

# OMOP (Observational Medical Outcomes Partnership) Common Data Model to i2b2 (Informatics for Integrating Bench to Bedside) Transformation Process

Last Revised: 17-November-2011

The purpose of this document is to describe the transformation process between the OMOP (Observational Medical Outcomes Partnership) common data model and the i2b2 ([Informatics for Integrating Bench to Bedside](#)) schema. The transformation scripts can be broken into two distinct processes: the loading process to move data from OMOP to the base schema in the i2b2 Data Repository (CRC) cell and the process of building the hierarchy tables used in the i2b2 Ontology Cell. For more information about i2b2, the i2b2 CRC cell, and the i2b2 Ontology cell, please read through the documentation at [www.i2b2.org](http://www.i2b2.org).

The transformation scripts assume several key tasks have already been completed before attempting to run the process. Specifically, an i2b2 instance must already be setup and installed and the underlying CRC database must reside on the same server as the OMOP CDM. Additional assumptions are laid out below.

Assumptions:

1. i2b2 Data Repository (CRC) database must already be created on the same server as the OMOP CDM.
2. The scripts were written for the Microsoft Sql Server 2005 platform and make use of T-SQL specific functionality. The scripts will need to be modified in order to support other platforms.
3. The user running the scripts must have full access to the i2b2 schema and the OMOP CDM.

Although the data loading transformation scripts are self contained and can be run in any order, we generally recommend starting with the patient dimension and gradually building up the i2b2 database one OMOP entity type at a time. Because of the amount of data being inserted into the i2b2 tables, performance may be better if indexes on Observation\_Fact and Visit\_Dimension are dropped or disabled before starting the transformation process. If this is done, integrity checking should be performed regularly to make sure data is not violating i2b2 primary keys.

The steps for converting OMOP CDM to i2b2 Data Repository Schema are listed below:

1. Create i2b2 Data Repository Schema
2. Load Patient Dimension Table
3. Load Patient Mapping Table
4. Load Condition Era Table
5. Load Condition Occurrence Table
6. Load Drug Era Table
7. Load Drug Exposure Table
8. Load Procedure Occurrent Table
9. Load Visit Occurrence Table
10. Load DOI Table
11. Load HOI Table
12. Load Observation Table

## 1. TRANSFORM DATA FROM OMOP COMMON DATA MODEL TO I2B2 DATA REPOSITORY SCHEMA

The process to transform the OMOP CDM to the i2b2 Data Repository is complex in that consolidates several OMOP tables (Drug Era, Drug Exposure, Condition Era, Condition Occurrence, Procedure Occurrence, Observation) into two i2b2 tables: Observation\_Fact and Visit\_Dimension. Because of this, new i2b2 specific encounter num identifiers had to be created to prevent violation of the primary keys on both Observation\_Fact and Visit\_Dimension. As a result, after the transformation, the Encounter\_Mapping table must be used to translate an i2b2 encounter num back to the corresponding OMOP CDM identifier.

Additionally, entity specific types (condition occurrence type, drug exposure type, etc.) had to find a place within the i2b2 specific schema where they related properly to the corresponding entry in the OMOP CDM. For i2b2 version 1.5, these relationship types became text value enumerations. For i2b2 version 1.6, these types became modifiers of the underlying concept type, and in some cases, required the insertion of additional rows for each modifier value.

**a. Create the i2b2 Data Repository Schema**

The installation guide for creating the i2b2 Data Repository cell can be found at [www.i2b2.org](http://www.i2b2.org). The installation process creates several tables that are required for querying the data using the i2b2 workbench tool; however, the data transformation scripts only make use of the following tables:

- Observation\_Fact
- Patient\_Dimension
- Visit\_Dimension
- Patient\_Mapping
- Encounter\_Mapping

**b. Load Patient Dimension Table (LoadPatientDimension.sql)**

The first step in the transformation process is loading the OMOP Person table into the i2b2 Patient\_Dimension table. Since both person\_id and patient\_num are defined as unique integer values, the OMOP person id can be directly mapped to the corresponding i2b2 patient num entry without requiring the creation of a new i2b2 identifier. This makes lookups across both systems simpler if the patient identifier is known. Mapping of the Person fields to Patient\_Dimension are outlined below:

