WhiteRabbit

Introduction

Scope and purpose

WhiteRabbit is a software tool to help prepare for ETLs (Extraction, Transformation, Loading) of longitudinal healthcare databases into the Observational Medical Outcomes Partnership (OMOP) Common Data Model (CDM). The source data can be in comma-separated text files, or in a database (MySQL, SQL Server, Oracle, PostgreSQL, Microsoft APS, Microsoft Access, Amazon RedShift). Note that the CDM will need to be in one of a limited set of database platforms (SQL Server, Oracle, PostgreSQL, Microsoft APS, Amazon RedShift).

WhiteRabbit's main function is to perform a scan of the source data, providing detailed information on the tables, fields, and values that appear in a field. This scan will generate a report that can be used as a reference when designing the ETL, for instance by using the RabbitIn-a-Hat tool. White Rabbit differs from standard data profiling tools in that it attempts to prevent the display of personally identifiable information (PII) data values in the generated output data file.

Process Overview

The typical sequence for using this software to scan source data in preparation of developing an ETL into an OMOP CDM:

1. Set working folder, the location on the local desktop computer where results will be exported.
2. Connect to the source database or CSV text file and test connection.
3. Select the tables of interest for the scan and scan the tables.
4. WhiteRabbit creates an export of information about the source data.

Once the scan report is created, this report can then be used in the Rabbit-In-a-Hat tool or as a stand-alone data profiling document.

Installation and support

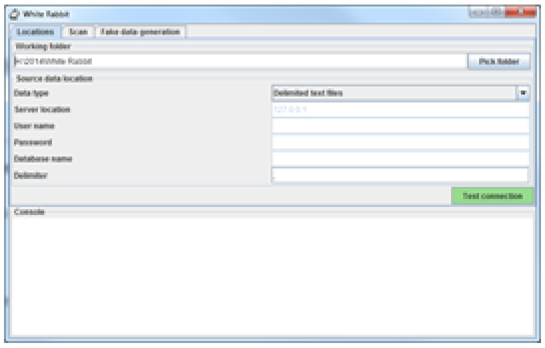
All source code and installation instructions available on GitHub: https://github.com/OHDSI/WhiteRabbit

Any bugs/issues/enhancements should be posted to the GitHub repository: https://github.com/OHDSI/WhiteRabbit/issues

Any questions/comments/feedback/discussion can be posted on the OHDSI Developer Forum: http://forums.ohdsi.org/c/developers

Using the Application Functions

Specifying the Location of Data



Working Folder

Any files that WhiteRabbit creates will be exported to this local folder. Use the “Pick Folder” button to navigate in your local environment where you would like the scan document to go.

Source Data

Here you can specify the location of the source data. The following source types are supported: delimited text files, MySQL, Oracle, SQL Server, and PostgreSQL. Below are connection instructions for each data type of data source. Once you have entered the necessary information, the “Test connection” button can ensure a connection can be made.

Delimited text files

- ***Delimiter:*** specifies the delimiter that separates columns, default is ‘,’ and your can write ‘tab’ for tab delimited.

WhiteRabbit will look for the files to scan in the same folder you set up as a working directory.

MySQL

- ***Server location:*** the name or IP address of the server running MySQL. You can also specify the port (ex: <host>:<port>), which defaults to 3306.
- ***User name:*** name of the user used to log into the server
- ***Password:*** password for the supplied user name
- ***Database name:*** name of the database containing the tables

Oracle

- ***Server location:*** this field contains the SID, service name, and optionally the port: '<host>/<sid>', '<host>:<port>/<sid>', '<host>/< service name >', or '<host>:<port>/<service name>'
- ***User name:*** name of the user used to log into the server
- ***Password:*** password for the supplied user name
- ***Database name:*** this field contains the schema (i.e. 'user' in Oracle terms) containing the tables

SQL Server

- ***Server location:*** the name or IP address of the server running SQL Server. You can also specify the port (ex: <host>:<port>), which defaults to 1433.
- ***User name:*** name of the user used to log into the server. Optionally, the domain can be specified as <domain>/<user> (e.g. 'MyDomain/Joe')
- ***Password:*** password for the supplied user name
- ***Database name:*** name of the database containing the tables

When the SQL Server JDBC drivers are installed, you can also use Windows authentication. In this case, user name and password should be empty.

