301    5 |  351    5 |
   2    5 |   52    5 |  102    5 |  152    5 |  202    5 |  252    5 |  302    5 |  352    5 |
   3    5 |   53    5 |  103    5 |  153    5 |  203    5 |  253    5 |  303    5 |  353    5 |
   4    5 |   54    5 |  104    0 |  154    5 |  204    5 |  254    5 |  304    5 |  354    5 |
   5    5 |   55    5 |  105    0 |  155    5 |  205    5 |  255    5 |  305    5 |  355    5 |
   6    5 |   56    5 |  106    0 |  156    5 |  206    5 |  256    5 |  306    5 |  356    5 |
   7    5 |   57    5 |  107    0 |  157    5 |  207    5 |  257    5 |  307    5 |  357    5 |
   8    5 |   58    5 |  108    0 |  158    5 |  208    5 |  258    5 |  308    5 |  358    5 |
   9    0 |   59    5 |  109    0 |  159    5 |  209    0 |  259    5 |  309    5 |  359    0 |
  10    5 |   60    5 |  110    0 |  160    0 |  210    0 |  260    5 |  310    5 |  360    0 |
  11    5 |   61    5 |  111    0 |  161    0 |  211    0 |  261    5 |  311    5 |  361    0 |
  12    5 |   62    5 |  112    0 |  162    0 |  212    0 |  262    5 |  312    5 |  362    0 |
  13    5 |   63    5 |  113    5 |  163    5 |  213    0 |  263    5 |  313    5 |  363    0 |
  14    5 |   64    5 |  114    5 |  164    5 |  214    0 |  264    5 |  314    5 |  364    0 |
  15    0 |   65    5 |  115    5 |  165    5 |  215    5 |  265    5 |  315    5 |  365    0 |
  16    5 |   66    5 |  116    5 |  166    5 |  216    5 |  266    0 |  316    4 |  366    0 |
  17    5 |   67    5 |  117    5 |  167    5 |  217    5 |  267    0 |  317    0 |  367    0 |
  18    0 |   68    5 |  118    0 |  168    5 |  218    5 |  268    0 |  318    0 |  368    5 |
  19    0 |   69    5 |  119    0 |  169    0 |  219    5 |  269    0 |  319    0 |  369    5 |
  20    5 |   70    5 |  120    0 |  170    0 |  220    5 |  270    0 |  320    0 |  370    5 |
  21    5 |   71    5 |  121    5 |  171    0 |  221    5 |  271    0 |  321    0 |  371    5 |
  22    5 |   72    5 |  122    5 |  172    0 |  222    5 |  272    0 |  322    0 |  372    5 |
  23    5 |   73    5 |  123    5 |  173    0 |  223    5 |  273    0 |  323    0 |  373    5 |
  24    5 |   74    5 |  124    5 |  174    0 |  224    5 |  274    0 |  324    0 |  374    5 |
  25    5 |   75    5 |  125    5 |  175    5 |  225    5 |  275    0 |  325    0 |  375    5 |
  26    5 |   76    5 |  126    0 |  176    5 |  226    5 |  276    0 |  326    5 |  376    5 |
  27    5 |   77    5 |  127    0 |  177    5 |  227    5 |  277    0 |  327    5 |  377    5 |
  28    5 |   78    5 |  128    0 |  178    5 |  228    5 |  278    0 |  328    5 |  378    5 |
  29    5 |   79    5 |  129    5 |  179    5 |  229    5 |  279    0 |  329    5 |  379    5 |
  30    5 |   80    5 |  130    5 |  180    5 |  230    5 |  280    5 |  330    5 |  380    5 |
  31    5 |   81    5 |  131    5 |  181    5 |  231    5 |  281    5 |  331    5 |  381    5 |
  32    5 |   82    5 |  132    5 |  182    5 |  232    5 |  282    5 |  332    5 |  382    5 |
  33    5 |   83    5 |  133    5 |  183    5 |  233    5 |  283    5 |  333    5 |  383    5 |
