OMOP Common Data Model
Extract, Transform & Load Tutorial
What this tutorial will provide . . .

- Suggested process for developing a CDM ETL
- OHDSI ETL tools: White Rabbit, Rabbit-In-A-Hat, and Usagi
- Resources like the Book of OHDSI and THEMIS
- Generation of a simple ETL examples
# Agenda

<table>
<thead>
<tr>
<th>Time</th>
<th>Agenda Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>9:00-9:30</td>
<td>Overview</td>
</tr>
<tr>
<td>9:30-10:45</td>
<td>ETL Step 1 – Design Your ETL</td>
</tr>
<tr>
<td>10:45-11:15</td>
<td>Break</td>
</tr>
<tr>
<td>11:15-12:00</td>
<td>ETL Step 2 – Mapping to the Vocabulary</td>
</tr>
<tr>
<td>12:00-1:00</td>
<td>Lunch</td>
</tr>
<tr>
<td>1:00-1:30</td>
<td>ETL Step 2 – Mapping to the Vocabulary (continued)</td>
</tr>
<tr>
<td>1:30-3:00</td>
<td>ETL Step 3 – Develop ETL</td>
</tr>
<tr>
<td>3:00-3:30</td>
<td>Break</td>
</tr>
<tr>
<td>3:30-4:15</td>
<td>ETL Step 4 – Quality Control</td>
</tr>
<tr>
<td>4:15-4:45</td>
<td>ETL Step 5 – ETL Maintenance</td>
</tr>
<tr>
<td>4:45-5:00</td>
<td>ETL Pain Points &amp; Conclusions</td>
</tr>
</tbody>
</table>
Ground Rules

• We have build in some decent sized breaks, please return before times up

• We are recording this presentation for future use

• We may take some questions off-line if too specific
## Instructors

<table>
<thead>
<tr>
<th>Clair Blacketer</th>
<th>Erica A. Voss</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Clair Blacketer" /></td>
<td><img src="image2" alt="Erica A. Voss" /></td>
</tr>
<tr>
<td><img src="image3" alt="Evanette K. Burrows" /></td>
<td><img src="image4" alt="Maxim Moinat" /></td>
</tr>
</tbody>
</table>

Evanette K. Burrows

Maxim Moinat

hey.
Connecting to the Hotel WIFI

Network: OHDSISYMP

Password: OHDSI2019
Follow Along

• This full deck can be found here:
  – https://github.com/OHDSI/Tutorial-ETL
  – Materials ➔ OMOP Common Data Model Extract, Transform & Load.pptx
How to Sign into the Remote Desktop

From your command prompt, type
%systemroot%/system32/mstsc.exe
to launch Remote Desktop
How to Sign into the Remote Desktop
How to Sign into the Remote Desktop

• Use the shortcut on the desktop named “Remote Desktop”

URL TBD

• Pick one of the rows and put your name on the second column
How to Sign into the Remote Desktop

• Take Column A from spreadsheet and copy into the “Computer” field
How to Sign into the Remote Desktop

• Pick ‘Use Another Account’

• Copy username from Column C

• Copy password from Column D

<table>
<thead>
<tr>
<th>RDP URL</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>ec2-34-226-245-112.compute-1.amazonaws.com</td>
<td>Name</td>
<td>Username</td>
<td>Password</td>
<td></td>
</tr>
<tr>
<td>ec2-52-87-207-197.compute-1.amazonaws.com</td>
<td>Erica Voss</td>
<td>iqvia-ohdsi</td>
<td>!QVIAOH@DS18</td>
<td></td>
</tr>
</tbody>
</table>
How to Sign into the Remote Desktop

• If you get this page, select “Yes”
OHDSI in a Box – Ready
OHDSI’s Mission & Vision

To improve health by empowering a community to collaboratively generate the evidence that promotes better health decisions and better care.

A world in which observational research produces a comprehensive understanding of health and disease.

Join us on the journey

http://ohdsi.org
"What's the adherence to my drug in the data assets I own?"

Current Approach: “One Study – One Script”

Analytical method: Adherence to Drug

Application to data

Current solution:

One SAS or R script for each study

- Not scalable
- Not transparent
- Expensive
- Slow
- Prohibitive to non-expert routine use
Solution: Data Standardization Enables Systematic Research

- Adherence
- Mortality
- Source of Business
- Safety Signals

Ohdsi Tools

OMOP CDM

- North America
- Southeast Asia
- China
- Europe
- UK
- Japan
- India
- So Africa
- Switzerland
- Italy
- Israel

Standardized data
Why the CDM?

Ability to pursue cross-institutional collaborations

Write one program to run on multiple data assets

OMOP Vocabularies has greatly increased our ability to find relevant codes

You truly know your data if you convert it to the CDM

If you know a problem with your data, you can use the ETL to address it

Whole community of researchers across diverse organizations and countries

You can use standardized tools developed by OHDSI like ATLAS and the Patient Level Prediction Package

The CDM brings consistency to observational research through standardization of many of its components

Buy vs Build: leverage an entire community of technical and scientific capability for “free”

Takes observational research towards open science
ETL

• Extract, Transform, Load

• In order to get from our native/raw data into the OMOP CDM we need to design and develop and ETL process

• Goal in ETLing is to standardize the format and terminology

• This tutorial
  – Will teach you best practices around designing an ETL and CDM maintenance
  – Will not teach you how to program an ETL
ETL Process

Data experts and CDM experts together design the ETL

People with medical knowledge create the code mappings

All are involved in quality control

A technical person implements the ETL

OHDSI Tools

White Rabbit

Rabbit In a Hat

Usagi

White Rabbit

ACHILLES

DQD

Rabbit In a Hat
Chapter 6  Extract Transform Load

Chapter leads: Clair Blacketer & Erica Voss

6.1  Introduction

In order to get from the native/raw data to the OMOP Common Data Model (CDM) we have to create an extract, transform, and load (ETL) process. This process should restructure the data to the CDM, and add mappings to the Standardized Vocabularies, and is typically implemented as a set of automated scripts, for example SQL scripts. It is important that this ETL process is repeatable, so that it can be rerun whenever the source data is refreshed.

Creating an ETL is usually a large undertaking. Over the years, we have developed best practices, consisting of four major steps:

1. Data experts and CDM experts together design the ETL.
2. People with medical knowledge create the code mappings.
3. A technical person implements the ETL.
Hands On Exercises for Today

• Scan a database with White Rabbit

• Practice building an ETL document with Rabbit in a Hat

• Mapping Source Codes by with the OMOP Vocabulary and USAGI
Data experts and CDM experts together design the ETL.

People with medical knowledge create the code mappings.

All are involved in quality control.

A technical person implements the ETL.
A Patient’s Story: Lauren

“Every step of this painful journey I’ve had to convince everyone how much pain I was in.”

“My first surgery taught me that I had to be very patient with my recovery and very patient with myself in general.”

https://www.endometriosis-uk.org/laurens-story
What data do we have?

Lauren's Timeline

-3 Years
-2 Years
-1 Years

/ / / / -2 Weeks

/ / / / -3 Days

Day 0

Endometriosis

Endometriosis

Dysmenorrhea

Abdominal pain

Missed work

Acetaminophen

GP visit

Pelvic exam

Ultrasound

Cyst of ovary

Hospital Visit

Severe pain

Temp 103°F

CT Scan

Ambulance

Bloated abdomen

Ascites

Surgery

Endometrioma

-3 Days

-2 Weeks

-1 Years

-2 Years

-3 Years

Missed work

Acetaminophen

Acetaminophen

Acetaminophen

-3 Days

-2 Weeks

-1 Years

-2 Years

-3 Years

27
Data Format

• Synthea™ is a Synthetic Patient Population Simulator. The goal is to output synthetic, realistic (but not real), patient data and associated health records in a variety of formats.

• The resulting data is free from cost, privacy, and security restrictions. It can be used without restriction for a variety of secondary uses in academia, research, industry, and government (although a citation would be appreciated).

• https://github.com/synthetichealth/synthea

# Synthea Tables

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allergies.csv</td>
<td>Patient allergy data.</td>
</tr>
<tr>
<td>careplans.csv</td>
<td>Patient care plan data, including goals.</td>
</tr>
<tr>
<td>conditions.csv</td>
<td>Patient conditions or diagnoses.</td>
</tr>
<tr>
<td>encounters.csv</td>
<td>Patient encounter data.</td>
</tr>
<tr>
<td>imaging_studies.csv</td>
<td>Patient imaging metadata.</td>
</tr>
<tr>
<td>immunizations.csv</td>
<td>Patient immunization data.</td>
</tr>
<tr>
<td>medications.csv</td>
<td>Patient medication data.</td>
</tr>
<tr>
<td>observations.csv</td>
<td>Patient observations including vital signs and lab reports.</td>
</tr>
<tr>
<td>organizations.csv</td>
<td>Provider organizations including hospitals.</td>
</tr>
<tr>
<td>patients.csv</td>
<td>Patient demographic data.</td>
</tr>
<tr>
<td>procedures.csv</td>
<td>Patient procedure data including surgeries.</td>
</tr>
<tr>
<td>providers</td>
<td>Clinicians that provide patient care.</td>
</tr>
</tbody>
</table>
Raw Data

1 Patient
Lauren Data
Synthea Format

1000 Patient
Synthetic Data
Synthea Format
Tools help us get started . . .

**White Rabbit**
- performs a scan of the source data, providing detailed information on the tables, fields, and values that appear in a field

**Rabbit In a Hat**
- Uses White Rabbit scan to provide a graphical user interface to help build an ETL document
  - Does not generate code*

*But people are test driving this*
White Rabbit - Location
White Rabbit - Scan
White Rabbit - Scan

[Image of White Rabbit software interface]

- Min cell count: 5
- Max distinct values: 1,000
- Rows per table: 100,000
White Rabbit – Scan Report

• We already ran the scan on raw_synthea

• To open the scan while we review:
  – https://github.com/OHDSI/Tutorial-ETL
  – Materials ➔ WhiteRabbit ➔ ScanReport_raw_synthea.xlsx
  – Click “View Raw” to download the XLSX
### White Rabbit – Scan Report: raw_synthea

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table</td>
<td>Field</td>
<td>Type</td>
<td>Max lengt</td>
<td>N rows</td>
<td>N rows ch</td>
<td>Fraction emp</td>
</tr>
<tr>
<td>allergies</td>
<td>start</td>
<td>date</td>
<td>10</td>
<td>619</td>
<td>619</td>
<td>0</td>
</tr>
<tr>
<td>allergies</td>
<td>stop</td>
<td>date</td>
<td>10</td>
<td>619</td>
<td>619</td>
<td>0.904685</td>
</tr>
<tr>
<td>allergies</td>
<td>patient</td>
<td>character</td>
<td>36</td>
<td>619</td>
<td>619</td>
<td>0</td>
</tr>
<tr>
<td>allergies</td>
<td>encounter</td>
<td>character</td>
<td>36</td>
<td>619</td>
<td>619</td>
<td>0</td>
</tr>
<tr>
<td>allergies</td>
<td>code</td>
<td>character</td>
<td>9</td>
<td>619</td>
<td>619</td>
<td>0</td>
</tr>
<tr>
<td>allergies</td>
<td>description</td>
<td>character</td>
<td>24</td>
<td>619</td>
<td>619</td>
<td>0</td>
</tr>
<tr>
<td>careplans</td>
<td>id</td>
<td>character</td>
<td>36</td>
<td>2939</td>
<td>2939</td>
<td>0</td>
</tr>
<tr>
<td>careplans</td>
<td>start</td>
<td>date</td>
<td>10</td>
<td>2939</td>
<td>2939</td>
<td>0</td>
</tr>
<tr>
<td>careplans</td>
<td>stop</td>
<td>date</td>
<td>10</td>
<td>2939</td>
<td>2939</td>
<td>0.380061</td>
</tr>
<tr>
<td>careplans</td>
<td>patient</td>
<td>character</td>
<td>36</td>
<td>2939</td>
<td>2939</td>
<td>0</td>
</tr>
<tr>
<td>careplans</td>
<td>encounter</td>
<td>character</td>
<td>36</td>
<td>2939</td>
<td>2939</td>
<td>0</td>
</tr>
<tr>
<td>careplans</td>
<td>code</td>
<td>character</td>
<td>15</td>
<td>2939</td>
<td>2939</td>
<td>0</td>
</tr>
<tr>
<td>careplans</td>
<td>description</td>
<td>character</td>
<td>62</td>
<td>2939</td>
<td>2939</td>
<td>0</td>
</tr>
<tr>
<td>careplans</td>
<td>reason_cc</td>
<td>character</td>
<td>14</td>
<td>2939</td>
<td>2939</td>
<td>0.090507</td>
</tr>
<tr>
<td>careplans</td>
<td>reason_de</td>
<td>character</td>
<td>69</td>
<td>2939</td>
<td>2939</td>
<td>0.090507</td>
</tr>
<tr>
<td>condition: start</td>
<td>date</td>
<td>10</td>
<td>7898</td>
<td>7898</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>condition: stop</td>
<td>date</td>
<td>10</td>
<td>7898</td>
<td>7898</td>
<td>0.458091</td>
<td></td>
</tr>
<tr>
<td>condition: patient</td>
<td>character</td>
<td>36</td>
<td>7898</td>
<td>7898</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>condition: encounter</td>
<td>character</td>
<td>36</td>
<td>7898</td>
<td>7898</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>condition: code</td>
<td>character</td>
<td>7</td>
<td>7898</td>
<td>7898</td>
<td>0.545455</td>
<td></td>
</tr>
<tr>
<td>condition: description</td>
<td>character</td>
<td>80</td>
<td>7898</td>
<td>7898</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>encounter: id</td>
<td>character</td>
<td>36</td>
<td>34275</td>
<td>34275</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>encounter: start</td>
<td>date</td>
<td>10</td>
<td>34275</td>
<td>34275</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>encounter: stop</td>
<td>date</td>
<td>10</td>
<td>34275</td>
<td>34275</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
## White Rabbit – Scan Report: raw_synthea