1. Download the .exe from http://msdn.microsoft.com/en-us/sqlserver/aa937724.aspx.
2. Run it, thereby extracting its contents to a folder.
3. In the extracted folder you will find the file *sqljdbc_4.0/enu/auth/x64/sqljdbc_auth.dll* (64-bits) or *sqljdbc_4.0/enu/auth/x86/sqljdbc_auth.dll* (32-bits), which needs to be moved to a location on the system path, for example to *c:/windows/system32*.

MS APS

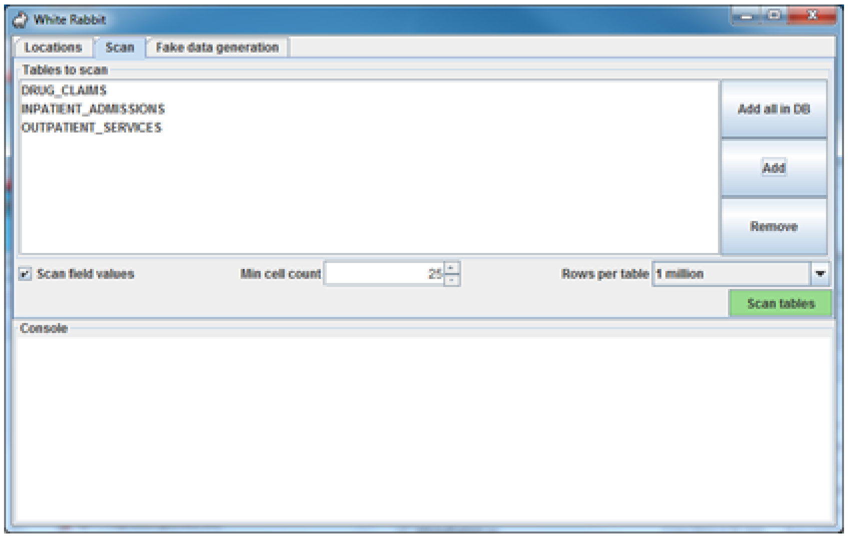
At this time WhiteRabbit does not run on this platform.

PostgreSQL

- ***Server location:*** this field contains the host name and database name (<host>/<database>)
- ***User name:*** name of the user used to log into the server
- ***Password:*** password for the supplied user name
- ***Database name:*** this field contains the schema containing the source tables

Scanning a Database

Performing the Scan



DRUG_CLAIMS
INPATIENT_ADMISSIONS
OUTPATIENT_SERVICES

Scan field values: ✓
Min cell count: 25
Rows per table: 1 million

A scan generates a report containing information on the source data that can be used to help design the ETL. Using the Scan tab in WhiteRabbit you can either select individual tables in the selected source database by clicking on 'Add' (Ctrl + mouse click), or automatically select all tables in the database by clicking on 'Add all in DB'.

There are a few setting options as well with the scan:

- Checking the "Scan field values" tells WhiteRabbit that you would like to investigate raw data items within tables selected for a scan (i.e. if you select Table A, WhiteRabbit will review the contents in each column in Table A).
  - "Min cell count" is an option when scanning field values. By default this is set to 25, meaning values in the source data that appear less than 25 will not appear in the report.
  - "Rows per table" is an option when scanning field values. By default, WhiteRabbit will random 1 million rows in the table. There are other options to review 100,000 or all rows within the table.
- Unchecking the "Scan field values" tells WhiteRabbit to not review or report on any of the raw data items.

Once all settings are completed, press the "Scan tables" button. After the scan is completed the report will be written to the working folder.

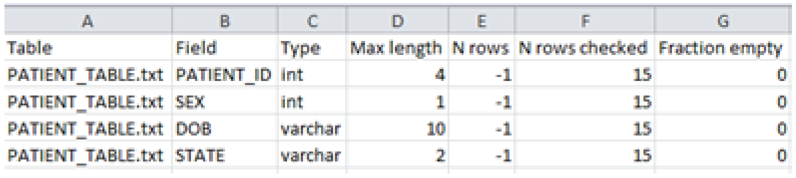
Running from the command line

For various reasons one could prefer to run WhiteRabbit from the command line. This is possible by specifying all the options one would normally select in the user interface in an .ini file. An example ini file can be found here. Then, we can reference the ini file when calling WhiteRabbit from the command line:

```
java -jar WhiteRabbit.jar -ini WhiteRabbit.ini
```

