CDM Builder: Setup and Usage

Version 1.8

September 15, 2016



Authors

Janssen Pharmaceutical Research & Development, LLC

# Table of Contents

[Table of Contents 3](#_Toc408502150)

[1. Introduction 4](#_Toc408502151)

[2. Scope and Purpose 4](#_Toc408502152)

[3. Process Overview 5](#_Toc408502153)

[4. Installation and Support 6](#_Toc408502154)

[5. Using the Application Functions 6](#_Toc408502155)

[Build Configuration 6](#_Toc408502156)

[Usage 8](#_Toc408502157)

# Introduction

The CDM Builder can be used to transform observational datasets from their native formats and schemas into the OMOP Common Data Model (version 4). This document details the setup and configuration needed to execute the Builder and produce CDM-based databases.

# Scope and Purpose

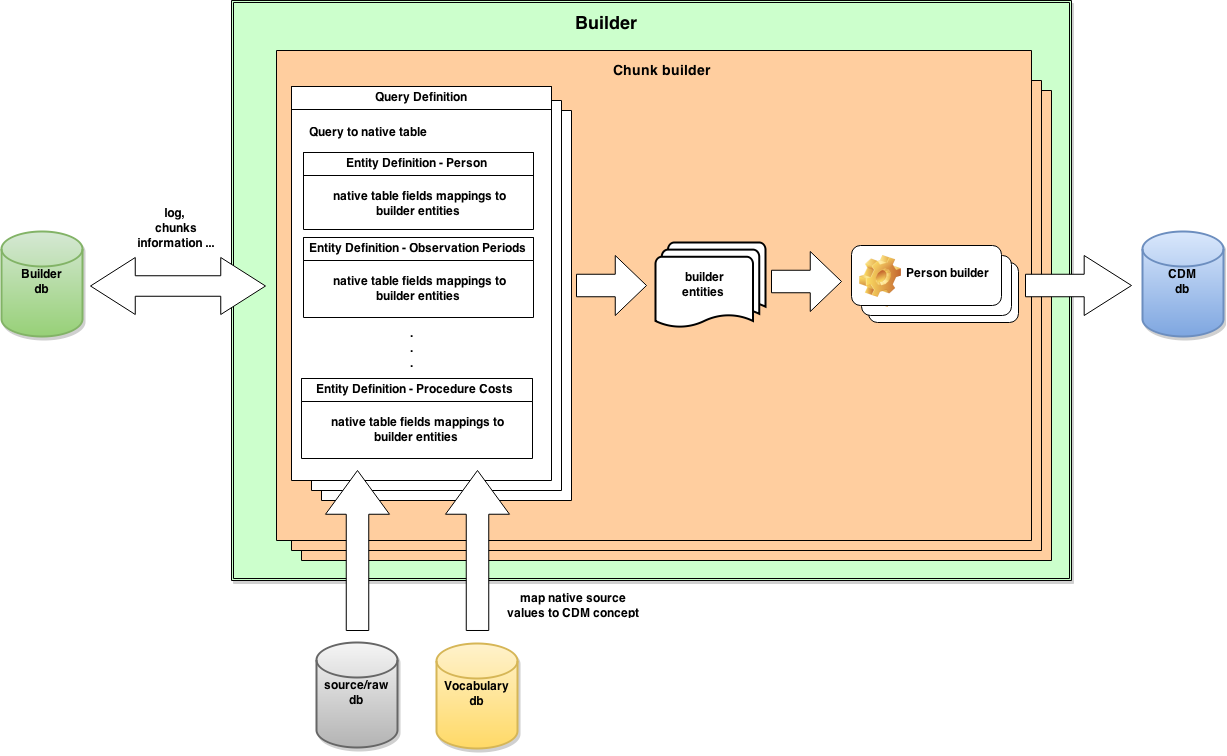
The purpose of this tool is to build CDMs off the following databases:

* Clinical Practice Research Datalink (CPRD)
* Japan Medical Data Center (JMDC)
* National Health and Nutrition Examination Survey (NHANES)
* Optum Extended Socio-Economic Status (SES)
* Optum Extended Date of Death (DOD)
* Optum Oncology EHR
* Premier
* Truven MarketScan Commercial Claims & Encounters (CCAE)
* Truven MarketScan Medicare (MDCR)
* Truven MarketScan Medicaid (MDCD)