<table>
<thead>
<tr>
<th>patients</th>
<th>id</th>
<th>character</th>
<th>36</th>
<th>1132</th>
<th>1132</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>patients</td>
<td>birthdate</td>
<td>date</td>
<td>10</td>
<td>1132</td>
<td>1132</td>
<td>0</td>
</tr>
<tr>
<td>patients</td>
<td>deathdate</td>
<td>date</td>
<td>10</td>
<td>1132</td>
<td>1132</td>
<td>0.893993</td>
</tr>
<tr>
<td>patients</td>
<td>ssn</td>
<td>character</td>
<td>11</td>
<td>1132</td>
<td>1132</td>
<td>0</td>
</tr>
<tr>
<td>patients</td>
<td>drivers</td>
<td>character</td>
<td>9</td>
<td>1132</td>
<td>1132</td>
<td>0.174912</td>
</tr>
<tr>
<td>patients</td>
<td>passport</td>
<td>character</td>
<td>10</td>
<td>1132</td>
<td>1132</td>
<td>0.218198</td>
</tr>
<tr>
<td>patients</td>
<td>prefix</td>
<td>character</td>
<td>4</td>
<td>1132</td>
<td>1132</td>
<td>0.188163</td>
</tr>
<tr>
<td>patients</td>
<td>first</td>
<td>character</td>
<td>15</td>
<td>1132</td>
<td>1132</td>
<td>0</td>
</tr>
<tr>
<td>patients</td>
<td>last</td>
<td>character</td>
<td>16</td>
<td>1132</td>
<td>1132</td>
<td>0</td>
</tr>
<tr>
<td>patients</td>
<td>suffix</td>
<td>character</td>
<td>3</td>
<td>1132</td>
<td>1132</td>
<td>0.992049</td>
</tr>
<tr>
<td>patients</td>
<td>maiden</td>
<td>character</td>
<td>16</td>
<td>1132</td>
<td>1132</td>
<td>0.725265</td>
</tr>
<tr>
<td>patients</td>
<td>marital</td>
<td>character</td>
<td>1</td>
<td>1132</td>
<td>1132</td>
<td>0.303887</td>
</tr>
<tr>
<td>patients</td>
<td>race</td>
<td>character</td>
<td>8</td>
<td>1132</td>
<td>1132</td>
<td>0</td>
</tr>
<tr>
<td>patients</td>
<td>ethnicity</td>
<td>character</td>
<td>16</td>
<td>1132</td>
<td>1132</td>
<td>0</td>
</tr>
<tr>
<td>patients</td>
<td>gender</td>
<td>character</td>
<td>1</td>
<td>1132</td>
<td>1132</td>
<td>0.001767</td>
</tr>
<tr>
<td>patients</td>
<td>birthplace</td>
<td>character</td>
<td>21</td>
<td>1132</td>
<td>1132</td>
<td>0</td>
</tr>
<tr>
<td>patients</td>
<td>address</td>
<td>character</td>
<td>36</td>
<td>1132</td>
<td>1132</td>
<td>0</td>
</tr>
<tr>
<td>patients</td>
<td>city</td>
<td>character</td>
<td>21</td>
<td>1132</td>
<td>1132</td>
<td>0</td>
</tr>
<tr>
<td>patients</td>
<td>state</td>
<td>character</td>
<td>13</td>
<td>1132</td>
<td>1132</td>
<td>0</td>
</tr>
<tr>
<td>patients</td>
<td>zip</td>
<td>character</td>
<td>5</td>
<td>1132</td>
<td>1132</td>
<td>0</td>
</tr>
<tr>
<td>W</td>
<td>X</td>
<td>Y</td>
<td>Z</td>
<td>AA</td>
<td>AB</td>
<td>AC</td>
</tr>
<tr>
<td>---</td>
<td>-----</td>
<td>-----------</td>
<td>-----</td>
<td>-------</td>
<td>------</td>
<td>------</td>
</tr>
<tr>
<td></td>
<td>marital</td>
<td>race</td>
<td>Frequency</td>
<td>Frequency</td>
<td>ethnicity</td>
<td>Frequency</td>
</tr>
<tr>
<td>M</td>
<td>622</td>
<td>white</td>
<td>846</td>
<td>irish</td>
<td>235</td>
<td>M</td>
</tr>
<tr>
<td></td>
<td>344</td>
<td>hispanic</td>
<td>112</td>
<td>italian</td>
<td>145</td>
<td>F</td>
</tr>
<tr>
<td>S</td>
<td>166</td>
<td>black</td>
<td>82</td>
<td>english</td>
<td>102</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>asian</td>
<td>70</td>
<td>puerto_rico</td>
<td>72</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>native</td>
<td>20</td>
<td>french</td>
<td>72</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>other</td>
<td>1</td>
<td>german</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unknown</td>
<td>1</td>
<td>chinese</td>
<td>51</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>polish</td>
<td>49</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>american</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>portugues</td>
<td>37</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>french_ca</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>african</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>west_indi</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>dominicar</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>american</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>russian</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>scottish</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>russian</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>asian_indi</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>mexican</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>swedish</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>central_ar</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>greek</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>immunizations</th>
<th>medications</th>
<th>observations</th>
<th>organization</th>
<th>patients</th>
</tr>
</thead>
</table>

Patients Tab
Now Your Turn:
Scan Lauren’s Data

- Click on WhiteRabbit shortcut
- Go into the WhiteRabbit folder
- Open bin\whiteRabbit.bat
Now Your Turn: Scan Lauren’s Data

• Connect to Lauren’s Data

  ![Database Configuration](image)

  - Data type: PostgreSQL
  - Server location: localhost/ETL
  - User name: postgres
  - Password: ohdsi
  - Database name: raw_lauren

• Test connection
Now Your Turn: Scan Lauren’s Data

- Go to the “Scan” tab
- Press “Add all in DB” button, set “Min cell count” to 0, and then “Scan tables”
- Open ScanReport.xlsx
Now Your Turn: Scan Lauren’s Data

• What is the most common condition Lauren has?

<table>
<thead>
<tr>
<th>Description</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dysmenorrhea</td>
<td>3</td>
</tr>
<tr>
<td>Endometriosis</td>
<td>1</td>
</tr>
<tr>
<td>Chronic pelvic pain of fe</td>
<td>1</td>
</tr>
<tr>
<td>Ascites</td>
<td>1</td>
</tr>
<tr>
<td>Fever</td>
<td>1</td>
</tr>
<tr>
<td>Cyst of left ovary</td>
<td>1</td>
</tr>
<tr>
<td>Abdominal distension, g</td>
<td>1</td>
</tr>
</tbody>
</table>

raw_lauren
White Rabbit

• White Rabbit creates an export of information about the source data

• The scan can be used to:
  – Learn about your source data
  – Used by Rabbit In a Hat
• Can read and display a White Rabbit scan document

• Provides a graphical interface to allow a user to connect source data to tables
Rabbit in a Hat

- We will use the ScanReport_raw_synthea.xlsx for this:
  - [https://github.com/OHDSI/Tutorial-ETL](https://github.com/OHDSI/Tutorial-ETL)
  - Materials → WhiteRabbit → ScanReport_raw_synthea.xlsx
  - Press the “Download” button

- Save it to the desktop
- Open it Rabbit in a Hat
Rabbit in a Hat

• The scan tells Rabbit in a Hat what is in the raw database
  – Orange Tables = Raw
  – Blue Tables = CDM
Rabbit in a Hat

Together

- person
- observation_period
- condition_occurrence

On your Own

- drug_exposure

Generate document
Resources

• Important links to keep in mind when working on an ETL:

  – CDM Wiki
    https://github.com/OHDSI/CommonDataModel/wiki
    Information about the CDM structure and conventions to follow can be found here

  – OHDSI Forums
    http://forums.ohdsi.org/
    http://forums.ohdsi.org/c/cdm-builders
    OHDSI is an active community, your questions may have already been asked on the forum however if not do not be afraid to ask it yourself!

  – Book of OHDSI: ETL Chapter
    The OHDSI community wrote the book to serve as a central knowledge repository for all things OHDSI.

  – THEMIS Working Group
    https://github.com/OHDSI/Themis
Rabbit in a Hat

• The full ETL document: https://ohdsi.github.io/ETL-Synthea/
Some Parting Thoughts On ETL

• Vocabulary will tell a source record where to go.
  – Example, just because it is a condition code and in a condition table does not mean it will end up in CONDITION_OCCURRENCE

  ICD9 783.1 - Abnormal weight gain

• STEM Table in Rabbit In a Hat
Upcoming enhancements

Sql generator #179

MaximMoinat merged 10 commits into GitHub: develop from thehyve:sql-generator 2 days ago

- Conversation 1
- Commits 10
- Checks 0
- Files changed 3

MaximMoinat commented 10 days ago • edited

Generates a SQL skeleton from the mappings defined in RiaH. One sql file per table mapping is created in selected folder. This skeleton helps implementers to fill in all the transformation logic. It has been very valuable for our team.

@clairblacketer Do you think this is more broadly applicable? Any suggested changes to the format?

An example of the skeleton (for brevity showing only a part of the target fields):

```sql
/*
Table comments and logic
*/

INSERT INTO person
(
    person_id, -- Auto-increment
    person_source_value, -- comments at target field level
    year_of_birth,
    month_of_birth,
    day_of_birth, -- For privacy purpose, do not capture day
    gender_concept_id,
    gender_source_value
)

SELECT
    -- [WARNING!] no source column found. See possible comment at the INSERT INTO
    NULL AS person_id,

    -- [VALUE COMMENT] Format: ABC-####
```

Reviewers
- schuemie
- blootvoets

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

Notifications
Unsubscribe
You're receiving notifications because you're watching this repository.
Upcoming enhancements

Additional scan report metrics

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
<th>N</th>
<th>O</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Table</td>
<td>Field</td>
<td>Type</td>
<td>Fraction empty</td>
<td>Unique values</td>
<td>Fraction Unique</td>
<td>Average</td>
<td>Standard Deviation</td>
<td>Min</td>
<td>q1</td>
<td>Median</td>
<td>q3</td>
</tr>
<tr>
<td>2</td>
<td>test</td>
<td>id</td>
<td>int</td>
<td>0%</td>
<td>20</td>
<td>100%</td>
<td>10.5</td>
<td>5.766281</td>
<td>1</td>
<td>6</td>
<td>11</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>test</td>
<td>gender</td>
<td>varchar</td>
<td>0%</td>
<td>2</td>
<td>10%</td>
<td>52.5</td>
<td>28.83141</td>
<td>5</td>
<td>30</td>
<td>55</td>
<td>80</td>
</tr>
<tr>
<td>4</td>
<td>test</td>
<td>age</td>
<td>int</td>
<td>0%</td>
<td>20</td>
<td>100%</td>
<td>56.5</td>
<td>124.6637</td>
<td>-200</td>
<td>-25</td>
<td>100</td>
<td>175</td>
</tr>
<tr>
<td>5</td>
<td>test</td>
<td>age2</td>
<td>int</td>
<td>0%</td>
<td>20</td>
<td>100%</td>
<td>1.4</td>
<td>0.961249</td>
<td>0.5</td>
<td>0.5</td>
<td>1.2</td>
<td>2.8</td>
</tr>
<tr>
<td>6</td>
<td>test</td>
<td>height</td>
<td>real</td>
<td>15%</td>
<td>3</td>
<td>15%</td>
<td>10.5</td>
<td>5.766281</td>
<td>1</td>
<td>6</td>
<td>11</td>
<td>16</td>
</tr>
<tr>
<td>7</td>
<td>test</td>
<td>race</td>
<td>varchar</td>
<td>20%</td>
<td>12</td>
<td>60%</td>
<td>5.5</td>
<td>3.6712</td>
<td>4</td>
<td>8</td>
<td>10</td>
<td>15</td>
</tr>
</tbody>
</table>