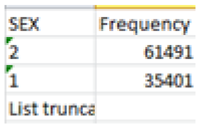
Reading the Scan

After the scan is completed, a “ScanReport” Excel document will be created in the working folder location selected earlier. The document will have multiple tabs, one as an “Overview” and then one tab for each database table or delimited text files selected for the scan. The “Overview” tab will tell you about each table selected, what the columns in each table are, the data type of the columns, the amount of data within the table, the number of rows scanned, and the fraction of data empty. Below is an example image of the “Overview” tab. Column A will list what table the information is about, Column B the column being reviewed, Column C, the type of the column, Column E is the number of rows (however with text files it will return - 1), Column F will tell you how many rows of the N rows were reviewed, and Column G will let you know how many rows are empty.



A	B	C	D	E	F	G
Table	Field	Type	Max length	N rows	N rows checked	Fraction empty
PATIENT_TABLE.txt	PATIENT_ID	int	4	-1	15	0
PATIENT_TABLE.txt	SEX	int	1	-1	15	0
PATIENT_TABLE.txt	DOB	varchar	10	-1	15	0
PATIENT_TABLE.txt	STATE	varchar	2	-1	15	0

For a tab that describes a single table, the columns names from the source table (or delimited text file) will be across the columns of the Excel tab. Each source table column will generate two columns in the Excel. One column will list all distinct values that have a “Min cell count” greater than what was set at time of the scan (Section - Performing the Scan). If a list of unique values was truncated, the last value in the list will be “List truncated”; this indicates that there are one or more additional unique source values that appear less than the number entered in the “Min cell count” described in Section - Performing the Scan. Next to each distinct value will be a second column that contains the frequency, or the number of times that value occurs in the data. These two columns (distinct values and frequency) will repeat for all the source columns in the table profiled in the workbook.



SEX	Frequency
2	61491
1	35401
List trunca	

The report is powerful in understanding your source data by highlighting what exists. For example, if the following results were given back on the “SEX” column within one of the tables scanned, we can see that there were two common values (1 and 2) that appeared 61,491 and 35,401 times respectively. WhiteRabbit will not define 1 as male and 2 as female; the data holder will typically need to define source codes unique to the source system. However these two values (1 & 2) are not the only values present in the data because we see this list was truncated. These other values appear with very low frequency (defined by “Min cell count”) and often represent incorrect or highly suspicious values. When generating an ETL we should not only plan to handle the high-frequency gender concepts 1 and 2 but the other low-frequency values that exist within this column.

Generating Fake Data

This feature allows one to create a fake dataset based on a WhiteRabbit scan report. This dataset could be used to develop ETL code when direct access to the data is not available.

This feature is currently still experimental.

Rabbit-In-a-Hat

RABBIT IN A HAT

Introduction

Scope and purpose

Rabbit-In-a-Hat comes with WhiteRabbit and is designed to read and display a WhiteRabbit scan document. WhiteRabbit generates information about the source data while Rabbit-In-a-Hat uses that information and through a graphical user interface to allow a user to connect source data to tables and columns within the CDM. Rabbit-In-a-Hat generates documentation for the ETL process it does not generate code to create an ETL.

Process Overview

The typical sequence for using this software to generate documentation of an ETL:

1. Scanned results from WhiteRabbit completed
2. Open scanned results; interface displays source tables and CDM tables
3. Connect source tables to CDM tables where the source table provides information for that corresponding CDM table
4. For each source table to CDM table connection, further define the connection with source column to CDM column detail
5. Save Rabbit-In-a-Hat work and export to a MS Word document.

Installation and support

Rabbit-In-a-Hat comes with WhiteRabbit, refer to WhiteRabbit's installation section.

Getting Started

Creating a New Document

To create a new document, navigate to *File –> Open Scan Report*. Use the “Open” window to browse for the scan document created by WhiteRabbit. When a scan document is open, the tables scanned will appear in orange boxes on the “Source” side of the Tables.

Save the Rabbit-In-a-Hat document by going *File –> Save as*.