  34    5 |   84    5 |  134    5 |  184    5 |  234    5 |  284    5 |  334    5 |  384    5 |
  35    5 |   85    5 |  135    5 |  185    5 |  235    5 |  285    5 |  335    5 |  385    5 |
  36    5 |   86    5 |  136    5 |  186    5 |  236    5 |  286    5 |  336    5 |  386    5 |
  37    5 |   87    5 |  137    5 |  187    5 |  237    5 |  287    5 |  337    5 |  387    5 |
  38    5 |   88    5 |  138    5 |  188    5 |  238    5 |  288    5 |  338    5 |  388    5 |
  39    5 |   89    5 |  139    5 |  189    5 |  239    5 |  289    5 |  339    5 |  389    5 |
  40    5 |   90    5 |  140    0 |  190    5 |  240    5 |  290    5 |  340    5 |  390    5 |
  41    5 |   91    5 |  141    0 |  191    5 |  241    5 |  291    5 |  341    5 |  391    5 |
  42    5 |   92    5 |  142    0 |  192    5 |  242    5 |  292    5 |  342    5 |  392    5 |
  43    0 |   93    5 |  143    0 |  193    5 |  243    5 |  293    5 |  343    5 |  393    5 |
  44    0 |   94    5 |  144    0 |  194    5 |  244    5 |  294    5 |  344    5 |  394    5 |
  45    0 |   95    5 |  145    0 |  195    5 |  245    5 |  295    5 |  345    5 |  395    5 |
  46    0 |   96    5 |  146    5 |  196    5 |  246    5 |  296    5 |  346    5 |  396    0 |
  47    0 |   97    5 |  147    5 |  197    5 |  247    5 |  297    5 |  347    5 |  397    0 |
  48    0 |   98    5 |  148    5 |  198    5 |  248    5 |  298    5 |  348    5 |  398    0 |
  49    0 |   99    5 |  149    5 |  199    5 |  249    5 |  299    5 |  349    5 |  399    0 |
  50    0 |  100    5 |  150    5 |  200    5 |  250    5 |  300    5 |  350    5 |  400    0 |

COUNTS BY DRG
 401    0 |  451    0 |  501    5 |  551    5 |  601    5 |  651    0 |  701    0 |  751    0 |
 402    0 |  452    0 |  502    5 |  552    5 |  602    5 |  652    5 |  702    0 |  752    0 |
 403    0 |  453    5 |  503    5 |  553    5 |  603    5 |  653    5 |  703    0 |  753    0 |
 404    0 |  454    5 |  504    5 |  554    5 |  604    5 |  654    5 |  704    0 |  754    5 |
 405    5 |  455    5 |  505    5 |  555    5 |  605    5 |  655    5 |  705    0 |  755    5 |
 406    5 |  456    5 |  506    5 |  556    5 |  606    5 |  656    5 |  706    0 |  756    5 |
 407    5 |  457    5 |  507    5 |  557    5 |  607    5 |  657    5 |  707    5 |  757    5 |
 408    5 |  458    5 |  508    5 |  558    5 |  608    0 |  658    5 |  708    5 |  758    5 |
 409    5 |  459    5 |  509    5 |  559    5 |  609    0 |  659    5 |  709    5 |  759    5 |
 410    5 |  460    5 |  510    5 |  560    5 |  610    0 |  660    5 |  710    5 |  760    5 |
 411    5 |  461    5 |  511    5 |  561    5 |  611    0 |  661    5 |  711    5 |  761    5 |
 412    5 |  462    5 |  512    5 |  562    5 |  612    0 |  662    5 |  712    5 |  762    0 |
 413    5 |  463    5 |  513    5 |  563    5 |  613    0 |  663    5 |  713    5 |  763    0 |
 414    5 |  464    5 |  514    5 |  564    5 |  614    5 |  664    5 |  714    5 |  764    0 |
 415    5 |  465    5 |  515    5 |  565    5 |  615    5 |  665    5 |  715    5 |  765    5 |
 416    5 |  466    5 |  516    5 |  566    5 |  616    5 |  666    5 |  716    5 |  766    5 |