Concept id hints

<table>
<thead>
<tr>
<th>Tables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
</tr>
<tr>
<td>CDMV6.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>General information</td>
</tr>
<tr>
<td>Field name:</td>
</tr>
<tr>
<td>Field type:</td>
</tr>
<tr>
<td>Description:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
</tr>
<tr>
<td>CDMV6.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
</tr>
<tr>
<td>CDMV6.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
</tr>
<tr>
<td>CDMV6.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>8507</td>
<td>MALE</td>
</tr>
<tr>
<td>8532</td>
<td>FEMALE</td>
</tr>
<tr>
<td>8521</td>
<td>OTHER</td>
</tr>
<tr>
<td>8551</td>
<td>UNKNOWN</td>
</tr>
<tr>
<td>8570</td>
<td>AMBIGUOUS</td>
</tr>
<tr>
<td>1555843</td>
<td>Temporary</td>
</tr>
</tbody>
</table>
Upcoming enhancements

Additional scan report metrics

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
<th>N</th>
<th>O</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Table</td>
<td>Field</td>
<td>Type</td>
<td>Fraction empty</td>
<td>Unique values</td>
<td>Fraction Unique</td>
<td>Average</td>
<td>StandardDeviation</td>
<td>Min</td>
<td>q1</td>
<td>Median</td>
<td>q3</td>
</tr>
<tr>
<td>2</td>
<td>test</td>
<td>id</td>
<td>int</td>
<td>0%</td>
<td>20</td>
<td>100%</td>
<td>10.5</td>
<td>5.766281</td>
<td>1</td>
<td>6</td>
<td>11</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>test</td>
<td>gender</td>
<td>varchar</td>
<td>0%</td>
<td>2</td>
<td>100%</td>
<td>52.5</td>
<td>28.83141</td>
<td>5</td>
<td>30</td>
<td>55</td>
<td>80</td>
</tr>
<tr>
<td>4</td>
<td>test</td>
<td>age</td>
<td>int</td>
<td>0%</td>
<td>20</td>
<td>100%</td>
<td>1.4</td>
<td>0.961249</td>
<td>0.5</td>
<td>0.5</td>
<td>1.2</td>
<td>2.8</td>
</tr>
<tr>
<td>5</td>
<td>test</td>
<td>age2</td>
<td>int</td>
<td>0%</td>
<td>3</td>
<td>15%</td>
<td>0.5</td>
<td>0.5</td>
<td>2.8</td>
<td>2.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>test</td>
<td>height</td>
<td>real</td>
<td>15%</td>
<td>3</td>
<td>15%</td>
<td>1.4</td>
<td>0.961249</td>
<td>0.5</td>
<td>0.5</td>
<td>1.2</td>
<td>2.8</td>
</tr>
<tr>
<td>7</td>
<td>test</td>
<td>race</td>
<td>varchar</td>
<td>20%</td>
<td>2</td>
<td>0%</td>
<td>10.5</td>
<td>5.766281</td>
<td>1</td>
<td>6</td>
<td>11</td>
<td>16</td>
</tr>
</tbody>
</table>

Plus performance and user experience improvements

Concept id hints

**Source**

<table>
<thead>
<tr>
<th>patients</th>
<th>CDMV6.0</th>
<th>person</th>
</tr>
</thead>
</table>

**Fields**

<table>
<thead>
<tr>
<th>Source</th>
<th>CDMV6.0</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>*person_id</td>
<td></td>
</tr>
<tr>
<td>gender</td>
<td>*gender_concept_id</td>
<td>MALE</td>
</tr>
<tr>
<td>birthdate</td>
<td>*year_of_birth</td>
<td>OTHER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNKNOWN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AMBIGUOUS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Data experts and CDM experts together design the ETL

People with medical knowledge create the code mappings

All are involved in quality control

A technical person implements the ETL
Standardizing Terminologies

SOURCE_CODE
XYZ
i.e. ICPC-1 Dutch codes, ICD9, etc.

? 

STANDARD_CONCEPT_ID
123456789
i.e. SNOMED for conditions and RxNorm for drugs

• What is standardize:

1. TABLE_CONCEPT_ID
   standard concept the source code maps to, used for analysis

2. TABLE_SOURCE_CONCEPT_ID
   concept representation of the source code, helps maintain tie to raw data

• Ways to get a source code to standard code:

1. OMOP Vocabulary

2. USAGI
OMOP Vocab

• There are two standard queries to help us use the OMOP Vocabulary:
  – SOURCE_TO_STANDARD.sql
  – SOURCE_TO_SOURCE.sql

• [https://github.com/OHDSI/Tutorial-ETL](https://github.com/OHDSI/Tutorial-ETL)
  – Materials ➔ Queries
OMOP Vocab

• If your source data’s codes are in the OMOP Vocab you can use it to translate to a standard

• For example:
  – ICD9 → SNOMED
  – NDC → RxNORM
Mapping a Lauren Row to CONCEPT_ID

```
SELECT *
FROM RAW_LAUREN.CONDITIONS
WHERE ENCOUNTER = '70'
```

<table>
<thead>
<tr>
<th>START</th>
<th>STOP</th>
<th>PATIENT</th>
<th>ENCOUNTER</th>
<th>CODE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/6/2010</td>
<td></td>
<td>1</td>
<td>70</td>
<td>N94.6</td>
<td>Dysmenorrhea</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CONDITION_CONCEPT_ID</th>
<th>CONDITION_SOURCE_CONCEPT_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
WITH CTE_VOCAB_MAP AS (  
  SELECT c.concept_code AS SOURCE_CODE, c.concept_id AS SOURCE_CONCEPT_ID, c.concept_name AS SOURCE_CODE_DESCRIPTION,  
  c.vocabulary_id AS SOURCE_VOCABULARY_ID, c.domain_id AS SOURCE_DOMAIN_ID, c.CONCEPT_CLASS_ID AS SOURCE_CONCEPT_CLASS_ID,  
  c.VALID_START_DATE AS SOURCE_VALID_START_DATE, c.VALID_END_DATE AS SOURCE_VALID_END_DATE, c.INVALID_REASON AS SOURCE_INVALID_REASON,  
  c1.concept_id AS TARGET_CONCEPT_ID, c1.concept_name AS TARGET_CONCEPT_NAME, c1.vocabulary_id AS TARGET_VOCABULARY_ID,  
  c1.domain_id AS TARGET_DOMAIN_ID, c1.concept_class_id AS TARGET_CONCEPT_CLASS_ID, c1.INVALID_REASON AS TARGET_INVALID_REASON,  
  c1.standard_concept AS TARGET_STANDARD_CONCEPT  
  FROM CONCEPT C  
  JOIN CONCEPT_RELATIONSHIP CR  
  ON C.CONCEPT_ID = CR.CONCEPT_ID_1  
  AND CR.invalid_reason IS NULL  
  AND cr.relationship_id = 'Maps to'  
  JOIN CONCEPT C1  
  ON CR.CONCEPT_ID_2 = C1.CONCEPT_ID  
  AND C1.INVALID_REASON IS NULL  
  UNION  
  SELECT source_code, SOURCE_CONCEPT_ID, SOURCE_CODE_DESCRIPTION, source_vocabulary_id, c1.domain_id AS SOURCE_DOMAIN_ID, c2.CONCEPT_CLASS_ID AS SOURCE_CONCEPT_CLASS_ID,  
  c1.VALID_END_DATE AS SOURCE_VALID_END_DATE, stcm.INVALID_REASON AS SOURCE_INVALID_REASON, target_concept_id,  
  c2.CONCEPT_NAME AS TARGET_CONCEPT_NAME, target_vocabulary_id, c2.domain_id AS TARGET_DOMAIN_ID,  
  c2.concept_class_id AS TARGET_CONCEPT_CLASS_ID, c2.INVALID_REASON AS TARGET_INVALID_REASON,  
  c2.standard_concept AS TARGET_STANDARD_CONCEPT  
  FROM source_to_concept_map stcm  
  LEFT OUTER JOIN CONCEPT c1  
  ON c1.concept_id = stcm.source_concept_id  
  LEFT OUTER JOIN CONCEPT c2  
  ON c2.CONCEPT_ID = stcm.target_concept_id  
  WHERE stcm.INVALID_REASON IS NULL  
)  
SELECT TARGET_CONCEPT_ID, TARGET_CONCEPT_NAME, TARGET_DOMAIN_ID  
FROM CTE_VOCAB_MAP  
WHERE SOURCE_CODE  = 'N94.6'  
AND SOURCE_VOCABULARY_ID  = 'ICD10CM'  
AND TARGET_STANDARD_CONCEPT  = 'S'
WITH CTE_VOCAB_MAP AS (  
  SELECT c.concept_code AS SOURCE_CODE, c.concept_id AS SOURCE_CONCEPT_ID, c.concept_name AS SOURCE_CODE_DESCRIPTION,  
c.vocabulary_id AS SOURCE_VOCABULARY_ID, c.domain_id AS SOURCE_DOMAIN_ID, c.CONCEPT_CLASS_ID AS SOURCE_CONCEPT_CLASS_ID,  
c.VALID_START_DATE AS SOURCE_VALID_START_DATE, c.VALID_END_DATE AS SOURCE_VALID_END_DATE, c.INVALID_REASON AS SOURCE_INVALID_REASON,  
c1.concept_id AS TARGET_CONCEPT_ID, c1.concept_name AS TARGET_CONCEPT_NAME, c1.VOCABULARY_ID AS TARGET_VOCABULARY_ID,  
c1.domain_id AS TARGET_DOMAIN_ID, c1.concept_class_id AS TARGET_CONCEPT_CLASS_ID, c1.INVALID_REASON AS TARGET_INVALID_REASON,  
c1.standard_concept AS TARGET_STANDARD_CONCEPT  
FROM CONCEPT C  
JOIN CONCEPT_RELATIONSHIP CR  
  ON C.CONCEPT_ID = CR.CONCEPT_ID_1  
  AND CR.INVALID_REASON IS NULL  
  AND cr.relationship_id = 'Maps to'  
JOIN CONCEPT C1  
  ON CR.CONCEPT_ID_2 = C1.CONCEPT_ID  
  AND C1.INVALID_REASON IS NULL  
UNION  
SELECT source_code, SOURCE_CONCEPT_ID, SOURCE_CODE_DESCRIPTION,  
sOURCE_VOCABULARY_ID, SOURCE_DOMAIN_ID, SOURCE_CONCEPT_CLASS_ID, SOURCE_VALID_START_DATE,  
sOURCE_VALID_END_DATE, SOURCE_INVALID_REASON,  
target_concept_id, TARGET_CONCEPT_NAME, target_vocabulary_id,  
tARGET_DOMAIN_ID, TARGET_CONCEPT_CLASS_ID, TARGET_STANDARD_CONCEPT  
FROM source_to_concept_map stcm  
LEFT OUTER JOIN CONCEPT c1  
  ON c1.concept_id = stcm.source_concept_id  
LEFT OUTER JOIN CONCEPT c2  
  ON c2.concept_id = stcm.target_concept_id  
WHERE stcm.INVALID_REASON IS NULL  
)  
SELECT TARGET_CONCEPT_ID, TARGET_CONCEPT_NAME, TARGET_DOMAIN_ID  
FROM CTE_VOCAB_MAP  
WHERE SOURCE_CODE = 'N94.6'  
AND SOURCE_VOCABULARY_ID = 'ICD10CM'  
AND TARGET_STANDARD_CONCEPT = 'S'
WITH CTE_VOCAB_MAP AS (  
    SELECT c.concept_code AS SOURCE_CODE, c.concept_id AS SOURCE_CONCEPT_ID, c.concept_name AS SOURCE_CODE_DESCRIPTION,  
    c.vocabulary_id AS SOURCE_VOCABULARY_ID, c.domain_id AS SOURCE_DOMAIN_ID, c.concept_class_id AS SOURCE_CONCEPT_CLASS_ID,  
    c.VALID_START_DATE AS SOURCE_VALID_START_DATE, c.VALID_END_DATE AS SOURCE_VALID_END_DATE, c.INVALID_REASON AS SOURCE_INVALID_REASON,  
    c1.concept_id AS TARGET_CONCEPT_ID, c1.concept_name AS TARGET_CONCEPT_NAME, c1.VOCABULARY_ID AS TARGET_VOCABULARY_ID,  
    c1.domain_id AS TARGET_DOMAIN_ID, c1.concept_class_id AS TARGET_CONCEPT_CLASS_ID,  
    c1.INVALID_REASON AS TARGET_INVALID_REASON,  
    c1.standard_concept AS TARGET_STANDARD_CONCEPT  
    FROM CONCEPT C  
    JOIN CONCEPT_RELATIONSHIP CR  
    ON C.CONCEPT_ID = CR.CONCEPT_ID  
    AND CR.INVALID_REASON IS NULL  
    AND cr.relationship_id = 'Maps to'  
    JOIN CONCEPT C1  
    ON CR.CONCEPT_ID_2 = C1.CONCEPT_ID  
    AND C1.INVALID_REASON IS NULL  
    UNION  
    SELECT source_code, SOURCE_CONCEPT_ID, SOURCE_CODE_DESCRIPTION, source_vocabulary_id, c1.domain_id AS SOURCE_DOMAIN_ID, c2.concept_class_id AS SOURCE_CONCEPT_CLASS_ID,  
    c1.VALID_END_DATE AS SOURCE_VALID_END_DATE, stcm.INVALID_REASON AS SOURCE_INVALID_REASON, target_concept_id,  
    c2.concept_name AS TARGET_CONCEPT_NAME, target_vocabulary_id, c2.domain_id AS TARGET_DOMAIN_ID,  
    c2.concept_class_id AS TARGET_CONCEPT_CLASS_ID, c2.INVALID_REASON AS TARGET_INVALID_REASON,  
    c2.standard_concept AS TARGET_STANDARD_CONCEPT  
    FROM source_to_concept_map stcm  
    LEFT OUTER JOIN CONCEPT c1  
    ON c1.concept_id = stcm.source_concept_id  
    LEFT OUTER JOIN CONCEPT c2  
    ON c2.concept_id = stcm.target_concept_id  
    WHERE stcm.INVALID_REASON IS NULL  
)  
SELECT TARGET_CONCEPT_ID, TARGET_CONCEPT_NAME, TARGET_DOMAIN_ID  
FROM CTE_VOCAB_MAP  
WHERE SOURCE_CODE = 'N94.6'  
AND SOURCE_VOCABULARY_ID = 'ICD10CM'  
AND TARGET_STANDARD_CONCEPT = 'S'
WITH CTE_VOCAB_MAP AS ( 
  
