**19 December 2011**

CDR Patient Table

for the MHS Data Repository (MDR)

(Version 1.00.00)

Current Specification

Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Date | Originator | Para/Tbl/Fig | Description of Change |
| 1.00.00 | 12/19/2011 | J. MacLeod | All | Baseline |

# MDR CDR Patient Table

1. Background

This specification describes the process required to maintain the Military Health System (MHS) Data Repository (MDR) Clinical Data Repository (CDR) Patient data set using data received from the Clinical Data Repository (CDR). The MDR CDR Patient data will be used primarily by other MDR CDR data processors to append EDIPN and other patient identifying information to their data.

1. Transmission (Format and Frequency)

The patient information required from the CDR is transmitted in three separate feed types. The main type is the CDR Patient Data. The filenames of the files that contain these data start with the letters PTPID. These feeds contain most of the information about the patients. Two other types of feeds are used to augment the patient data. The first contains, among other things, SPONSOR Social Security Number (SSN) and Family Member Prefix (FMP) data. The filenames of the files that contain these data start with the letters PTGID. The second contains, among other things, the patient category. The filenames of the files that contain these data start with the letters PTGDT.

These files, and the offices which provide them, are listed below:

**Table 1 - Source Data**

|  |  |
| --- | --- |
| **Source File** | **Source (CDR)** |
| CDR Patient Table (PTPID\*.DAT) | CDR (DHIMS) |
| CDR Sponsor SSN/FMP data (PTGID\*.DAT) | CDR (DHIMS) |
| CDR Patient Category data (PTGDT\*.DAT) | CDR (DHIMS) |

Source files are provided according to the frequency described in the table below.

**Table 2 - Frequency of Source File**

|  |  |
| --- | --- |
| **Source File** | **Frequency** |
| CDR Patient Table (PTPID\*.DAT) | Weekly |
| CDR Sponsor SSN/FMP data (PTGID\*.DAT) | Weekly |
| CDR Patient Category data (PTGDT\*.DAT) | Weekly |

1. Organization and batching

The first step in MDR processing is to batch the raw data received from CDR. The batched raw data are stored in /mdr/raw/cdr according to routine MDR operating procedures.

The raw data are used to update the master patient, using the CDR Patient Id as the key. The updated table is then merged with other MDR data (e.g., Master Patient Index (MPI)) to append certain variables. See section V for the details of the derivations and merges. The master patient data are then saved in a single SAS data set.

This processing will occur whenever new raw data are placed in the MDR.

1. Receiving Filters

The raw files sent from the CDR contain header and trailer records. The first five lines of the files are header records that contain meta data that describe the file, e.g., the start time of the extract that created the file. The last two lines of the files are trailer records that also contain meta data that describe the file, e.g., the total number of actual data records in the file. The processor must strip these records from the data.

1. Field Transformations and File Types for MDR Core Databases

There are several merges required to prepare the MDR Patient dataset. An asterisk after the merge file name indicates that existing MDR processing utilities should be used.

**Table 3 - External Reference File Merges**

| **Merge** | **Date Matching** | **Additional Matching** |
| --- | --- | --- |
| MPI\* | None | See the MPI specification |
| DEERS VM6 | None | Patient SSN and date of birth |

Business rules for each of the appended fields that result from the merges above, are described in the body of the table in Section VIII, or in an appendix, referenced in that table.