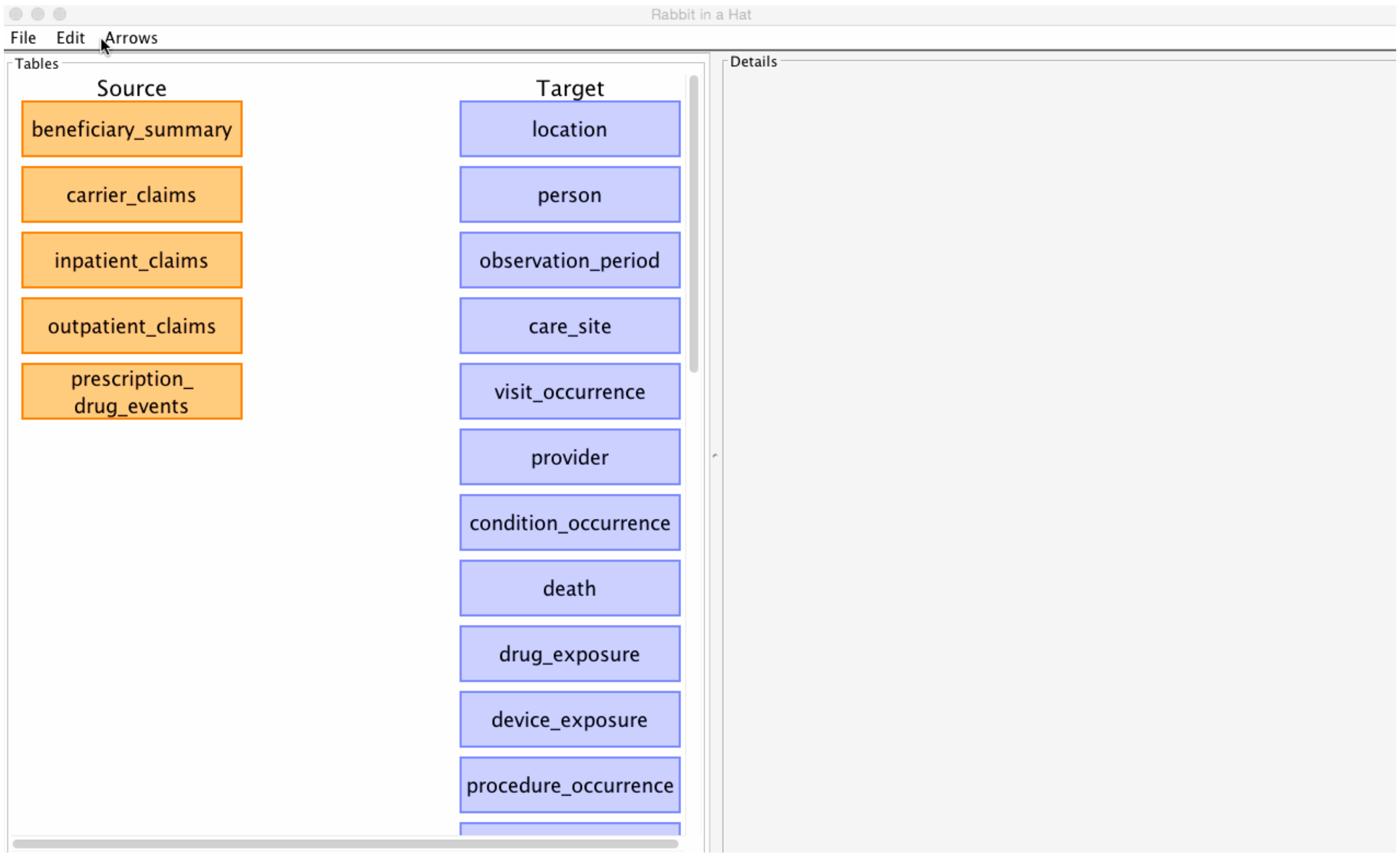
Open an Existing Document

To open an existing Rabbit-In-a-Hat document use *File –> Open ETL specs*.

Selecting Desired CDM Version

Rabbit-In-a-Hat allows you to select which CDM version (v4 or v5) you'd like to built your ETL specification against.