  SELECT c.concept_code AS SOURCE_CODE, c.concept_id AS SOURCE_CONCEPT_ID, c.concept_name AS SOURCE_CODE_DESCRIPTION, 
  
c.vocabulary_id AS SOURCE_VOCABULARY_ID, c.domain_id AS SOURCE_DOMAIN_ID, c.concept_class_id AS SOURCE_CONCEPT_CLASS_ID, 
  
c.VALID_START_DATE AS SOURCE_VALID_START_DATE, c.VALID_END_DATE AS SOURCE_VALID_END_DATE, c.INVALID_REASON AS SOURCE_INVALID_REASON, 
  
c1.concept_id AS TARGET_CONCEPT_ID, c1.concept_name AS TARGET_CONCEPT_NAME, c1.vocabulary_id AS TARGET_VOCABULARY_ID, 
  
c1.domain_id AS TARGET_DOMAIN_ID, c1.concept_class_id AS TARGET_CONCEPT_CLASS_ID, c1.INVALID_REASON AS TARGET_INVALID_REASON, 
  
c1.standard_concept AS TARGET_STANDARD_CONCEPT 
FROM CONCEPT c 
JOIN CONCEPT_RELATIONSHIP cr 
  ON c.concept_id = cr.concept_id_1 
  AND cr.invalid_reason IS NULL 
  AND cr.relationship_id = 'Maps to' 
JOIN CONCEPT c1 
  ON cr.concept_id_2 = c1.concept_id 
  AND c1.INVALID_REASON IS NULL 
UNION 
  
SELECT source_code, SOURCE_CODE_DESCRIPTION, source_vocabulary_id, c1.domain_id AS SOURCE_DOMAIN_ID, c2.concept_class_id AS SOURCE_CONCEPT_CLASS_ID, 
  
c1.VALID_START_DATE AS SOURCE_VALID_START_DATE, c1.VALID_END_DATE AS SOURCE_VALID_END_DATE, c1.INVALID_REASON AS SOURCE_INVALID_REASON, 
  
target_concept_id, target_vocabulary_id, c2.domain_id AS TARGET_DOMAIN_ID, 
  
c2.INVALID_REASON AS TARGET_INVALID_REASON, 
  
c2.standard_concept AS TARGET_STANDARD_CONCEPT 
FROM source_to_concept_map stcm 
LEFT OUTER JOIN CONCEPT c1 
  ON stcm.source_concept_id = c1.concept_id 
LEFT OUTER JOIN CONCEPT c2 
  ON stcm.target_concept_id = c2.concept_id 
WHERE stcm.INVALID_REASON IS NULL ) 

SELECT TARGET_CONCEPT_ID, TARGET_CONCEPT_NAME, TARGET_DOMAIN_ID 
FROM CTE_VOCAB_MAP 
WHERE SOURCE_CODE = 'N94.6' 
AND SOURCE_VOCABULARY_ID = 'ICD10CM' 
AND TARGET_STANDARD_CONCEPT = 'S' 

Look up your source Code here
### Mapping a Lauren Row to CONCEPT_ID:
Source to Standard

<table>
<thead>
<tr>
<th>START</th>
<th>STOP</th>
<th>PATIENT</th>
<th>ENCOUNTER</th>
<th>CODE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/6/2010</td>
<td></td>
<td>1</td>
<td>70</td>
<td>N94.6</td>
<td>Dysmenorrhea</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TARGET_CONCEPT_ID</th>
<th>TARGET_CONCEPT_NAME</th>
<th>TARGET_DOMAIN_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>194696</td>
<td>Dysmenorrhea</td>
<td>Condition</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CONDITION_CONCEPT_ID</th>
<th>CONDITION_SOURCE_CONCEPT_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>194696</td>
<td>194696</td>
</tr>
</tbody>
</table>
WITH CTE_VOCAB_MAP AS (  
  SELECT c.concept_code AS SOURCE_CODE, c.concept_id AS SOURCE_CONCEPT_ID,  
  c.CONCEPT_NAME AS SOURCE_CODE_DESCRIPTION, c.vocabulary_id AS SOURCE_VOCABULARY_ID,  
  c.domain_id AS SOURCE_DOMAIN_ID, c.concept_class_id AS SOURCE_CONCEPT_CLASS_ID,  
  c.VALID_START_DATE AS SOURCE_VALID_START_DATE, c.VALID_END_DATE AS SOURCE_VALID_END_DATE,  
  c.invalid_reason AS SOURCE_INVALID_REASON, c.concept_id as TARGET_CONCEPT_ID,  
  c.concept_name AS TARGET_CONCEPT_NAME, c.vocabulary_id AS TARGET_VOCABULARY_ID,  
  c.domain_id AS TARGET_DOMAIN_ID, c.concept_class_id AS TARGET_CONCEPT_CLASS_ID,  
  c.INVALID_REASON AS TARGET_INVALID_REASON, c.STANDARD_CONCEPT AS TARGET_STANDARD_CONCEPT  
FROM CONCEPT c  
  UNION  
  SELECT source_code, SOURCE_CONCEPT_ID, SOURCE_CODE_DESCRIPTION, source_vocabulary_id,  
  c1.domain_id AS SOURCE_DOMAIN_ID, c2.CONCEPT_CLASS_ID AS SOURCE_CONCEPT_CLASS_ID,  
  c1.VALID_START_DATE AS SOURCE_VALID_START_DATE, c1.VALID_END_DATE AS SOURCE_VALID_END_DATE,  
  stcm.INVALID_REASON AS SOURCE_INVALID_REASON,target_concept_id,  
  c2.CONCEPT_NAME AS TARGET_CONCEPT_NAME, target_vocabulary_id, c2.domain_id AS TARGET_DOMAIN_ID,  
  c2.concept_class_id AS TARGET_CONCEPT_CLASS_ID, c2.INVALID_REASON AS TARGET_INVALID_REASON,  
  c2.standard_concept AS TARGET_STANDARD_CONCEPT  
FROM source_to_concept_map stcm  
  LEFT OUTER JOIN CONCEPT c1  
    ON c1.concept_id = stcm.source_concept_id  
  LEFT OUTER JOIN CONCEPT c2  
    ON c2.concept_id = stcm.target_concept_id  
WHERE stcm.INVALID_REASON IS NULL  
  )  
SELECT *  
FROM CTE_VOCAB_MAP  
WHERE SOURCE_CODE = 'N94.6'  
AND SOURCE_VOCABULARY_ID = 'ICD10CM'
Source to Source

WITH CTE_VOCAB_MAP AS (  
    SELECT c.concept_code AS SOURCE_CODE, c.concept_id AS SOURCE_CONCEPT_ID,  
           c.CONCEPT_NAME AS SOURCE_CODE_DESCRIPTION, c.vocabulary_id AS SOURCE_VOCABULARY_ID,  
           c.domain_id AS SOURCE_DOMAIN_ID, c.concept_class_id AS SOURCE_CONCEPT_CLASS_ID,  
           c.VALID_START_DATE AS SOURCE_VALID_START_DATE, c.VALID_END_DATE AS SOURCE_VALID_END_DATE,  
           c.invalid_reason AS SOURCE_INVALID_REASON, c.concept_id AS TARGET_CONCEPT_ID,  
           c.concept_name AS TARGET_CONCEPT_NAME, c.vocabulary_id AS TARGET_VOCABULARY_ID,  
           c.domain_id AS TARGET_DOMAIN_ID, c.concept_class_id AS TARGET_CONCEPT_CLASS_ID,  
           cINVALID_REASON AS TARGET_INVALID_REASON, c.STANDARD_CONCEPT AS TARGET_STANDARD_CONCEPT  
    FROM CONCEPT c
    UNION  
    SELECT source_code, SOURCE_CONCEPT_ID, SOURCE_CODE_DESCRIPTION, source_vocabulary_id,  
           c1.domain_id AS SOURCE_DOMAIN_ID, c2.CONCEPT_CLASS_ID AS SOURCE_CONCEPT_CLASS_ID,  
           c1.VALID_START_DATE AS SOURCE_VALID_START_DATE, c1.VALID_END_DATE AS SOURCE_VALID_END_DATE,  
           c1.INVALID_REASON, target_concept_id,  
           c1.concept_name, target_vocabulary_id, c2.domain_id AS TARGET_DOMAIN_ID,  
           c2.CONCEPT_CLASS_ID, c2.INVALID_REASON AS TARGET_INVALID_REASON,  
           c2.STANDARD_CONCEPT  
    FROM source_to_concept_map stcm  
    LEFT OUTER JOIN CONCEPT c1 ON c1.concept_id = stcm.source_concept_id  
    LEFT OUTER JOIN CONCEPT c2 ON c2.concept_id = stcm.target_concept_id  
    WHERE stcm.INVALID_REASON IS NULL
)

SELECT *  
FROM CTE_VOCAB_MAP  
WHERE SOURCE_CODE = 'N94.6'  
AND SOURCE_VOCABULARY_ID = 'ICD10CM'
Mapping a Lauren Row to CONCEPT_ID:
Source to Source

<table>
<thead>
<tr>
<th>START</th>
<th>STOP</th>
<th>PATIENT</th>
<th>ENCOUNTER</th>
<th>CODE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/6/2010</td>
<td></td>
<td>1</td>
<td>70</td>
<td>N94.6</td>
<td>Dysmenorrhea</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TARGET_CONCEPT_ID</th>
<th>TARGET_CONCEPT_NAME</th>
<th>TARGET_DOMAIN_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>35209488</td>
<td>Dysmenorrhea, unspecified</td>
<td>Condition</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CONDITION_CONCEPT_ID</th>
<th>CONDITION_SOURCE_CONCEPT_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>194696</td>
<td>35209488</td>
</tr>
</tbody>
</table>
Mapping Source Codes – Your turn

• Let’s open PostgreSQL
  – Open up pgAdmin4 using the icon on the task bar
  – Expand the server list and right-click on PostgreSQL 10 and choose Connect Server from the drop-down menu
  – When it asks for a password, type in ohdsi
Mapping Source Codes – Your turn

• Open up to and select the CDM (which has a copy of the vocab)

• Tools → Query Tool

• Type the following and hit F5 to run:

```
SET SEARCH_PATH TO CDM_SYNTHAEA_V1;
```
### Mapping Source Codes – Your turn

<table>
<thead>
<tr>
<th>CODE</th>
<th>DESCRIPTION</th>
<th>CODE TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>C83.3</td>
<td>Diffuse large B-cell lymphoma</td>
<td>ICD10 (not ICD10CM)</td>
</tr>
</tbody>
</table>

Mapping Source Codes – Your turn

<table>
<thead>
<tr>
<th>CODE</th>
<th>DESCRIPTION</th>
<th>CODE TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>C83.3</td>
<td>Diffuse large B-cell lymphoma</td>
<td>ICD10 (not ICD10CM)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CONDITION_CONCEPT_ID</th>
<th>CONDITION_SOURCE_CONCEPT_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>4300704</td>
<td>1567654</td>
</tr>
</tbody>
</table>

https://github.com/OHDSI/Tutorial-ETL/tree/master/materials/Queries
What do you do with the mapping information?