OMOP		I2b2	
Person.Person_Id	=>	Patient_Dimension.Patient_Num	
Person.Year_of_Birth	=>	Patient_Dimension.Birth_Date	
Person.Gender_Concept_Id	=>	Patient_Dimension.Sex_Cd	
Person.Race_Concept_Id	=>	Patient_Dimension.Race_Cd	
Person.Location_Concept_Id	=>	Patient_Dimension.Zip_Cd	

**c. Load Patient Mapping Table (LoadPatientMapping.sql)**

Next, it is important to provide a link in the i2b2 warehouse that maps each patient to their original source person key. This is accomplished by loading the Patient\_Mapping table with the appropriate mappings from patient\_num to source.

OMOP		I2b2	
Person.Person_Id	=>	Patient_Mapping.Patient_Num	
Person.Source_Person_Key	=>	Patient_Mapping.Patient_Ide	
“OMOP_SOURCE”	=>	Patient_Mapping.Patient_Ide_Source	
“A”	=>	Patient_Mapping.Patient_Ide_Status	

All the remaining transforms follow the same format. First, the OMOP primary key for each table is mapped to a new i2b2 encounter sequence number and this entry is added to the Encounter\_Mapping table. Second, a visit dimension record is created that contains information about when the event occurred. Last, the observation fact table is loaded with the concepts from each table.

**d. Load Condition Era Table (LoadConditionEra.sql)**

Below are the key mappings for loading the Condition\_Era table into the i2b2 Encounter\_Mapping table:

OMOP		I2b2	
Condition_Era.Condition_Era_Id	=>	Encounter_Mapping.Encounter_Ide	
Condition_Era.Person_Id	=>	Encounter_Mapping.Patient_Num	
“OMOP CONDITION_ERA”	=>	Encounter_Mapping.Encounter_Ide_Source	
“A”	=>	Encounter_Mapping.Encounter_Ide_Status	
New Unique Encounter Num	=>	Encounter_Mapping.Encounter_Num	

Key mappings for loading the Condition\_Era table into the i2b2 Visit\_Dimension table:

OMOP		I2b2	
Mapped Encounter Num	=>	Visit_Dimension.Encounter_Num	
Condition_Era.Person_Id	=>	Visit_Dimension.Patient_Num	
Condition_Era.Condition_Start_Date	=>	Visit_Dimension.Start_Date	
Condition_Era.Condition_End_Date	=>	Visit_Dimension.End_Date	
“OMOP”	=>	Visit_Dimension.SourceSystem_Cd	

Key mappings for loading the Condition\_Era table into the i2b2 Observation\_Fact table:

OMOP		I2b2	
Mapped Encounter Num	=>	Observation_Fact.Encounter_Num	
Condition_Era.Person_Id	=>	Observation_Fact.Patient_Num	
Condition_Era.Condition_Concept_Id	=>	Observation_Fact.Concept_Cd	
Condition_Era.Condition_Start_Date	=>	Observation_Fact.Start_Date	
Condition_Era.Condition_End_Date	=>	Observation_Fact.End_Date	
“T”	=>	Observation_Fact.ValType_Cd	
Condition_Era.Persistence_Window	=>	Observation_Fact.TVal	
Condition_Era.Condition_Occurrence_Count	=>	Observation_Fact.Quantity_Num	
“OMOP”	=>	Observation_Fact.SourceSystem_Cd	

**e. Load Condition Occurrence Table (LoadConditionOcc.sql)**

Below are the key mappings for loading the Condition\_Occurrence table into the i2b2 Encounter\_Mapping table:

OMOP		I2b2	
Condition_Occurrence.Condition_Occurrence_Id	=>	Encounter_Mapping.Encounter_Ide	
Condition_Occurrence.Person_Id	=>	Encounter_Mapping.Patient_Num	
“OMOP CONDITION_OCCURRENCE”	=>	Encounter_Mapping.Encounter_Ide_Source	
“A”	=>	Encounter_Mapping.Encounter_Ide_Status	
New Unique Encounter Num	=>	Encounter_Mapping.Encounter_Num	

Key mappings for loading the Condition\_Occurrence table into the i2b2 Visit\_Dimension table:

OMOP		I2b2	
Mapped Encounter Num	=>	Visit_Dimension.Encounter_Num	
Condition_Occurrence.Person_Id	=>	Visit_Dimension.Patient_Num	
Condition_Occurrence.Condition_Start_Date	=>	Visit_Dimension.Start_Date	
Condition_Occurrence.Condition_End_Date	=>	Visit_Dimension.End_Date	
“OMOP”	=>	Visit_Dimension.SourceSystem_Cd	

Key mappings for loading the Condition\_Occurrence table into the i2b2 Observation\_Fact table:

OMOP		I2b2	
Mapped Encounter Num	=>	Observation_Fact.Encounter_Num	
Condition_Occurrence.Person_Id	=>	Observation_Fact.Patient_Num	
Condition_Occurrence.Condition_Concept_Id	=>	Observation_Fact.Concept_Cd	
Condition_Occurrence.Condition_Start_Date	=>	Observation_Fact.Start_Date	
Condition_Occurrence.Condition_End_Date	=>	Observation_Fact.End_Date	
“T”	=>	Observation_Fact.ValType_Cd	
Condition_Occurrence_Ref. Condition_Occurrence_Type_Desc	=>	Observation_Fact.TVal	
Condition_Occurrence.Stop_Reason, Condition_Occurrence.Dx_Qualifier, Condition_Occurrence.Source_Condition_Code	=>	Observation_Fact.Observation_Blob	
“OMOP”	=>	Observation_Fact.SourceSystem_Cd	

**f. Load Drug Era Table (LoadDrugEra.sql)**

Below are the key mappings for loading the Drug\_Era table into the i2b2 Encounter\_Mapping table:

OMOP		I2b2	
Drug_Era.Drug_Era_Id	=>	Encounter_Mapping.Encounter_Ide	
Drug_Era.Person_Id	=>	Encounter_Mapping.Patient_Num	
“OMOP DRUG_ERA”	=>	Encounter_Mapping.Encounter_Ide_Source	
“A”	=>	Encounter_Mapping.Encounter_Ide_Status	
New Unique Encounter Num	=>	Encounter_Mapping.Encounter_Num	

Key mappings for loading the Drug\_Era table into the i2b2 Visit\_Dimension table:

OMOP		I2b2	
Mapped Encounter Num	=>	Visit_Dimension.Encounter_Num	
Drug_Era.Person_Id	=>	Visit_Dimension.Patient_Num	
Drug_Era.Drug_Era_Start_Date	=>	Visit_Dimension.Start_Date	
Drug_Era.Drug_Era_End_Date	=>	Visit_Dimension.End_Date	
“OMOP”	=>	Visit_Dimension.SourceSystem_Cd	

Key mappings for loading the Drug\_Era table into the i2b2 Observation\_Fact table:

OMOP		I2b2	
Mapped Encounter Num	=>	Observation_Fact.Encounter_Num	
Drug_Era.Person_Id	=>	Observation_Fact.Patient_Num	
Drug_Era.Drug_Concept_Id	=>	Observation_Fact.Concept_Cd	
Drug_Era.Drug_Era_Start_Date	=>	Observation_Fact.Start_Date	
Drug_Era.Drug_Era_End_Date	=>	Observation_Fact.End_Date	

“T”	=>	Observation_Fact.ValType_Cd
Drug_Exposure_Ref. Drug_Exposure_Type_Desc	=>	Observation_Fact.TVal
Drug_Era.Drug_Exposure_Count	=>	Observation_Fact.Quantity_Num
“OMOP”	=>	Observation_Fact.SourceSystem_Cd

