ESP Data Extract Testing Guide

**Revision History**

| Version Number | Modification Date | By | Description of Changes |
| --- | --- | --- | --- |
| 1.0 | 22 May 2018 | K. Eberhardt | Initial version |
| 1.1 | 09 Apr 2019 | K Eberhardt | Added Sections For Database Validation |
| 1.2 | 11 Apr 2019 | K. Eberhardt | Updated sections for database validation for lab results |
| 1.3 | 24 Apr 2019 | J. Miller | Minor updates after testing – fixed single quotes |
| 1.4 | 29 Apr 2019 | K. Eberhardt | Updated Deleting Test Data section to better warn against deletion of required provenances. Updates to formatting and layout. Updated TOC. |
| 1.5 | 05 Dec 2019 | K. EberhardtJ. Miller | Update queries. Add new queries to get load counts by years and months |

Contents

[1 Overview 3](#_Toc31196703)

[2 Naming Files & Uploading To The ESP Server 4](#_Toc31196704)

[3 Preparing File(s) For Processing 4](#_Toc31196705)

[4 Loading The File(s) 4](#_Toc31196706)

[5 Using the UI to Validate 6](#_Toc31196707)

[6 Using the Database to Validate: Member 8](#_Toc31196708)

[7 Using the Database to Validate: Provider 10](#_Toc31196709)

[8 Using the Database to Validate: Visits 10](#_Toc31196710)

[9 Using the Database to Validate: Medications 11](#_Toc31196711)

[10 Using the Database to Validate: Lab Results 12](#_Toc31196712)

[11 Using the Database to Validation: Historical Data Load 13](#_Toc31196713)

[12 Troubleshooting Load Failures 13](#_Toc31196714)

[13 Deleting Test Data 14](#_Toc31196715)

# Overview

This document outlines a basic process for testing/validating client EMR data extract files. This process assumes sample extract files matching ESP file requirements have already been created.

Document written for ESP version 3.4

# Naming Files & Uploading To The ESP Server

1. Ensure the sample extract files match the naming convention outlined in the ESP FileSpec document.


2. Upload the files to the following directory on the ESP server:

**/srv/esp/data/fake**

# Preparing File(s) For Processing

1. It is recommended to process 1 file at a time for testing. When possible, start with the provider file first followed by the member file and then any of the additional files.
2. Use the following commands to move the test file in to the proper directory for processing:

**cd /srv/esp/data/fake
mv <insert\_filename\_here> ../epic/incoming**

For example:
**cd /srv/esp/data/fake
cp epicmed.esp.11292017 ../epic/incoming/**

# Loading The File(s)

1. Use the load\_epic command to attempt the loading of the file(s):

 **cd /srv/esp/prod
./bin/esp load\_epic**
2. When processing a particularly large file that may take a long time to complete, it is recommended to make the call using nohup. This will prevent the process from failing should you experience a network interruption while processing. This will log the output of the script to the nohup file.

**cd /srv/esp/prod**

**rm nohup.out
nohup ./bin/esp load\_epic &**

**tail –f nohup.out**

1. Review the processing output. If errors are encountered that prevent the load from completing, see the Troubleshooting section below. The number of valid records loaded should match the number of rows in the file (assuming there are no duplicate natural keys in the data):



1. A load report is also written to the /srv/esp/data/load\_reports directory:

**cd /srv/esp/data/load\_reports**

**ls -ltr** (this will list the files by time modified. The most recent file will be listed at the bottom)

**view *<insert\_filename\_here>***

Sample load file contents:



# Using the UI to Validate

1. Log in to the ESP User Interface (UI).
2. Navigate to “Administration -> Site Administration”



1. In the “EMR” section click on the item name that matches the test file that was loaded:



1. Click on a row to view the detailed information for the record that was loaded.


2. Review the fields that were populated (and not populated).

3. Items to Review:
* Are all expected fields populated?
* Is data appearing the correct field?
* Is the data formatted correctly?
* Any items missing?

# Using the Database to Validate: Member

Login to the database using psql or a utility that gives you sql command line access and run the following commands:

1. Count of patients
Query:
select count(\*) from emr\_patient;

Questions/Comments:
Does the count match the expected number?
2. Check for null MRN
Query:
select count(\*) from emr\_patient where mrn is null or mrn in ('', ' ');

Questions/Comments:
Do an exceptionally large number of patients not have an MRN?

1. Check for null Last Name
Query:
select count(\*) from emr\_patient where last\_name is null or last\_name in ('', ' ');

Questions/Comments:
Are there any patients without a last name?

1. Verify gender counts and values
Query:
select count(\*), gender

from emr\_patient

group by gender;

Questions/Comments:
Gender should typically be populated for all patients and values should be one of (M, F, T, or U)

1. Verify race counts and values
Query:
select count(\*), race

from emr\_patient

group by race;

Questions/Comments:
Race should typically be populated for all patients and values should be one of the following:

CAUCASIAN
W
WHITE
BLACK
B
OTHER
O
MULTIRACIAL
INDIAN
I
AMERICAN INDIAN/ALASKAN NATIVE
ASIAN
A
NAT AMERICAN
NATIVE HAWAI
PACIFIC ISLANDER/HAWAIIAN
P
ALASKAN
HISPANIC

# Using the Database to Validate: Provider

Login to the database using psql or a utility that gives you sql command line access and run the following commands:

1. Count of providers

Query:
select count(\*) from emr\_provider;

Questions/Comments:
Does the count match the expected number?
2. Count of patients assigned as a PCP for each provider

Query:
select count(patient.id) patient\_count, pro.id provider\_id

from emr\_provider pro, emr\_patient patient

where pro.id = patient.pcp\_id

group by pro.id;

Questions/Comments:
Are patients matching up to providers?