# Table 4 - MDR Patient SAS Data Set

| **Variable Name** | **SAS Name** | **Format** | **Business Rule** |
| --- | --- | --- | --- |
| CDR EDIPN | cdr\_edipn | $10. | No transformation.  |
| CDR Patient Id | cdr\_patient\_id | $20. | No transformation.  |
| Date of Birth | dob | 8. | No transformation.  |
| Gender | gender | $1. | No transformation.  |
| Patient Social Security Number | patssn | $9. | No transformation.  |
| CDR Sponsor Social Security Number | cdr\_sponssn | $9. | No transformation.  |
| Family Member Prefix | fmp | $2. | No transformation.  |
| Patient Category | patcat | $3. | No transformation.  |
| **Internally Derived Fields (or simple proc format assignments)** |
| EDI\_PN | edipn | $10. | If the VM6 EDIPN is populated set the edipn = vm6\_edipn. If it is not and the MPI EDIPN is populated, set the edipn = mpi\_edipn. If neither of those two is populated, set the edipn = cdr\_edipn.  |
| Sponsor Social Security Number | sponssn | $10. | If the VM6 SPONSSN is populated set the sponssn = vm6\_sponssn. If it is not and the MPI SPONSSN is populated, set the sponssn = mpi\_sponssn. If neither of those two is populated, set the sponssn = cdr\_sponssn.  |
| Universal Patient ID | upid | $14 | If the edipn is populated, set the UPID=edipn. If it is not populated and the PATSSN is populated, set the UPID=patssn. If neither is populated, set the UPID = “UN”||cdr\_patient\_id. |
| **From MPI Merge** |
| MPI-assigned EDIPN | mpi\_edipn | $10. | Use a combination of the VM6-assigned sponssn and the cdr\_sponssn as the key for merging with the MPI data. If the VM6 sponssn is populated use that as the key, otherwise use the CDR sponssn. Use the utility macros for the merge. See the MPI specification for details about the merge. |
| MPI-assigned SPONSSN | mpi\_sponssn | $9. | Use a combination of the VM6-assigned EDIPN, the MPI-assigned EDIPN and the CDR EDIPN as the key for merging with the MPI data. If the VM6 EDIPN is populated use that as the key. If it is not and the mpi-assigned EDIPN is populated, use the MPI-assigned EDIPN as the key. If neither of those two is populated, use the CDR EDIPN as the key. Use the utility macros for the merge. See the MPI specification for details about the merge. |
| **From VM6 Merge** |
| VM6-assigned EDIPN | VM6\_edipn | $10. | Read in each month of VM6 data from FY05 FM01 through present, keeping only primary records with a populated PATSSN. Then keep only the most recent record for each patient SSN. Merge the patient table to the resulting VM6 data by patient SSN and date of birth to assign the EDIPN. |
| VM6-assigned SPONSSN | VM6\_sponssn | $9. | Read in each month of VM6 data from FY05 FM01 through present, keeping only primary records with a populated PATSSN. Then keep only the most recent record for each patient SSN. Merge the patient table to the resulting VM6 data by patient SSN and date of birth to assign the Sponsor SSN. |

1. Refresh Frequency

This processing will occur whenever new raw data are placed in the MDR. This should occur on a weekly basis.

1. Data Quality

It is expected that when the CDR Patient processor is run, that basic quality checks are performed throughout the process. It is recommended that the person in charge of processing develop a spreadsheet to track key characteristics of the data across processing cycles, making it relatively easy to understand how the data should generally look. The processor should review these statistics each processing cycle to releasing the data. DHCAPE (the functional proponent and the specification author) should be contacted immediately should any quality issues arise. These checks, at a minimum, should include:

* Total record counts in the data feed should have a relatively stable distribution across processing cycles. Any anomalies should immediately be investigated.
* The number of update, insert, and delete records in the raw data should be similar in scope and proportion across processing cycles.
* The number of records that match when doing the external merge should be consistent.
* The distribution of all categorical fields (ex. patcat, fmp) should be consistent. The results of proc freq analyses will verify this.
* The number of null values for important fields such as CDR\_PATIENT\_ID, EDIPN, Patient SSN and Sponsor SSN should be tracked across processing cycles.
* When reading in the raw data feed, a small number of records should be printed off and manually inspected to ensure they have been properly ingested.
* Cross tabulations and prints should be reviewed on derived elements to ensure the derivation logic works.
* A data flow tracker should be built to ensure that all records that are intended to make it into the final dataset do. In other words, all inserts, updates, and deletions should be tracked and explained in the data flow worksheet.