See the graphic below for how to select your desired CDM:



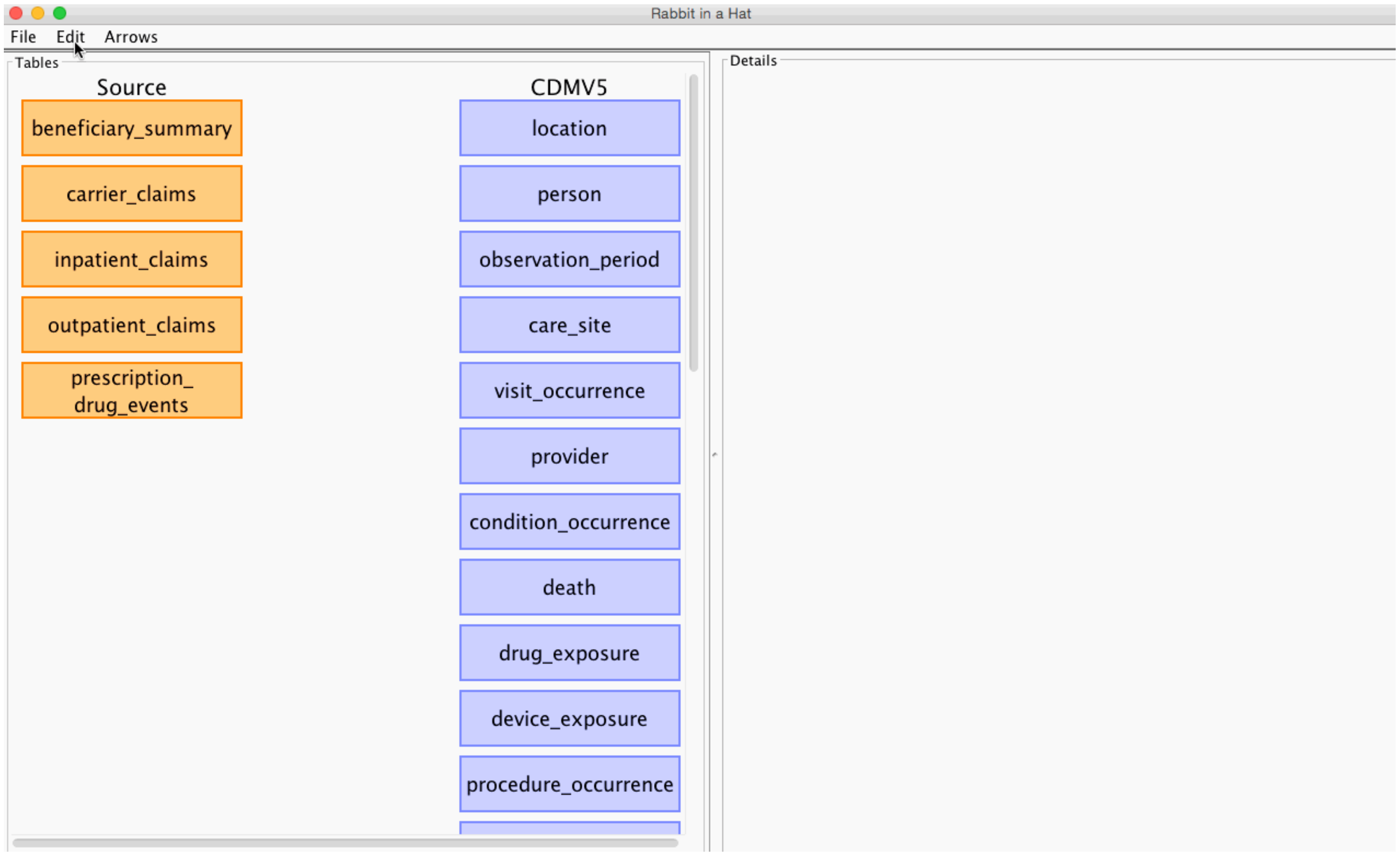
The CDM version can be changed at any time, but beware that some of your existing mappings may be lost in the process. By default, Rabbit-In-a-Hat will attempt to pereserve as many mappings between the source data and the newly selected CDM as possible. When a new CDM is selected, Rabbit-In-a-Hat will drop any mappings if the mapping's CDM table or CDM column name no longer exist

For instance, switching from CDMv4 to CDMv5, a mapping from source to CDM person.person_source_value will be kept because the person table has person_source_value in both CDMv4 and CDMv5. However, person.assocaited_provider_id exists only in CDMv4 and has been renamed to person.provider_id in CDMv5 and so that mapping will not be kept when switching between these two CDMs.