# Process Overview

1. Based on the user selections for number of batches and batch size, a queue of chunks is created for processing. That is, the workload of converting all of the patients in the source database, along with their associated records, is divided into parts that can be more easily handled by multiple servers.
2. Figure 1.1 illustrates how the CDM Builder works. The Builder database is meant to store the overall Builder workload. The initial run of the Builder will create this metadata. All servers running Builder can then execute these jobs, using the source database for the content, and the Vocabulary database for the concept mapping. The results of these operations, along with the creation of the patients (in the Person table) within that chunk, are then added into the CDM database.

**Figure 1.1**



# Installation and Support

All source code and installation instructions are available on GitHub:   
<https://github.com/OHDSI/ETL-CDMBuilder>

Any bugs/issues/enhancements should be posted to the GitHub repository: <https://github.com/OHDSI/ETL-CDMBuilder/issues>

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

# Using the Application Functions

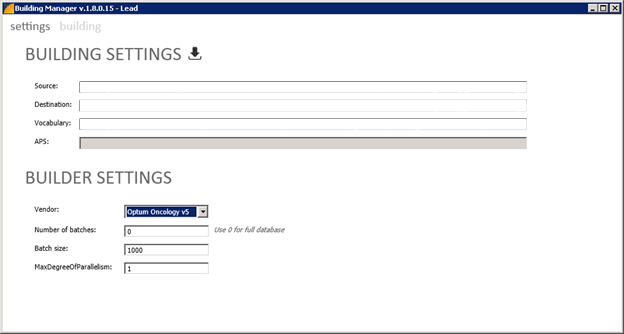
## Build Configuration

1. Adjust the Configuration file (located at \org.ohdsi.cdm\Presentation\org.ohdsi.cdm.presentation.buildingmanager\app.config). This file includes 4 connection strings: Source, Destination, Vocabulary, and Builder (see Table 1.0).

Please note that this configuration is used to generate the initial setup for your CDM build. Its details are then written to the Builder database once execution begins, so that other servers can leverage it.

|  |  |
| --- | --- |
| **Table 1.0** | |
| **Connection String** | **Definition** |
| Source | The original database in its native schema |
| Destination | The CDM database to be built; actual database does not need to exist as Builder can create it |
| Vocabulary | The database that contains the OMOP Vocabulary to reference for concept mapping |
| Builder | The database that will be used to support Builder processing operations |

**Figure 1.2**



1. Figure 1.2 shows the “Settings” window in the Builder GUI.
   1. The Source, Destination, and Vocabulary fields are all populated either via the config file (org.ohdsi.cdm.presentation.buildingmanager.exe.config) or the Builder database if this configuration has already been submitted. The Builder database has several tables that auto-populate
2. There are a few additional settings needed before the Build can be initiated:
   1. The “Number of batches” field can be used to specify if you want to convert the entire source database (0) or a sample size for testing purposes.
   2. The “Batch size” refers to the number of person(s) that should be part of each those batches.
   3. The “MaxDegreeofParallelism” establishes the number of processes to start on this particular machine. This value should be based on the hardware resources at your disposal.
      1. It has been observed that a server with a 24-core CPU and 256 GB RAM should invoke 10 processes for optimal performance. Similarly, for a workstation with a 2-core CPU and 12 GB RAM, 3-4 processes was the best choice. Using these examples, you should determine the number of processes that fit best with your hardware configuration.

## Usage

1. After establishing the Build settings, click on the “Building” tab in the Builder GUI to manage the processing on this server. The play button can be used to start the full build and the timeline indicates the progress of the build.
2. Each of the steps in the timeline can be skipped if you’ve already completed them. It is recommended to allow CDM builder to create the destination database instead of creating it manually.
3. The timeline refreshes every few seconds to provide you real-time information on the build.
4. If any errors are encountered, you will see them in the “Errors” section for debugging. Please log these to the GitHub issues list.

**Figure 1.3**