<table>
<thead>
<tr>
<th>Destination Field</th>
<th>Source field</th>
<th>Logic</th>
<th>Comment field</th>
</tr>
</thead>
<tbody>
<tr>
<td>person_id</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gender_concept_id</td>
<td>gender</td>
<td>When gender = 'M' then set gender_concept_id to 8507, when gender = 'F' then set to 8532</td>
<td>Drop any rows with missing/unknown gender.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Destination Field</th>
<th>Source field</th>
<th>Logic</th>
<th>Comment field</th>
</tr>
</thead>
<tbody>
<tr>
<td>condition_concept_id</td>
<td>code</td>
<td>Use code to lookup target_concept_id in SOURCE_TO_STANDARD_VOCAB_MAP: select v.target_concept_id from conditions c join source_to_standard_vocab_map v on v.source_code = c.code and v.target_domain_id = 'Condition' and v.target_standard_concept = 'S' and v.source_vocabulary_id in ('ICD10CM')</td>
<td></td>
</tr>
</tbody>
</table>
• When the Vocabulary does not have your source codes you will need to create a map to OMOP Vocabulary Concepts

• Usagi is Japanese for rabbit and was named after the first mapping exercise it was used for; mapping source codes used in a Japanese dataset into OMOP Vocabulary concepts

• Usagi software tool to help with process of mapping source codes to OMOP concepts

• Usagi performs text similarity between your source codes and what is in the OMOP Vocabulary
Usagi Process

1. Get a copy of the Vocabulary from ATHENA

2. Download Usagi

3. Have Usagi build an index on the Vocabulary

4. Load your source codes and let Usagi process them

5. Review and update suggest mappings with someone who has medical knowledge

6. Export codes into the SOURCE_TO_CONCEPT_MAP
Usagi Process

1. Get a copy of the **Vocabulary** from ATHENA

http://athena.ohdsi.org
Usagi Process

1. Get a copy of the **Vocabulary** from ATHENA
Usagi Process

2. Download Usagi

https://github.com/OHDSI/Usagi
Usagi Process

3. Have Usagi **build an index** on the Vocabulary
4. **Load your source codes, let Usagi process them**

- Generate an XLSX of **distinct codes** from source system with descriptions and frequency

- If the codes are not in English, use Google Translate to convert
Usagi Process

4. Load your source codes, let Usagi process them

- Import the codes into Usagi
Usagi Process

5. **Review and update suggest mappings with someone who has medical knowledge**

![Usagi Process Interface](image)
5. Review and update suggest mappings with someone who has medical knowledge
Usagi Process

5. Review and update suggest mappings with someone who has medical knowledge
Usagi Process

5. Review and update suggest mappings with someone who has medical knowledge
Usagi Process

5. Review and update suggest mappings with someone who has medical knowledge

• It may be valuable to sort on “Match Score”; reviewing codes that Usagi is most confident on first may quickly knock out a significant chunk of codes

• Sorting on “Frequency” is valuable, spending more effort on frequent codes versus non-frequent is important

• It is okay to map to zero or 0 – “No matching concept”

• A source code might end up being mapped to two concepts

• You might have what the system considers one domain but the OMOP Vocabulary lumps into another domain
Usagi Process

6. Export codes into the SOURCE_TO_CONCEPT_MAP

- After you have completed, you will export the relationships

- When exporting you will give a Vocabulary ID, for example JNJ_JMDC_PROVIDERS would signify a Johnson & Johnson map for the database JMDC for provider codes.

<table>
<thead>
<tr>
<th>source_code</th>
<th>source_concept_id</th>
<th>source_vocab_id</th>
<th>source_code_description</th>
<th>target_concept_id</th>
<th>target_vocab_id</th>
<th>valid_start_date</th>
<th>valid_end_date</th>
<th>invalid_reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>R74.02</td>
<td>0</td>
<td>TEST VOCAB</td>
<td>Acute pharyngitis</td>
<td>25297</td>
<td>SNOMED</td>
<td>1/1/1970</td>
<td>12/31/2099</td>
<td></td>
</tr>
</tbody>
</table>

R74.02 - Acute pharyngitis = 25297 - Acute pharyngitis
Usagi Process

6. **Export codes into the SOURCE_TO_CONCEPT_MAP**

- You then load your generated maps into the empty Vocabulary table.
Usagi – Your Turn

1. Get a copy of the Vocabulary from ATHENA

2. Download Usagi

3. Have Usagi build an index on the Vocabulary

4. Load your source codes and let Usagi process them

5. Review and update suggest mappings with someone who has medical knowledge

6. Export codes into the SOURCE_TO_CONCEPT_MAP
Now Your Turn:
Open Usagi

- Click on Usagi shortcut
- Go into the Usagi-1.1.6 folder
- Open Usagi.jar
Usagi – Your Turn

• We have provided a small subset of codes to try to map

  https://github.com/OHDSI/
  Tutorial-ETL/

  -> Materials -> Usagi ->
  DUTCH_ICPC_CONDITION_CODES_TO_MAP.xlsx

• These condition codes are in Dutch ICPC codes and need to be mapped to standard concepts
Usagi – Your Turn

• Your mission:
  – Download the codes to map
  – Translate codes to English
  – Import codes into Usagi
  – Map to standard concepts
  – Export SOURCE_TO_CONCEPT_MAP table

• For help review the User Guide:
Usagi – Your Turn

• What CONCEPT_ID do you map “Dermatomycosis (s)” to?

<table>
<thead>
<tr>
<th>Source Term</th>
<th>Concept ID</th>
<th>Concept Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dermatomycosis (s)</td>
<td>135473</td>
<td>Dermatophytosis</td>
</tr>
<tr>
<td>Dermatophytosis</td>
<td>137213</td>
<td>Dermal mycosis</td>
</tr>
</tbody>
</table>

a fungal infection of the skin, especially by a dermatophyte

fungal infection of the skin, especially by a dermatophyte
# Usagi – Your Turn

<table>
<thead>
<tr>
<th>Source term</th>
<th>Freque...</th>
<th>ICP...</th>
<th>Match score</th>
<th>Concept ID</th>
<th>Concept name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Immunization / preventive medication</td>
<td>1000000</td>
<td>Imm...</td>
<td>0.70</td>
<td>4144375</td>
<td>Active immunization</td>
</tr>
<tr>
<td>Cough</td>
<td>880000</td>
<td>Hoe...</td>
<td>1.00</td>
<td>254761</td>
<td>Cough</td>
</tr>
<tr>
<td>Acute pharyngitis</td>
<td>800000</td>
<td>Acut...</td>
<td>1.00</td>
<td>25297</td>
<td>Acute pharyngitis</td>
</tr>
<tr>
<td>Acute upper respiratory tract infection</td>
<td>800000</td>
<td>Acut...</td>
<td>1.00</td>
<td>257011</td>
<td>Acute upper respiratory infection</td>
</tr>
<tr>
<td>No illness</td>
<td>500000</td>
<td>Gee...</td>
<td>0.82</td>
<td>0</td>
<td>Unmapped</td>
</tr>
<tr>
<td>Cystitis / urinary tract infections</td>
<td>500000</td>
<td>Cysti...</td>
<td>0.71</td>
<td>81902</td>
<td>Urinary tract infectious disease</td>
</tr>
<tr>
<td>Acute bronchitis / bronchiolitis</td>
<td>300000</td>
<td>Acut...</td>
<td>0.84</td>
<td>260125</td>
<td>Acute bronchiolitis</td>
</tr>
<tr>
<td>being overweight</td>
<td>100000</td>
<td>over...</td>
<td>0.88</td>
<td>437525</td>
<td>Overweight</td>
</tr>
<tr>
<td>Pregnancy (confirmed)</td>
<td>100000</td>
<td>Zwa...</td>
<td>0.84</td>
<td>4299535</td>
<td>Pregnant</td>
</tr>
<tr>
<td>Dermatomycosis (s)</td>
<td>100000</td>
<td>Der...</td>
<td>0.81</td>
<td>137213</td>
<td>Dermal mycosis</td>
</tr>
<tr>
<td>Other disease (s) musculoskeletal system</td>
<td>1000000</td>
<td>Ande...</td>
<td>0.77</td>
<td>4244662</td>
<td>Disorder of musculoskeletal system</td>
</tr>
<tr>
<td>episode on initiative third</td>
<td>1</td>
<td>epis...</td>
<td>0.36</td>
<td>0</td>
<td>Unmapped</td>
</tr>
</tbody>
</table>
Data experts and CDM experts together design the ETL

People with medical knowledge create the code mappings

All are involved in quality control

A technical person implements the ETL
ETL Implementation

There are multiple tools available to implement your ETL

In this example we created a builder using SQL and R, though your choice will largely depend on the size and complexity of the ETL design
ETL Implementation

General Flow of Implementation

A good rule of thumb is to always create the PERSON table first

The VISIT_OCCURRENCE table must be created before the standardized clinical data tables as they all refer to the VISIT_OCCURRENCE_ID

- person
- observation_period
- visit_occurrence
  - condition_occurrence
  - drug_exposure
  - measurement
  - observation
  - procedure_occurrence
  - Additional clinical data tables...
CDM Version 6 Key Domains

**Standardized clinical data**
- Person
  - Observation_period
  - Visit_occurrence
    - Visit_detail
  - Condition_occurrence
  - Drug_exposure
  - Procedure_occurrence
  - Device_exposure
  - Measurement
  - Note
    - Note_NLP
  - Survey_conduct
  - Observation
  - Specimen
  - Fact_relationship

**Standardized derived elements**
- Condition_era
- Drug_era
- Dose_era

**Standardized health system data**
- Location
- Location_history
- Care_site
- Provider

**Standardized health economics**
- Cost
- Payer_plan_period

**Standardized vocabularies**
- Concept
- Vocabulary
- Domain
- Concept_class
- Concept_relationship
- Relationship
- Concept_synonym
- Concept_ancestor
- Source_to_concept_map
- Drug_strength

**Standardized metadata**
- CDM_source
- Metadata

**Results Schema**
- Cohort
- Cohort_definition
ETL Implementation

First, let us review the logic we decided on for how the PERSON table should be created.

Navigate in your browser to: https://ohdsi.github.io/ETL-Synthea/Person.html
First, let’s review the logic we decided on for how the PERSON table should be created.

**Gender:**
- gender_concept_id: When gender = ‘M’ then set gender_concept_id to 8507, when gender = ‘F’ then set to 8532
- gender: Drop any rows with missing/unknown gender.

**Birthdate:**
- year_of_birth: Take year from birthdate
- month_of_birth: Take month from birthdate
- day_of_birth: Take day from birthdate
- birth_datetime: With midnight as time 00:00:00

**Race:**
- race_concept_id: When race = ‘WHITE’ then set as 8527, when race = ‘BLACK’ then set as 8516, when race = ‘ASIAN’ then set as 8515, otherwise set as 0

**Ethnicity:**
- ethnicity_concept_id: When race = ‘HISPANIC’, or when ethnicity in (‘CENTRAL_AMERICAN’, ‘DOMINICAN’, ‘MEXICAN’, ‘PUERTO_RICAN’, ‘SOUTH_AMERICAN’ ) then set as 38003563, otherwise set as 0
ETL Implementation

How should the PERSON table logic be implemented in SQL?