Loading in a Custom CDM

There are times when users might need to load in a customized version of the CDM, for instance if they are sandboxing new features. To load in a custom CDM schema, first you must create a CSV file that uses the same format as the existing CDMv5 schema file.

Once you have created the CSV file, load it into RiaH as shown below:

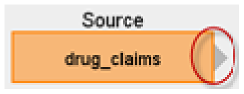


Please note that the name of the file you load in becomes the label that appears above the target tables, so "My Super File.csv" will create the label "My Super File" above the target tables, so name your CSV accordingly.

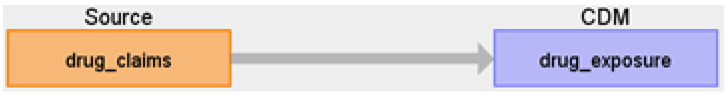
Connecting Source Tables to CDM Tables

It is assumed that the owners of the source data should be able to provide detail of what the data table contains, Rabbit-In-a-Hat will describe the columns within the table but will not provide the context a data owner should provide. For the CDM tables, if more information is needed navigate to the OMOP CDM web page (http://omop.org/CDM) and review the current OMOP specification.

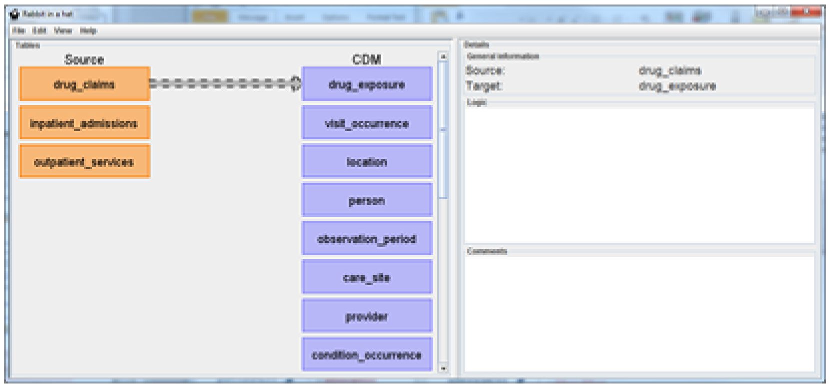
To connect a source table to a CDM table, simply hover over the source table until an arrow head appears.



Use your mouse to grab the arrow head and drag it to the corresponding CDM table. In the example below, the *drug_claims* data will provide information for the *drug_exposure* table.



If you click on the arrow once it will highlight and a *Details* window will appear in the right pane. Use this to describe *Logic or Comments* that someone developing the ETL code should know about this source table to CDM table mapping.

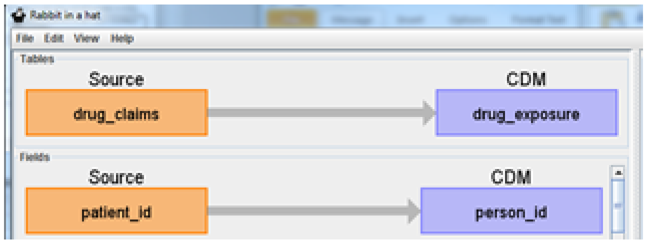


Source: drug_claims
Target: drug_exposure

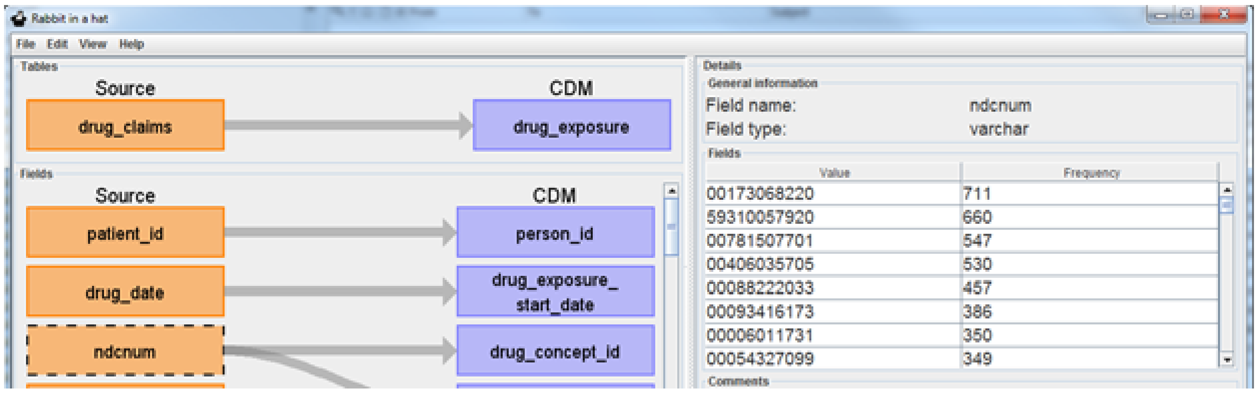
Continue this process until all tables that are needed to build a CDM are mapped to their corresponding CDM tables. One source table can map to multiple CDM tables and one CDM table can receive multiple mappings. There may be tables in the source data that should not be map into the CDM and there may be tables in the CDM that cannot be populated from the source data.