Are a large number of patients assigned to pcp\_id of 1 (unknown)?

# Using the Database to Validate: Visits

Login to the database using psql or a utility that gives you sql command line access and run the following commands:

1. Count of visits

Query:
select count(\*) from emr\_encounter;

Questions/Comments:
Does the count match the expected number?
2. Count of encounters per diagnosis code

Query:
select count(\*), dx\_code\_id from emr\_encounter e, emr\_encounter\_dx\_codes dx

where e.id = dx.encounter\_id

group by dx\_code\_id;

Questions/Comments:
Are diagnosis codes being included and correctly parsed for encounters?

1. Count of records with pregnancy information

Query:
select count(\*), pregnant

from emr\_encounter

group by pregnant;

Questions/Comments:
Is pregnancy information being populated interpreted based on extract data?

1. Count of records with EDD (estimated due date) populated.

Query:
select count(\*),

case when edd is not null then 'edd\_populated' else null end as edd\_check

from emr\_encounter

group by edd\_check;

If EDD is provided, does it match the general number of encounters expected?

1. Count of records and values with height and weight and bmi

Queries:
select count(\*), raw\_height

from emr\_encounter

group by raw\_height;

select count(\*), raw\_weight

from emr\_encounter

group by raw\_weight;

select count(\*), raw\_bmi

from emr\_encounter

group by raw\_bmi;

Questions/Comments:
Are fields being populated? Do the formats match the spec? Note, some sites send raw\_bmi as opposed to height and weight so not having these values may not be of concern.

# Using the Database to Validate: Medications

Login to the database using psql or a utility that gives you sql command line access and run the following commands:

1. Count of medications
Query:
select count(\*) from emr\_prescription;

Questions/Comments:
Does the count match the expected number?
2. Count of records without a start date
Query:
select count(\*) from emr\_prescription where start\_date is null;

Questions/Comments:
Start Date is important so it really should be populated for true rx records. If the count of null records is high, please investigate.
3. Count of records without a start date
Query:
select count(\*) from emr\_prescription where end\_date is null;

Questions/Comments:
End Date is often not populated, but ideally it should be sent as it is important for computing rx spans.

# Using the Database to Validate: Lab Results

Login to the database using psql or a utility that gives you sql command line access and run the following commands:

1. Count of lab results
Query:
select count(\*) from emr\_labresult;

Questions/Comments:
Does the count match the expected number?
2. Count of lab tests by native\_name
Query:
select count(\*), native\_name, procedure\_name
from emr\_labresult
group by native\_name, procedure\_name;

Questions/Comments:
Do the native names appear as expected? Do the number of tests appear reasonable per test?
Note: this may not be advised on systems with a large amount of lab results
3. Count of records with specimen source populated
Query:
select count(\*), specimen\_source from emr\_labresult
group by specimen\_source;

Questions/Comments:
At least some tests should have a specimen source.
4. Count of records without a result string
Query:
select count(\*) from emr\_labresult
where result\_string is null or result\_string in ('', ' ');

Questions/Comments:
Ensure that most lab results have a result value.

# Using the Database to Validation: Historical Data Load

Login to the database using psql or a utility that gives you sql command line access and run the following commands:

1. Count of lab result records with breakdown by month and year
Query:
select count(\*), date\_part('year', date) as dateyear, date\_part('month', date) as datemon

from emr\_labresult
group by dateyear, datemon order by dateyear, datemon;

1. Count of medication records with breakdown by month and year
Query:
select count(\*), date\_part('year', date) as dateyear, date\_part('month', date) as datemon
from emr\_prescription

group by dateyear, datemon order by dateyear, datemon;

1. Count of encounter records with breakdown by month and year
Query:
select count(\*), date\_part('year', date) as dateyear, date\_part('month', date) as datemon

from emr\_encounter

group by dateyear, datemon order by dateyear, datemon;

# Troubleshooting Load Failures

1. Carefully review any error messages displayed in the screen or in the log file (var/log/esp.log). Often the error message will provide detailed information to identify the issue.
2. For errors relating to a field exceeding the length of the column, this can sometimes happen if data file contains a header row or the fields are not populated in the correct order. Check the file to ensure that a) it does not contain header row and
b) the data fields line up with the FileSpec document.
If a field length does need to be expanded to support correct EMR data, please contact CII for assistance.
3. If no error message is displayed, however the load report is reporting 0 rows were loaded, ensure that a file with this filename has not been loaded before. Each loaded file name must be unique and only loaded once in ESP. To “reload” a file that has previously been loaded in ESP, use the following command:
**/srv/esp/prod/bin load\_epic --reload**

# Deleting Test Data

1. To fully delete the test file data from ESP, login to the ESP UI.
2. Navigate to “Administration” -> “Site Administration”


3. In the “EMR” section, select “Provenances”


4. The following are special provenance files for ESP use. **DO NOT DELETE** the following:

5. Place a checkbox next to the data file you want to delete and then choose “Delete Selected Provenances” from the dropdown menu. **Reminder: DO NOT DELETE Provenances 1 and 2.**



1. Click “Go”
2. Review the data to confirm what will be deleted. Please note, deleting one type of record may delete additional data that may be tied to record being deleted. If satisfied, select “Yes I’m sure”.
3. Please note, large files may take a few minutes to display and/or delete.