To open the query while we review:

https://github.com/OHDSI/Tutorial-ETL
Materials → Implementation →
Insert_Person_Lauren.sql

You can either view it directly in GitHub or download it and open it in pgAdmin4
ETL Implementation

How should the PERSON table logic be implemented in SQL?

```sql
truncate cdm_lauren.person;
insert into cdm_lauren.person (person_id, ... ethnicity_source_concept_id)
select row_number() over (order by p.id) as person_id,
case upper(p.gender)
    when 'M' then 8507
    when 'F' then 8532
end as gender_concept_id,
date_part('year', p.birthdate) as year_of_birth,
date_part('month', p.birthdate) as month_of_birth,
date_part('day', p.birthdate) as day_of_birth,
p.birthdate as birth_datetime,
case upper(p.race)
    when 'WHITE' then 8527
    when 'BLACK' then 8516
    when 'ASIAN' then 8515
else 0
end as race_concept_id,
case
    when upper(p.race) = 'HISPANIC'
    then 38003563 else 0
end as ethnicity_concept_id,
...
ETL Implementation

Let’s review the logic we decided on for how the PERSON table should be created.

### Gender:

<table>
<thead>
<tr>
<th>field</th>
<th>type</th>
<th>condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender_concept_id</td>
<td>gender</td>
<td>When gender = ‘M’ then set gender_concept_id to 8507, when gender = ‘F’ then set to 8532</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Drop any rows with missing/unknown gender.</td>
</tr>
</tbody>
</table>

### Birthdate:

<table>
<thead>
<tr>
<th>field</th>
<th>type</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>year_of_birth</td>
<td>birthdate</td>
<td>Take year from birthdate</td>
</tr>
<tr>
<td>month_of_birth</td>
<td>birthdate</td>
<td>Take month from birthdate</td>
</tr>
<tr>
<td>day_of_birth</td>
<td>birthdate</td>
<td>Take day from birthdate</td>
</tr>
<tr>
<td>birth_datetime</td>
<td>birthdate</td>
<td>With midnight as time 00:00:00</td>
</tr>
</tbody>
</table>

### Race:

<table>
<thead>
<tr>
<th>field</th>
<th>type</th>
<th>condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>race_concept_id</td>
<td>race</td>
<td>When race = ‘WHITE’ then set as 8527, when race = ‘BLACK’ then set as 8516, when race = ‘ASIAN’ then set as 8515, otherwise set as 0</td>
</tr>
</tbody>
</table>

### Ethnicity:

<table>
<thead>
<tr>
<th>field</th>
<th>type</th>
<th>condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ethnicity_concept_id</td>
<td>race_ethnicity</td>
<td>When race = ‘HISPANIC’, or when ethnicity in (‘CENTRAL_AMERICAN’, ‘DOMINICAN’, ‘MEXICAN’, ‘PUERTO_RICAN’, ‘SOUTH_AMERICAN’ ) then set as 38003563, otherwise set as 0</td>
</tr>
</tbody>
</table>
How should the PERSON table logic be implemented in SQL?

```sql
truncate cdm_lauren.person;
insert into cdm_lauren.person (  
    person_id,
    ...
    ethnicity_source_concept_id
)  
select  
    new_number(order(order by p_id) as person_id  
    case upper(p.gender)  
    when 'M' then 8507
    when 'F' then 8532
    end as gender_concept_id,
    date_part('year', p.birthday) as year_of_birth,
    ...  
    when 'BLACK' then 8518
    when 'ASIAN' then 8515
    else 0
    end as race_concept_id,
    case  
    when upper(p.race) = 'HISPANIC'
    then 38003563 else 0
    end as ethnicity_concept_id,
    ...
```

Gender

<table>
<thead>
<tr>
<th>gender_concept_id</th>
<th>gender</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8507</td>
<td>M</td>
<td>When gender = 'M' then set gender_concept_id to 8507, when gender = 'F' then set to 8532</td>
</tr>
<tr>
<td>8532</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>8518</td>
<td>BLACK</td>
<td>Drop any rows with missing/unknown gender.</td>
</tr>
<tr>
<td>8515</td>
<td>ASIAN</td>
<td></td>
</tr>
<tr>
<td>38003563</td>
<td>'HISPANIC'</td>
<td></td>
</tr>
</tbody>
</table>
How should the PERSON table logic be implemented in SQL?

```sql
truncate cdm_lauren.person;
insert into cdm_lauren.person (person_id, ... ethnicity_source_concept_id )
select row_number(order by p.id) as person_id,
    case upper(p.gender)
        when 'M' then 8507
        when 'F' then 8532
    end as gender_concept_id,
    date_part('year', p.birthdate) as year_of_birth,
    ... as race_concept_id,
    case when upper(p.race) = 'HISPANIC' then 38003563 else 0 end as ethnicity_concept_id,
    ...
```

Gender

- When gender = ‘M’ then set gender_concept_id to 8507, when gender = ‘F’ then set to 8532

- Drop any rows with missing/unknown gender.
ETL Implementation

How should the PERSON table logic be implemented in SQL?

```sql
... end as gender_concept_id,
date_part('year', p.birthdate) as year_of_birth,
date_part('month', p.birthdate) as month_of_birth,
date_part('day', p.birthdate) as day_of_birth,
p.birthdate as birth_datetime,
case upper(p.race)
  when 'WHITE' then 8527
  when 'BLACK' then 8516
  when 'ASIAN' then 8515
else 0
end as race_concept_id,
case
  when upper(p.race) = 'HISPANIC'
    when gender = 'M' then set gender_concept_id to 8507, when gender = 'F' then set to 8532
  end

from raw_lauren.patients p
where p.gender is not null;
```
Let’s review the logic we decided on for how the PERSON table should be created.

<table>
<thead>
<tr>
<th>Gender:</th>
<th>Birthdate:</th>
<th>Race:</th>
<th>Ethnicity:</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender_concept_id</td>
<td>gender</td>
<td>year_of_birth</td>
<td>birthdate</td>
</tr>
<tr>
<td>When gender = ‘M’ then set</td>
<td>When gender = ‘M’ then set</td>
<td>Take year from birthdate</td>
<td>Take month from birthdate</td>
</tr>
<tr>
<td>gender_concept_id to 8507, when</td>
<td>gender_concept_id to 8532</td>
<td>Take day from birthdate</td>
<td>Take day from birthdate</td>
</tr>
<tr>
<td>gender = ‘F’ then set to 8532</td>
<td></td>
<td>With midnight as time 00:00:00</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Drop any rows with missing/unknown gender.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ETL Implementation

How should the PERSON table logic be implemented in SQL?

```sql
truncate cdm_lauren.person;
insert into cdm_lauren.person (person_id, ... ethnicity_source_concept_id
)
select row_number() over(order by p.id) as person_id,
case upper(p.gender)
    when 'M' then 8507
    when 'F' then 8532
end as gender_concept_id,
    date_part('year', p.birthdate) as year_of_birth,
    date_part('month', p.birthdate) as month_of_birth,
    date_part('day', p.birthdate) as day_of_birth,
    p.birthdate as birth_datetime,
    case when year_of_birth = 1900 then 38003563 else 0
    end as ethnicity_concept_id,
    ...
```
ETL Implementation

How should the PERSON table logic be implemented in SQL?

```sql
truncate cdm_lauren.person;
insert into cdm_lauren.person (person_id, ...
  ethnicity_source_concept_id
)

select row_number() over(order by p.id) as person_id,
case upper(p.gender)
  when 'M' then 8507
  when 'F' then 8532
end as gender_concept_id,
date_part('year', p.birthdate) as year_of_birth,
date_part('month', p.birthdate) as month_of_birth,
date_part('day', p.birthdate) as day_of_birth,
p.birthdate as birth_datetime,

table |
------|
year_of_birth | birthdate | Take year from birthdate
month_of_birth | birthdate | Take month from birthdate
day_of_birth | birthdate | Take day from birthdate
birth_datetime | birthdate | With midnight as time 00:00:00

then 38003563 else 0
end as ethnicity_concept_id,
...
Let's review the logic we decided on for how the PERSON table should be created.

## Gender:

<table>
<thead>
<tr>
<th>gender_concept_id</th>
<th>gender</th>
<th>When gender = 'M' then set gender_concept_id to 8507, when gender = 'F' then set to 8532</th>
<th>Drop any rows with missing/unknown gender</th>
</tr>
</thead>
</table>

## Birthdate:

<table>
<thead>
<tr>
<th>year_of_birth</th>
<th>birthdate</th>
<th>Take year from birthdate</th>
</tr>
</thead>
<tbody>
<tr>
<td>month_of_birth</td>
<td>birthdate</td>
<td>Take month from birthdate</td>
</tr>
<tr>
<td>day_of_birth</td>
<td>birthdate</td>
<td>Take day from birthdate</td>
</tr>
<tr>
<td>birth_datetime</td>
<td>birthdate</td>
<td>With midnight as time 00:00:00</td>
</tr>
</tbody>
</table>

## Race:

<table>
<thead>
<tr>
<th>race_concept_id</th>
<th>race</th>
<th>When race = 'WHITE' then set as 8527, when race = 'BLACK' then set as 8516, when race = 'ASIAN' then set as 8515, otherwise set as 0</th>
</tr>
</thead>
</table>

## Ethnicity:

<table>
<thead>
<tr>
<th>ethnicity_concept_id</th>
<th>race ethnicity</th>
<th>When race = 'HISPANIC', or when ethnicity in ('CENTRAL_AMERICAN', 'DOMINICAN', 'MEXICAN', 'PUERTO_RICAN', 'SOUTH_AMERICAN') then set as 38003563, otherwise set as 0</th>
</tr>
</thead>
</table>

ETL Implementation

How should the PERSON table logic be implemented in SQL?

```sql
truncate cdm_lauren.person;
insert into cdm_lauren.person (person_id, ... ethnicity_source_concept_id )
select row_number() over(order by p.id) as person_id,
case upper(p.gender)
    when 'M' then 8507
    when 'F' then 8532
end as race_concept_id,
    race
    when race = 'WHITE' then set as 8527,
    when race = 'BLACK' then set as 8516,
    when race = 'ASIAN' then set as 8515,
    otherwise set as 0
    end as race_concept_id,
    case
    when upper(p.race) = 'HISPANIC'
    then 38003563 else 0
end as ethnicity_concept_id,
...
Let’s review the logic we decided on for how the PERSON table should be created.

<table>
<thead>
<tr>
<th>Gender:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>gender_concept_id</td>
<td>gender</td>
</tr>
<tr>
<td></td>
<td>When gender = ‘M’ then set gender_concept_id to 8507, when gender = ‘F’ then set to 8532</td>
</tr>
<tr>
<td></td>
<td>Drop any rows with missing/unknown gender</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Birthdate:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>year_of_birth</td>
<td>birthdate</td>
</tr>
<tr>
<td></td>
<td>Take year from birthdate</td>
</tr>
<tr>
<td>month_of_birth</td>
<td>birthdate</td>
</tr>
<tr>
<td></td>
<td>Take month from birthdate</td>
</tr>
<tr>
<td>day_of_birth</td>
<td>birthdate</td>
</tr>
<tr>
<td></td>
<td>Take day from birthdate</td>
</tr>
<tr>
<td>birth_datetime</td>
<td>birthdate</td>
</tr>
<tr>
<td></td>
<td>With midnight as time 00:00:00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Race:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>race_concept_id</td>
<td>race</td>
</tr>
<tr>
<td></td>
<td>When race = ‘WHITE’ then set as 8527, when race = ‘BLACK’ then set as 8516, when race = ‘ASIAN’ then set as 8515, otherwise set as 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ethnicity:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ethnicity_concept_id</td>
<td>race ethnicity</td>
</tr>
<tr>
<td></td>
<td>When race = ‘HISPANIC’, or when ethnicity in (‘CENTRAL_AMERICAN’, ‘DOMINICAN’, ‘MEXICAN’, ‘PUERTO_RICAN’, ‘SOUTH_AMERICAN’) then set as 38003563, otherwise set as 0</td>
</tr>
</tbody>
</table>
ETL Implementation

How should the PERSON table logic be implemented in SQL?