Connecting Source Fields to CDM Fields

By double clicking on an arrow connecting a source and CDM table, it will open a *Fields* pane below the arrow selected. The *Fields* pane will have all the source table and CDM fields and is meant to make the specific column mappings between tables. Hovering over a source table will generate an arrow head that can then be selected and dragged to its corresponding CDM field. For example, in the *drug_claims* to *drug_exposure* table mapping example, the source data owners know that *patient_id* is the patient identifier and corresponds to the *CDM.person_id*. Also, just as before, the arrow can be selected and *Logic* and *Comments* can be added.



If you select the source table orange box, Rabbit-In-a-Hat will expose values the source data has for that table. This is meant to help in the process in understanding the source data and what logic may be required to handle the data in the ETL. In the example below *ndcnum* is selected and raw NDC codes are displayed starting with most frequent (note that in the WhiteRabbit scan a “Min cell count” could have been selected and values below that frequency will not show).



Field name: ndcnum
Field type: varchar

Value	Frequency
00173068220	711
59310057920	660
00781507701	547
00406035705	530
00088222033	457
00093416173	386
00006011731	350
00054327099	349

Continue this process until all source columns necessary in all mapped tables have been mapped to the corresponding CDM column. Not all columns must be mapped into a CDM column and not all CDM columns require a mapping. One source column may supply information to multiple CDM columns and one CDM column can receive information from multiple columns.

Generating an ETL Document

To generate an ETL MS Word document use *File –> Generate ETL* document and select a location to save. It may take a moment before the document is finished creating. This document will contain all notes from Rabbit-In-a-Hat.

Once the information is in the document, if an update is needed you must either update the information in Rabbit-In-a-Hat and regenerate the document or update the document. If you make changes in the document, Rabbit-In-a-Hat will not read those changes and update the information in the tool. However it is common to generate the document with the core mapping information and fill in more detail within the document.

Once the document is completed, this should be shared with the individuals who plan to implement the code to execute the ETL.

Generating a testing framework

To make sure the ETL process is working as specified, it is highly recommended to create unit tests that evaluate the behavior of the ETL process. To efficiently create a set of unit tests Rabbit-in-a-Hat can generate a testing framework.

Best Practices

The following lists best practices in using WhiteRabbit and Rabbit-In-a-Hat to manage your ETL documentation process:

- **Overall Process:**
  - When going through the ETL exercise, it is critical to get all key stakeholders in the room such as the data owners and individuals who plan to perform/manage research using the CDM. We have found different stakeholders have different perspectives on source data and the conversation that occurs improves the ETL process.
- **WhiteRabbit:**
  - If it is known some tables will not be needed from the source data, do not include them in the scan. If there is a question to if the table is necessary it is better to include it.
- **Rabbit-In-a-Hat:**
  - Start with mapping tables to tables and then dive into how one table's fields map into CDM fields. In other words, stay at the table level until all tables a mapped, and then start to map fields.
  - If your source data does not contain certain information you do not need to impute or generate information to fulfil a CDM requirement. For example, if your source data does not contain an end date for when a medication was stopped you do not need to population the DRUG_EXPOSURE.DRUG_EXPOSURE_START_DATE column.
  - Derived CDM tables, like DRUG_ERA, typically will not receive a mapping from the source data because they are generated off the CDM table (in this case DRUG_ERA is generated off DRUG_EXPOSURE).