 417    5 |  467    5 |  517    5 |  567    0 |  617    5 |  667    5 |  717    5 |  767    5 |
 418    5 |  468    5 |  518    0 |  568    0 |  618    5 |  668    5 |  718    5 |  768    0 |
 419    5 |  469    5 |  519    0 |  569    0 |  619    5 |  669    5 |  719    0 |  769    4 |
 420    5 |  470    5 |  520    0 |  570    5 |  620    5 |  670    5 |  720    0 |  770    5 |
 421    5 |  471    5 |  521    0 |  571    5 |  621    5 |  671    5 |  721    0 |  771    0 |
 422    5 |  472    5 |  522    0 |  572    5 |  622    5 |  672    5 |  722    5 |  772    0 |
 423    5 |  473    5 |  523    0 |  573    5 |  623    5 |  673    5 |  723    5 |  773    0 |
 424    5 |  474    5 |  524    0 |  574    5 |  624    5 |  674    5 |  724    5 |  774    5 |
 425    5 |  475    5 |  525    0 |  575    5 |  625    5 |  675    5 |  725    5 |  775    5 |
 426    0 |  476    5 |  526    0 |  576    5 |  626    5 |  676    0 |  726    5 |  776    5 |
 427    0 |  477    5 |  527    0 |  577    5 |  627    5 |  677    0 |  727    5 |  777    5 |
 428    0 |  478    5 |  528    0 |  578    5 |  628    5 |  678    0 |  728    5 |  778    5 |
 429    0 |  479    5 |  529    0 |  579    5 |  629    5 |  679    0 |  729    5 |  779    5 |
 430    0 |  480    5 |  530    0 |  580    5 |  630    5 |  680    0 |  730    5 |  780    4 |
 431    0 |  481    5 |  531    0 |  581    5 |  631    0 |  681    0 |  731    0 |  781    5 |
 432    5 |  482    5 |  532    0 |  582    5 |  632    0 |  682    5 |  732    0 |  782    5 |
 433    5 |  483    5 |  533    5 |  583    5 |  633    0 |  683    5 |  733    0 |  783    0 |
 434    5 |  484    5 |  534    5 |  584    5 |  634    0 |  684    5 |  734    5 |  784    0 |
 435    5 |  485    5 |  535    5 |  585    5 |  635    0 |  685    5 |  735    5 |  785    0 |
 436    5 |  486    5 |  536    5 |  586    0 |  636    0 |  686    5 |  736    5 |  786    0 |
 437    5 |  487    5 |  537    5 |  587    0 |  637    5 |  687    5 |  737    5 |  787    0 |
 438    5 |  488    5 |  538    5 |  588    0 |  638    5 |  688    5 |  738    5 |  788    0 |
 439    5 |  489    5 |  539    5 |  589    0 |  639    5 |  689    5 |  739    5 |  789    5 |
 440    5 |  490    5 |  540    5 |  590    0 |  640    5 |  690    5 |  740    5 |  790    0 |
 441    5 |  491    5 |  541    5 |  591    0 |  641    5 |  691    5 |  741    5 |  791    5 |
 442    5 |  492    5 |  542    5 |  592    5 |  642    5 |  692    5 |  742    5 |  792    0 |
 443    5 |  493    5 |  543    5 |  593    5 |  643    5 |  693    5 |  743    5 |  793    5 |
 444    5 |  494    5 |  544    5 |  594    5 |  644    5 |  694    5 |  744    5 |  794    0 |
 445    5 |  495    5 |  545    5 |  595    5 |  645    5 |  695    5 |  745    5 |  795    2 |
 446    5 |  496    5 |  546    5 |  596    5 |  646    0 |  696    5 |  746    5 |  796    0 |
 447    0 |  497    5 |  547    5 |  597    5 |  647    0 |  697    5 |  747    5 |  797    0 |
 448    0 |  498    5 |  548    5 |  598    5 |  648    0 |  698    5 |  748    5 |  798    0 |