```sql
truncate cdm_lauren.person;
insert into cdm_lauren.person (  
    person_id,
    ...
    ethnicity_source_concept_id
)

select
    row_number() over(order by p.id) as person_id,
    case upper(p.gender)  
        when 'M' then 8507  
        when 'F' then 8532
    end as gender_concept_id,
    date_part('year', p.birthdate) as year_of_birth,
    date_part('month', p.birthdate) as month_of_birth,
    date_part('day', p.birthdate) as day_of_birth

<table>
<thead>
<tr>
<th>ethnicity_concept_id</th>
<th>race ethnicity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>When race = ‘HISPANIC’, or when ethnicity in {‘CENTRAL_AMERICAN’, ‘DOMINICAN’, ‘MEXICAN’, ‘PUERTO_RICAN’, ‘SOUTH_AMERICAN’} then set as 38003563, otherwise set as 0</td>
</tr>
</tbody>
</table>

```
ETL Implementation

Now let us run the code and create the PERSON table in the cdm_lauren schema

1. Download the query from:
   
   https://github.com/OHDSI/Tutorial-ETL
   
   Materials ➔ Implementation ➔ Insert_Person_Lauren.sql

2. Open up pgAdmin4 using the icon on the task bar
ETL Implementation

3. Expand the server list and right-click on PostgreSQL 10 and choose Connect Server from the drop-down menu.

4. When it asks for a password, type in ohdsi.
ETL Implementation

• Open up to and select the CDM (which has a copy of the vocab)

• Tools → Query Tool

• Type the following and hit F5 to run:

  \texttt{SET \hspace{1em} SEARCH\_PATH \hspace{1em} TO \hspace{1em} CDM\_LAUREN;\hspace{1em}}
ETL Implementation

7. Paste the sql code to create the PERSON table into the query window and press F5 or

NOTE:
The ‘truncate’ statement at the beginning deletes anything that is in the table already without deleting the table itself (helpful if you make a mistake)

QUESTIONS:
How would you check that your PERSON table was created?
How could you fix the ethnicity mapping?
ETL Implementation

Data Quality at implantation – ethnicity correction

Ethnicity

```sql
select row_number() over(order by p.id) as person_id,
    case upper(p.gender)
        when 'M' then 8507
        when 'F' then 8532
    end as gender_concept_id,
    date_part('year', p.birthdate) as year_of_birth,
    date_part('month', p.birthdate) as month_of_birth,
    p.birthdate,
    case when upper(p.race) = 'HISPANIC'
        then 38003563 else (case
            when upper(p.ethnicity) in ('CENTRAL_AMERICAN', 'DOMINICAN', 'MEXICAN', 'PUERTO_RICAN', 'SOUTH_AMERICAN')
            then 38003563 else 0 end
        )
end as race_concept_id;
```
Build the rest of the tables . . .

```r
### Synthea OMOP Builder code to run

library("ETLsyntheabuilder")
library("SqlRender")
library("DatabaseConnector")

# Create connectionDetails object to postgres (or other db)
connectionDetails <- DatabaseConnector::createConnectionDetails(
dbms="postgresql",
server="localhost/ETL",
user="postgres",
password="ohdsi",
port=5432
)

# Assuming the raw data and vocabulary has been loaded, this will run the synthea cdm sql builder
CreateEventTables(connectionDetails, "cdm_synthea_v2")

# Copy vocab tables into new schema

# CreateVocabMapTables(connectionDetails, "cdm_synthea_v2")

CreateVisitRollupTables(connectionDetails,
cdmDatabaseSchema = "cdm_synthea_v2",
syntheaDatabaseSchema = "raw_synthea"
)

LoadEventTables(connectionDetails,
cdmDatabaseSchema = "cdm_synthea_v2",
syntheaDatabaseSchema = "raw_synthea",
vocabDatabaseSchema = "cdm_synthea_v2"
)
```

github.com/ohdsi/ETL-synthea
Resources

• The full Synthea builder can be found here: https://github.com/OHDSI/ETL-Synthea

• Another example of a R/SQL builder for a much larger database:
  https://github.com/OHDSI/ETL-HealthVerityBuilder

• A builder created using .NET:
  https://github.com/OHDSI/ETL-CDMBuilder

• A builder created using the AWS lambda functionality:
  https://github.com/OHDSI/ETL-lambdabuilder
  (in development)
Example Builder 1: Janssen CDM Builder Over Time

Simple
- Simple SQL Queries
- Simple SQL Queries + Cursors
- SAS Builder

Sophisticated
- C# Single Machine
- C# Multiple Machine
- C# in Cloud Enabled Environment

https://github.com/OHDSI/ETL-CDMBuilder
Example Builder 2: PEDSnet

PEDSnet (n=6.2 million patients)  
8 contributing sites  

Data Coordinating Center:  
Children’s Hospital of Philadelphia (CHOP)

January 2009 to September 2014
• Children’s Hospital of Philadelphia
  – Data Coordinating Center (quarterly submissions)
    • PEDSnet DDL
    • ETL Conventions
    • Data Quality
    • Data Science
  – Also, a submitting site:
    • ~ 1.2 million patients
    • ~ 55 million visits
    • ~ 700 million clinical facts
CHOP ETL Flow – More like LTE

- **Load**: (very little re-organization of data)
- **Transform**: (Mapping of concepts, remapping ETL)
- **Extract**: to final PEDSnet (OMOP-like) tables

**Diagram**:
- Epic Clarity
- Staging Postgres DB
- Staging Tables
- Final Tables
Challenges/Lessons Learned

• We ultimately have to make decisions about our data:

  – What do we include?
    • Cancelled visits with associated information, reflects known workflow for research visits

  – What data do we exclude?
    • Cancelled Labs, Procedures
    • Test patients
    • Patients with lab only data (Adults lab/blood work, genetics)

  – Who makes these decisions?
    • Data Committee/Data Modeling Working Group
    • Local Informaticist and Analyst team
Challenges/Lessons Learned

• Our ETL is time-constraint due to clinical system ETL
  – Structured program to take into account midnight system wide shutdown for ETL

• Clinical data does not always fit OMOP rules
  – Multivitamin prescriptions with 2055 `end_date`
  – Fetal Procedures `procedure_start_date` before `birth_date`
  – Autopsies procedures `procedure_start_date` after `death_date`
  – Multiple “encounters” associated with one visit

• Intermediate/Temporary tables are crucial for debugging
  – Tables containing source identification numbers (IDS such as MRNS, patient ids, source system ids) alongside OMOP data before “final version”
Data Validation: Data Model Validator

- Validates table structures and data types
- Prompts user to specify the model and version number
- Alerts if there are any unexpected columns and/or tables
- [https://github.com/infomodels/infomodels](https://github.com/infomodels/infomodels) (OMOP model supported)
Data Validation:  
Data Quality Framework

- Automated Program where issues are flagged as GitHub issues categorized by table, domain and priority (High, Medium, Low)

- Checks fall into the following categories:
  - **Fidelity/Reliability**: Is this data correct? Is it being coded/mapped correctly?
  - **Consistency/Internal Validity**: Are there any drops/inconsistencies between submissions?
  - **Accuracy**: Does the data correctly reflect clinical characteristics of patients?
  - **Completeness**: Is there data that is missing?

- [https://pedsnet.org/data/data-quality/](https://pedsnet.org/data/data-quality/)
Data experts and CDM experts together design the ETL

People with medical knowledge create the code mappings

All are involved in quality control

A technical person implements the ETL
Quality

What tools are available to check that the CDM logic was implemented correctly?

- Rabbit-in-a-Hat Test Case Framework
- Achilles
- DataQualityDashboard (DQD)
Unit Test Cases

• Testing your CDM builder is important:
  – ETL often complex, increasing the danger of making mistakes that go unnoticed
  – CDM can update
  – Source data structure/contents can change over time

• Rabbit-In-a-Hat can construct unit test, or small pieces of code that can automatically check single aspects of the ETL design
Rabbit-in-a-Hat

The application has a feature called ‘Generate ETL Test Framework’. This feature allows you to create ‘fake’ people as a way to test your ETL logic.
Unit Test Cases

The test framework creates a series of R functions that enables you to specify your ‘fake’ people and records in the same structure as your source data using the scan report as a guide.

```r
source("Framework.R")
declareTest(101, "Person gender mappings")
add_enrollment(member_id = "M000000102", gender_of_member = "male")
add_enrollment(member_id = "M000000103", gender_of_member = "female")
expect_person(PERSON_ID = 102, GENDER_CONCEPT_ID = 8507)
expect_person(PERSON_ID = 103, GENDER_CONCEPT_ID = 8532)
```
Unit Test Cases

- White Rabbit scan report
- CDM specifications
- Test framework (R)
- Unit tests
- SQL to generate test data
- Test source DB
- Extract Transform Load
- Test CDM DB
- Test results

<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>Person gender mappings</td>
<td>PASS</td>
</tr>
<tr>
<td>101</td>
<td>Person gender mappings</td>
<td>PASS</td>
</tr>
</tbody>
</table>
Unit Test Cases

• An example of how this was done for the Synthea data is available from: https://github.com/OHDSI/Tutorial-ETL/tree/master/materials/Unit%20Tests

• The file that creates the test cases as a series of insert statement is RunSyntheaTestCases.r
Let us revisit the PERSON table logic:

| gender_concept_id | gender                      | When gender = ‘M’ then set gender_concept_id to 8507, when gender = ‘F’ then set to 8532 | Drop any rows with missing/unknown gender. |

How could we create a test case for this?

```r
createPersonTests <- function () {
  patient <- createPatient()
  declareTest(id = patient$id, description = "Drop patients with no gender, id is PERSON_SOURCE_VALUE")
  add_patients(id = patient$id, gender = NULL)
  expect_no_person(person_source_value = patient$id)
}
```

```sql
-- 1: Drop patients with no gender, id is PERSON_SOURCE_VALUE
INSERT INTO synthia_test.[patients](id, birthdate, ssn, prefix, first, last, marital, race, ethnicity, birthplace, address, city, state, zip) VALUES ('1', '1926-02-23', '999-41-5589', 'Mr.', 'Benito209', 'Marks830', 'M', 'white', 'irish', 'Boston', '192 MacGyver Dam', 'Boston', 'Massachusetts', '02108');
```
Achilles

Achilles is a data characterization and quality tool available for download here:

https://github.com/OHDSI/Achilles

For an example of how it was run for our sample data, that R script is located here:

This plot shows that the bulk of the data starts in 2005. However, there also appear to be a few records from around 1961, which is likely an error in the data.
This change coincides with changes in the reimbursement rules in this specific country, leading to more diagnoses but probably not a true increase in prevalence in the underlying population.
Achilles heel is a report generated by the Achilles application that will run a series of data quality checks on the CDM using the Achilles data.

<table>
<thead>
<tr>
<th>Message Type</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERROR</td>
<td>410-Number of condition occurrence records outside valid observation period; count (n=134) should not be &gt;</td>
</tr>
<tr>
<td>ERROR</td>
<td>610-Number of procedure occurrence records outside valid observation period; count (n=11) should not be &gt;</td>
</tr>
<tr>
<td>ERROR</td>
<td>710-Number of drug exposure records outside valid observation period; count (n=241) should not be &gt; 0</td>
</tr>
<tr>
<td>ERROR</td>
<td>712-Number of drug exposure records with invalid provider_id; count (n=29,518) should not be &gt; 0</td>
</tr>
<tr>
<td>ERROR</td>
<td>810-Number of observation records outside valid observation period; count (n=134) should not be &gt; 0</td>
</tr>
<tr>
<td>ERROR</td>
<td>812-Number of observation records with invalid provider_id; count (n=8,518) should not be &gt; 0</td>
</tr>
<tr>
<td>ERROR</td>
<td>909-Number of drug eras outside valid observation period; count (n=55) should not be &gt; 0</td>
</tr>
<tr>
<td>ERROR</td>
<td>1,009-Number of condition eras outside valid observation period; count (n=134) should not be &gt; 0</td>
</tr>
<tr>
<td>NOTIFICATION</td>
<td>[General] Not all decimals represented at first observation</td>
</tr>
<tr>
<td>NOTIFICATION</td>
<td>Unmapped data over percentage threshold in Measurement</td>
</tr>
<tr>
<td>NOTIFICATION</td>
<td>Unmapped data over percentage threshold in DrugExposure</td>
</tr>
<tr>
<td>NOTIFICATION</td>
<td>Unmapped data over percentage threshold in Observation</td>
</tr>
<tr>
<td>NOTIFICATION</td>
<td>99+ percent of persons have exactly one observation period</td>
</tr>
<tr>
<td>NOTIFICATION</td>
<td>percentage of non-numerical measurement records exceeds general population threshold</td>
</tr>
<tr>
<td>NOTIFICATION</td>
<td>Unmapped data over percentage threshold in Condition</td>
</tr>
</tbody>
</table>
DataQualityDashboard (DQD)

- Runs a prespecified set of data quality checks and thresholds on the CDM

### DATA QUALITY ASSESSMENT

**SYNTHA SYNTHETIC HEALTH DATABASE**

Results generated at 2019-08-22 14:15:06 in 29 mins

<table>
<thead>
<tr>
<th></th>
<th>Verification</th>
<th>Validation</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pass</td>
<td>Fail</td>
<td>Total</td>
</tr>
<tr>
<td>Plausibility</td>
<td>159</td>
<td>21</td>
<td>180</td>
</tr>
<tr>
<td>Conformance</td>
<td>637</td>
<td>34</td>
<td>671</td>
</tr>
<tr>
<td>Completeness</td>
<td>369</td>
<td>17</td>
<td>386</td>
</tr>
<tr>
<td>Total</td>
<td>1165</td>
<td>72</td>
<td>1237</td>
</tr>
</tbody>
</table>

|                | Pass         | Fail       | Total  | % Pass |
|----------------|--------------|------------|--------|
|                | 442          | 21         | 463    | 95%    |
|                | 741          | 34         | 775    | 96%    |
|                | 374          | 27         | 401    | 93%    |
|                | 1557         | 82         | 1639   | 95%    |
## DQD Example Rules

<table>
<thead>
<tr>
<th>Fraction violated rows</th>
<th>Check description</th>
<th>Threshold</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.34</td>
<td>A yes or no value indicating if the provider_id in the VISIT_OCCURRENCE is the expected data type based on the specification.</td>
<td>0.05</td>
<td>FAIL</td>
</tr>
<tr>
<td>0.99</td>
<td>The number and percent of distinct source values in the measurement_source_value field of the MEASUREMENT table mapped to 0.</td>
<td>0.30</td>
<td>FAIL</td>
</tr>
<tr>
<td>0.09</td>
<td>The number and percent of records that have a value in the drug_concept_id field in the DRUG_ERA table that do not conform to the ingredient class.</td>
<td>0.10</td>
<td>PASS</td>
</tr>
<tr>
<td>0.02</td>
<td>The number and percent of records with a value in the verbatim_end_date field of the DRUG_EXPOSURE that occurs prior to the date in the DRUG_EXPOSURE_START_DATE field of the DRUG_EXPOSURE table.</td>
<td>0.05</td>
<td>PASS</td>
</tr>
<tr>
<td>0.00</td>
<td>The number and percent of records that have a duplicate value in the procedure_occurrence_id field of the PROCEDURE_OCCURRENCE.</td>
<td>0.00</td>
<td>PASS</td>
</tr>
</tbody>
</table>
Issues in our synthetic data?

• Did our test cases run?