 449    0 |  499    5 |  549    5 |  599    5 |  649    0 |  699    5 |  749    5 |  799    5 |
 450    0 |  500    5 |  550    5 |  600    5 |  650    0 |  700    5 |  750    5 |  800    5 |

COUNTS BY DRG
 801    5 |  826    5 |  851    0 |  876    5 |  901    5 |  926    0 |  951    5 |  976    5 |
 802    5 |  827    5 |  852    0 |  877    0 |  902    5 |  927    5 |  952    0 |  977    5 |
 803    5 |  828    5 |  853    5 |  878    0 |  903    5 |  928    5 |  953    0 |  978    0 |
 804    5 |  829    5 |  854    5 |  879    0 |  904    5 |  929    5 |  954    0 |  979    0 |
 805    0 |  830    5 |  855    5 |  880    5 |  905    5 |  930    0 |  955    5 |  980    0 |
 806    0 |  831    0 |  856    5 |  881    5 |  906    5 |  931    0 |  956    5 |  981    5 |
 807    0 |  832    0 |  857    5 |  882    5 |  907    5 |  932    0 |  957    5 |  982    5 |
 808    5 |  833    0 |  858    5 |  883    5 |  908    5 |  933    5 |  958    5 |  983    5 |
 809    5 |  834    5 |  859    0 |  884    5 |  909    5 |  934    5 |  959    5 |  984    5 |
 810    5 |  835    5 |  860    0 |  885    5 |  910    0 |  935    5 |  960    0 |  985    5 |
 811    5 |  836    5 |  861    0 |  886    5 |  911    0 |  936    0 |  961    0 |  986    5 |
 812    5 |  837    5 |  862    5 |  887    5 |  912    0 |  937    0 |  962    0 |  987    5 |
 813    5 |  838    5 |  863    5 |  888    0 |  913    5 |  938    0 |  963    5 |  988    5 |
 814    5 |  839    5 |  864    5 |  889    0 |  914    5 |  939    5 |  964    5 |  989    5 |
 815    5 |  840    5 |  865    5 |  890    0 |  915    5 |  940    5 |  965    5 |  990    0 |
 816    5 |  841    5 |  866    5 |  891    0 |  916    5 |  941    5 |  966    0 |  991    0 |
 817    0 |  842    5 |  867    5 |  892    0 |  917    5 |  942    0 |  967    0 |  992    0 |
 818    0 |  843    5 |  868    5 |  893    0 |  918    5 |  943    0 |  968    0 |  993    0 |
 819    0 |  844    5 |  869    5 |  894    5 |  919    5 |  944    0 |  969    5 |  994    0 |
 820    5 |  845    5 |  870    5 |  895    5 |  920    5 |  945    5 |  970    5 |  995    0 |
 821    5 |  846    5 |  871    5 |  896    5 |  921    5 |  946    5 |  971    0 |  996    0 |
 822    5 |  847    5 |  872    5 |  897    5 |  922    5 |  947    5 |  972    0 |  997    0 |
 823    5 |  848    5 |  873    0 |  898    0 |  923    5 |  948    5 |  973    0 |  998    0 |
 824    5 |  849    5 |  874    0 |  899    0 |  924    0 |  949    5 |  974    5 |  999   40 |
 825   5 |  850   0 |  875   0 |  900   0 |  925   0 |  950   5 |  975   5 |

COUNTS BY MDC
    0    40
    1    385
    2    50
    3    139
    4    212
    5    471
    6    308
    7    185
    8    495
    9    163
   10    156
   11    223
   12    105
   13    125
   14    68
   15    17
   16    75
   17    136
   18    86
   19    45
   20    20
   21    101
   22    34
   23    50
   24    40
   25    30

COUNTS BY RTC
    0   3719
    1    1
    2    0
    3    0
    4    6
    5    5
    6    0
    7    8
    8    0
    9    5
   10    5
   11    5
   12    0
   13    0
   14    0
   15    5
```

```
TOTAL RECORDS PROCESSED   3759
```

Index