```sql
select * from cdm_synthea.test.test_results
```
Issues in our synthetic data?

• Did Achilles notice anything?
**Issues in our synthetic data?**

- Did DQD notice anything?

---

### SYNTHESA

Results generated at 2019-09-10 01:19:09 in 4 mins

<table>
<thead>
<tr>
<th>STATUS</th>
<th>CONTEXT</th>
<th>CATEGORY</th>
<th>SUBCATEGORY</th>
<th>LEVEL</th>
<th>DESCRIPTION</th>
<th>% RECORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>FAIL</td>
<td>Verification</td>
<td>Completeness</td>
<td>None</td>
<td>FIELD</td>
<td>The number and percent of records with a value of 0 in the standard concept field condition_concept_id in the CONDITION_OCCURRENCE table. (Threshold=5%).</td>
<td>60.86%</td>
</tr>
</tbody>
</table>

### METADATA

**RESULTS**

**ABOUT**

- PASS Validation Conformance Relational FIELD: The number and percent of records with a NULL value in the condition_concept_id of the CONDITION_OCCURRENCE that is considered not nullable. (Threshold=0%). 0%
- PASS Validation Conformance Relational FIELD: The number and percent of records with a NULL value in the condition_concept_id of the CONDITION_ERA that is considered not nullable. (Threshold=0%). 0%
- PASS Verification Conformance Value FIELD: A yes or no value indicating if the condition_concept_id in the CONDITION_OCCURRENCE is the expected data type based on the specification. (Threshold=0%). 0%
Maybe we have a bug?

- In the CONDITION_OCCURRENCE, 61% rows are mapped to 0
ETL Maintenance

- Changed or Updated Raw Data?
- Bug Found?
- New Vocab?
- CDM Update?

ETL Documentation

ETL

All are involved in quality control

A technical person implements the ETL

Updated CDM
Document the Bug

**Conditions not getting mapped to 0 #36**

**Expected behavior**

The majority of the source codes are mapped to concepts.

**Actual behavior**

About 63% of the codes are mapped to 0. It looks like some values are coming across as descriptions rather than ICD10 codes. We need to figure out how to get these mapped.

```
SELECT '0 RECORDS' AS TYPE, COUNT(*) ROW_COUNT
FROM CONDITION_OCCURRENCE
WHERE CONDITION_CONCEPT_ID = 0
UNION ALL
SELECT 'ALL RECORDS' AS TYPE, COUNT(*) ROW_COUNT
FROM CONDITION_OCCURRENCE
```

0 RECORDS = 4942
ALL RECORDS = 8120

<table>
<thead>
<tr>
<th>condition_occurrence_id</th>
<th>person_id</th>
<th>condition_concept_id</th>
<th>condition_source_value</th>
<th>character varying (250)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>28060</td>
<td>J02.0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>260139</td>
<td>J20</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>0</td>
<td>Stroke</td>
<td></td>
</tr>
</tbody>
</table>
Vocabulary to fix the problem

```sql
select * from cdn_synthea_v2.source_to_concept_map
```
Vocabulary to fix the problem

WITH CTE_VOCAB_MAP AS (  
  SELECT c.concept_code AS SOURCE_CODE, c.concept_id AS SOURCE_CONCEPT_ID, c.concept_name AS SOURCE_CODE_DESCRIPTION,  
  c.vocabulary_id AS SOURCE_VOCABULARY_ID, c.domain_id AS SOURCE_DOMAIN_ID, c.CONCEPT_CLASS_ID AS SOURCE_CONCEPT_CLASS_ID,  
  c.VALID_START_DATE AS SOURCE_VALID_START_DATE, c.VALID_END_DATE AS SOURCE_VALID_END_DATE, c.INVALID_REASON AS SOURCE_INVALID_REASON, c1.concept_id AS TARGET_CONCEPT_ID, c1.concept_name AS TARGET_CONCEPT_NAME, c1.VOCABULARY_ID AS TARGET_VOCABULARY_ID,  
  c1.domain_id AS TARGET_DOMAIN_ID, c1.concept_class_id AS TARGET_CONCEPT_CLASS_ID,  
  c1.INVALID_REASON AS TARGET_INVALID_REASON, c1.standard_concept AS TARGET_STANDARD_CONCEPT  
  FROM CONCEPT C  
  JOIN CONCEPT_RELATIONSHIP CR ON C.CONCEPT_ID = CR.CONCEPT_ID  
  AND CR.INVALID_REASON IS NULL  
  AND cr.relationship_id = 'Maps to'  
  JOIN CONCEPT C1 ON CR.CONCEPT_ID_2 = C1.CONCEPT_ID  
  AND C1.INVALID_REASON IS NULL  
  UNION  
  SELECT source_code, SOURCE_CONCEPT_ID, SOURCE_CODE_DESCRIPTION, source_vocabulary_id, c1.domain_id AS SOURCE_DOMAIN_ID,  
  c2.CONCEPT_CLASS_ID AS SOURCE_CONCEPT_CLASS_ID, c1.VALID_START_DATE AS SOURCE_VALID_START_DATE,  
  c1.VALID_END_DATE AS SOURCE_VALID_END_DATE, stcm.INVALID_REASON AS SOURCE_INVALID_REASON,target_concept_id,  
  c2.CONCEPT_NAME AS TARGET_CONCEPT_NAME, target_vocabulary_id, c2.domain_id AS TARGET_DOMAIN_ID,  
  c2.concept_class_id AS TARGET_CONCEPT_CLASS_ID, c2.INVALID_REASON AS TARGET_INVALID_REASON,  
  c2.standard_concept AS TARGET_STANDARD_CONCEPT  
  FROM source_to_concept_map stcm  
  LEFT OUTER JOIN CONCEPT c1  
  ON c1.concept_id = stcm.source_concept_id  
  LEFT OUTER JOIN CONCEPT c2  
  ON c2.CONCEPT_ID = stcm.target_concept_id  
  WHERE stcm.INVALID_REASON IS NULL  
)  
SELECT TARGET_CONCEPT_ID, TARGET_CONCEPT_NAME, TARGET_DOMAIN_ID  
FROM CTE_VOCAB_MAP  
WHERE SOURCE_VOCABULARY_ID = 'Synthea_conditions'
Update the ETL document

- [https://ohdsi.github.io/Tutorial-ETL/docs/cdm_synthea_v2](https://ohdsi.github.io/Tutorial-ETL/docs/cdm_synthea_v2)

<table>
<thead>
<tr>
<th>Destination Field</th>
<th>Source Field</th>
<th>Logic</th>
</tr>
</thead>
<tbody>
<tr>
<td>condition_concept_id</td>
<td>code</td>
<td>Use code to lookup target_concept_id in SOURCE_TO_STANDARD_VOCAB_MAP: select v.target_concept_id from conditions c join source_to_standard_vocab_map v on v.source_code = c.code and v.target_domain_id = 'Condition' and v.target_standard_concept = 'S' and v.source_vocabulary_id in ('ICD10CM', 'Synthea_conditions')</td>
</tr>
</tbody>
</table>
Re-run the DQD

### SYNTHEA

Results generated at 2019-09-10 12:57:12 in 5 mins

<table>
<thead>
<tr>
<th>STATUS</th>
<th>CONTEXT</th>
<th>CATEGORY</th>
<th>SUBCATEGORY</th>
<th>LEVEL</th>
<th>DESCRIPTION</th>
<th>% RECORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>FAIL</td>
<td>Verification</td>
<td>Completeness</td>
<td>None</td>
<td>FIELD</td>
<td>The number and percent of records with a value of 0 in the standard concept field condition_concept_id in the CONDITION_OCCURRENCE table. (Threshold=5%).</td>
<td>14.03%</td>
</tr>
<tr>
<td>FAIL</td>
<td>Verification</td>
<td>Completeness</td>
<td>None</td>
<td>FIELD</td>
<td>The number and percent of records with a value of 0 in the standard concept field condition_concept_id in the CONDITION_ERA table. (Threshold=0%).</td>
<td>14.02%</td>
</tr>
<tr>
<td>PASS</td>
<td>Validation</td>
<td>Conformance</td>
<td>Relational</td>
<td>FIELD</td>
<td>The number and percent of records with a NULL value in the condition_concept_id of the CONDITION_OCCURRENCE that is considered not nullable. (Threshold=0%).</td>
<td>0%</td>
</tr>
<tr>
<td>PASS</td>
<td>Validation</td>
<td>Conformance</td>
<td>Relational</td>
<td>FIELD</td>
<td>The number and percent of records with a NULL value in the condition_concept_id of the CONDITION_ERA that is considered not nullable. (Threshold=0%).</td>
<td>0%</td>
</tr>
<tr>
<td>PASS</td>
<td>Verification</td>
<td>Conformance</td>
<td>Value</td>
<td>FIELD</td>
<td>A yes or no value indicating if the condition_concept_id in the CONDITION_OCCURRENCE is the expected data type based on the specification. (Threshold=0%).</td>
<td>0%</td>
</tr>
</tbody>
</table>

Showing 1 to 5 of 14 entries (filtered from 3,351 total entries)
Re-run Achilles
Final Hard Lessons Learned
80/20 Rule

You don’t need to map all terms to get good data coverage!
Comfort with Data Loss

• If there is data that is not of research quality or there are methods to adjust, use the ETL to standardize that

<table>
<thead>
<tr>
<th>Reason to Drop Someone</th>
<th>Person Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unknown gender</td>
<td>23,592</td>
</tr>
<tr>
<td>Implausible year of birth - past</td>
<td>749</td>
</tr>
<tr>
<td>Implausible year of birth - post earliest observation period</td>
<td>3,836</td>
</tr>
<tr>
<td>Gender changes</td>
<td>2</td>
</tr>
</tbody>
</table>
ETL Process

Data experts and CDM experts together design the ETL

People with medical knowledge create the code mappings

All are involved in quality control

A technical person implements the ETL

ETL Documentation

OHDSI Tools

White Rabbit
Rabbit In a Hat
Usagi
White Rabbit
ACHILLES
DQD
Rabbit In a Hat
ETL Maintenance

- Changed or Updated Raw Data?
- Bug Found?
- New Vocab?
- CDM Update?

All are involved in quality control
A technical person implements the ETL
Updated CDM

ETL DocumentaJon
Updated CDM
Thank you!

This tutorial would not have been possible without the contribution of many collaborators in the OHDSI Community.

We like to thank Amazon Web Services for their valuable technical support and resources.
Acknowledgements

Anthony Molinaro who wrote the Synthea CDM Builder

James Wiggins who helps us prepare an AWS instance for use today

Pusheen the Cat

http://pusheen.com/
2019 OHDSI SYMPOSIUM

TUTORIALS:
SEPTEMBER
15TH & 17TH

Bethesda North Marriott
& Conference Center
5701 Marinelli Rd
North Bethesda, MD 20852

WWW.OHDSI.ORG/EVENTS/2019-OHDSI-SYMPOSIUM