**g. Load Drug Exposure Table (LoadDrugExposure.sql)**

Below are the key mappings for loading the Drug\_Exposure table into the i2b2 Encounter\_Mapping table:

OMOP		I2b2	
Drug_Exposure.Drug_Exposure_Id	=>	Encounter_Mapping.Encounter_Ide	
Drug_Exposure.Person_Id	=>	Encounter_Mapping.Patient_Num	
“OMOP DRUG_EXPOSURE”	=>	Encounter_Mapping.Encounter_Ide_Source	
“A”	=>	Encounter_Mapping.Encounter_Ide_Status	
New Unique Encounter Num	=>	Encounter_Mapping.Encounter_Num	

Key mappings for loading the Drug\_Exposure table into the i2b2 Visit\_Dimension table:

OMOP		I2b2	
Mapped Encounter Num	=>	Visit_Dimension.Encounter_Num	
Drug_Exposure.Person_Id	=>	Visit_Dimension.Patient_Num	
Drug_Exposure.Drug_Exposure_Start_Date	=>	Visit_Dimension.Start_Date	
Drug_Exposure.Drug_Exposure_End_Date	=>	Visit_Dimension.End_Date	
“OMOP”	=>	Visit_Dimension.SourceSystem_Cd	

Key mappings for loading the Drug\_Exposure table into the i2b2 Observation\_Fact table:

OMOP		I2b2	
Mapped Encounter Num	=>	Observation_Fact.Encounter_Num	
Drug_Exposure.Person_Id	=>	Observation_Fact.Patient_Num	
Drug_Exposure.Drug_Concept_Id	=>	Observation_Fact.Concept_Cd	
Drug_Exposure.Drug_Exposure_Start_Date	=>	Observation_Fact.Start_Date	
Drug_Exposure.Drug_Exposure_End_Date	=>	Observation_Fact.End_Date	
“T”	=>	Observation_Fact.ValType_Cd	
Drug_Exposure_Ref. Drug_Exposure_Type_Desc	=>	Observation_Fact.TVal	
Drug_Exposure.Stop_Reason, Drug_Exposure.Refills, Drug_Exposure.Source_Drug_Code, Drug_Exposure.Drug_Quantity, Drug_Exposure.Days_Supply	=>	Observation_Fact.Observation_Blob	
Drug_Exposure.Drug_Quantity	=>	Observation_Fact.Quantity_Num	
“OMOP”	=>	Observation_Fact.SourceSystem_Cd	

**h. Load Procedure Occurrence Table (LoadProcedureOcc.sql)**

Below are the key mappings for loading the Procedure\_Occurrence table into the i2b2 Encounter\_Mapping table:

OMOP	I2b2
------	------

Procedure_Occurrence.Procedure_Occurrence_Id	=>	Encounter_Mapping.Encounter_Ide
Procedure_Occurrence.Person_Id	=>	Encounter_Mapping.Patient_Num
“OMOP PROCEDURE_OCCURRENCE”	=>	Encounter_Mapping.Encounter_Ide_Source
“A”	=>	Encounter_Mapping.Encounter_Ide_Status
New Unique Encounter Num	=>	Encounter_Mapping.Encounter_Num

Key mappings for loading the Procedure\_Occurrence table into the i2b2 Visit\_Dimension table:

OMOP		I2b2
Mapped Encounter Num	=>	Visit_Dimension.Encounter_Num
Procedure_Occurrence.Person_Id	=>	Visit_Dimension.Patient_Num
Procedure_Occurrence.Procedure_Date	=>	Visit_Dimension.Start_Date
Procedure_Occurrence.Procedure_Date	=>	Visit_Dimension.End_Date
“OMOP”	=>	Visit_Dimension.SourceSystem_Cd

Key mappings for loading the Procedure\_Occurrence table into the i2b2 Observation\_Fact table:

OMOP		I2b2
Mapped Encounter Num	=>	Observation_Fact.Encounter_Num
Procedure_Occurrence.Person_Id	=>	Observation_Fact.Patient_Num
Procedure_Occurrence.Procedure_Concept_Id	=>	Observation_Fact.Concept_Cd
Procedure_Occurrence.Procedure_Date	=>	Observation_Fact.Start_Date
Procedure_Occurrence.Procedure_Date	=>	Observation_Fact.End_Date
“T”	=>	Observation_Fact.ValType_Cd
Procedure_Occurrence_Ref. Procedure_Occurrence_Type_Desc	=>	Observation_Fact.TVal
Procedure_Occurrence.Source_Procedure_Code	=>	Observation_Fact.Observation_Blob
“OMOP”	=>	Observation_Fact.SourceSystem_Cd

**i. Load Visit Occurrence Table (LoadVisitOcc.sql)**

Below are the key mappings for loading the Visit\_Occurrence table into the i2b2 Encounter\_Mapping table:

OMOP		I2b2
Visit_Occurrence.Visit_Occurrence_Id	=>	Encounter_Mapping.Encounter_Ide
Visit_Occurrence.Person_Id	=>	Encounter_Mapping.Patient_Num
“OMOP VISIT_OCCURRENCE”	=>	Encounter_Mapping.Encounter_Ide_Source
“A”	=>	Encounter_Mapping.Encounter_Ide_Status
New Unique Encounter Num	=>	Encounter_Mapping.Encounter_Num

Key mappings for loading the Visit\_Occurrence table into the i2b2 Visit\_Dimension table:

OMOP		I2b2
Mapped Encounter Num	=>	Visit_Dimension.Encounter_Num
Visit_Occurrence.Person_Id	=>	Visit_Dimension.Patient_Num
Visit_Occurrence.Visit_Start_Date	=>	Visit_Dimension.Start_Date
Visit_Occurrence.Visit_End_Date	=>	Visit_Dimension.End_Date
“OMOP”	=>	Visit_Dimension.SourceSystem_Cd

Key mappings for loading the Visit\_Occurrence table into the i2b2 Observation\_Fact table:

OMOP		I2b2	
Mapped Encounter Num	=>	Observation_Fact.Encounter_Num	
Visit_Occurrence.Person_Id	=>	Observation_Fact.Patient_Num	
Visit_Occurrence.Visit_Concept_Id	=>	Observation_Face.Concept_Cd	
Visit_Occurrence.Visit_Start_Date	=>	Observation_Fact.Start_Date	
Visit_Occurrence.Visit_End_Date	=>	Observation_Fact.End_Date	
Visit_Occurrence.Source_Visit_Code	=>	Observation_Fact.Observation_Blob	
“OMOP”	=>	Observation_Fact.SourceSystem_Cd	

**j. Load DOI Table**

Below are the key mappings for loading the DOI\_Era table into the i2b2 Encounter\_Mapping table:

OMOP		I2b2	
DOI_Era.DOI_Occurrence_Id	=>	Encounter_Mapping.Encounter_Ide	
DOI_Era.Person_Id	=>	Encounter_Mapping.Patient_Num	
“OMOP DOI_ERA”	=>	Encounter_Mapping.Encounter_Ide_Source	
“A”	=>	Encounter_Mapping.Encounter_Ide_Status	
New Unique Encounter Num	=>	Encounter_Mapping.Encounter_Num	

Key mappings for loading the DOI\_Era table into the i2b2 Visit\_Dimension table:

OMOP		I2b2	
Mapped Encounter Num	=>	Visit_Dimension.Encounter_Num	
DOI_Era.Person_Id	=>	Visit_Dimension.Patient_Num	
DOI_Era.DOI_Start_Date	=>	Visit_Dimension.Start_Date	
DOI_Era.DOI_End_Date	=>	Visit_Dimension.End_Date	
“OMOP”	=>	Visit_Dimension.SourceSystem_Cd	

Key mappings for loading the DOI\_Era table into the i2b2 Observation\_Fact table:

OMOP		I2b2	
Mapped Encounter Num	=>	Observation_Fact.Encounter_Num	
DOI_Era.Person_Id	=>	Observation_Fact.Patient_Num	
DOI_Era.DOI_Concept_Id	=>	Observation_Face.Concept_Cd	
DOI_Era.DOI_Start_Date	=>	Observation_Fact.Start_Date	
DOI_Era.DOI_End_Date	=>	Observation_Fact.End_Date	
“T”	=>	Observation_Fact.ValType_Cd	
Drug_Exposure_Ref. Drug_Exposure_Type_Desc	=>	Observation_Fact.TVal	
DOI_Era.Drug_Exposure_Count	=>	Observation_Fact.Quantity_Num	
“OMOP”	=>	Observation_Fact.SourceSystem_Cd	

**k. Load HOI Table**

Below are the key mappings for loading the HOI\_Occurrence table into the i2b2 Encounter\_Mapping table:

OMOP		I2b2	
HOI_Occurrence.HOI_Occurrence_Id	=>	Encounter_Mapping.Encounter_Ide	
HOI_Occurrence.Person_Id	=>	Encounter_Mapping.Patient_Num	

“OMOP HOI_ERA”	=>	Encounter_Mapping.Encounter_Ide_Source
“A”	=>	Encounter_Mapping.Encounter_Ide_Status
New Unique Encounter Num	=>	Encounter_Mapping.Encounter_Num

Key mappings for loading the HOI\_Occurrence table into the i2b2 Visit\_Dimension table:

OMOP		I2b2	
Mapped Encounter Num	=>	Visit_Dimension.Encounter_Num	
HOI_Occurrence.Person_Id	=>	Visit_Dimension.Patient_Num	
HOI_Occurrence.HOI_Occurrence_Start_Date	=>	Visit_Dimension.Start_Date	
HOI_Occurrence.HOI_Occurrence_End_Date	=>	Visit_Dimension.End_Date	
“OMOP”	=>	Visit_Dimension.SourceSystem_Cd	

Key mappings for loading the HOI\_Occurrence table into the i2b2 Observation\_Fact table:

OMOP		I2b2	
Mapped Encounter Num	=>	Observation_Fact.Encounter_Num	
HOI_Occurrence.Person_Id	=>	Observation_Fact.Patient_Num	
HOI_Occurrence.HOI_Concept_Id	=>	Observation_Fact.Concept_Cd	
HOI_Occurrence.HOI_Occurrence_Start_Date	=>	Observation_Fact.Start_Date	
HOI_Occurrence.HOI_Occurrence_End_Date	=>	Observation_Fact.End_Date	
“OMOP”	=>	Observation_Fact.SourceSystem_Cd	

## 1. Load Observation Table

Below are the key mappings for loading the Observation table into the i2b2 Encounter\_Mapping table:

OMOP		I2b2	
Observation.Obs_Occurrence_Id	=>	Encounter_Mapping.Encounter_Ide	
Observation.Person_Id	=>	Encounter_Mapping.Patient_Num	
“OMOP OBSERVATION”	=>	Encounter_Mapping.Encounter_Ide_Source	
“A”	=>	Encounter_Mapping.Encounter_Ide_Status	
New Unique Encounter Num	=>	Encounter_Mapping.Encounter_Num	

Key mappings for loading the Procedure\_Occurrence table into the i2b2 Visit\_Dimension table:

OMOP		I2b2	
Mapped Encounter Num	=>	Visit_Dimension.Encounter_Num	
Observation.Person_Id	=>	Visit_Dimension.Patient_Num	
Observation.Obs_Date	=>	Visit_Dimension.Start_Date	
Observation.Obs_Date	=>	Visit_Dimension.End_Date	
“OMOP”	=>	Visit_Dimension.SourceSystem_Cd	

Key mappings for loading the Observation table into the i2b2 Observation\_Fact table:

OMOP		I2b2	
Mapped Encounter Num	=>	Observation_Fact.Encounter_Num	
Observation.Person_Id	=>	Observation_Fact.Patient_Num	
Observation.Obs_Concept_Id	=>	Observation_Fact.Concept_Cd	
Observation.Obs_Date	=>	Observation_Fact.Start_Date	
Observation.Obs_Date	=>	Observation_Fact.End_Date	

“T” or “N” depending on what columns contain values	=>	Observation_Fact.ValType_Cd
Observation.Value_As_Number	=>	Observation_Fact.NVal
Observation.Value_As_String, Observation.Value_As_Concept_Id	=>	Observation_Fact.TVal
Observation.Source_Obs_Code, Observation.Source_Obs_Units, Observation.Obs_Range_High, Observation.Obs_Range_Low	=>	Observation_Fact.Observation_Blob
Observation.Obs_Units_Concept_Id	=>	Observation_Fact.Units_Cd
“OMOP”	=>	Observation_Fact.SourceSystem_Cd

## 2. CREATE OMOP METADATA FOR THE I2B2 ONTOLOGY CELL

Although transforming the OMOP CDM to the i2b2 Data Repository requires, the data is unqueryable in the i2b2 framework without creating an ontology hierarchy to organization and navigate the vast amount of metadata. The OMOP CDM provides some relationships to help with this organization; however, for many vocabularies, it is up to the institution implementing these scripts to map the missing concepts into an organized structure.

The general philosophy around the creation of the OMOP metadata was to a) ensure OMOP identifiers were used at all times instead of coded vocabulary values and b) build as much of the hierarchy as possible from the OMOP relationship tables before importing vocabularies from other sources. In this way, institutions that didn't have access to the organizational structure of all vocabularies could still make use of the concepts that appear in the OMOP concept\_ancestor table.

The following diagram shows the OMOP tables that can be used to organize the data. For the i2b2 mapping scripts, only the relationships contained in concept\_ancestor, concept, and vocabulary\_ref were used to build the i2b2 metadata tables. Inter-vocabulary relationships found in the concept\_relationship table could be used to build a more robust ontology tree; however, this involves additional complexity so was not included in the initial release of the metadata build scripts.

The steps for creating the OMOP Metadata table for the i2b2 Ontology Cell are listed below:

1. Create OMOPMetadata Table
2. Populate OMOPMetadata From Concept Ancestor
3. Insert Institutional Vocabularies Not Included in OMOP (ICD-9, CPT-4, RxNorm, LOINC)
4. Insert Demographics
5. Insert DOI and HOI
6. Check Integrity of Tree
7. Insert Metadata XML
8. Create Indexes
9. Create and Index Trimmed Metadata Table (Optional)

**a. Create OMOPMetadata Table**

The first step is creating a new metadata table to hold the OMOP ontology. This table mirrors the standard ontology tables used in i2b2 with the exception of two new columns: `i_source_ontology` lists the vocabulary where this concept came from (i.e. LOINC, SNOMED, CPT-4, etc) and `i_source_code` gives the coded value associated with this concept for the given vocabulary.

OmopMetaData	
	<b>C_FULLNAME</b>
	<b>C_HLEVEL</b>
	<b>C_NAME</b>
	<b>C_SYNONYM_CD</b>
	<b>C_VISUALATTRIBUTES</b>
	C_TOTALNUM
	C_BASECODE
	C_METADATAXML
	<b>C_FACTTABLECOLUMN</b>
	<b>C_TABLENAME</b>
	<b>C_COLUMNNAME</b>
	<b>C_COLUMNDATATYPE</b>
	<b>C_OPERATOR</b>
	C_DIMCODE
	C_COMMENT
	C_TOOLTIP
	UPDATE_DATE
	DOWNLOAD_DATE
	IMPORT_DATE
	SOURCESYSTEM_CD
	VALUETYPE_CD
	i_source_ontology
	i_source_code

**b. Populate OMOPMetaData From Concept Ancestor**

Once the table has been built, it is now ready to load with data from the concept\_ancestor table. The script uses the following process to build the table:

- The topmost level of the OMOPMetaData tree is the node “OMOP”. All other concepts will be descendants of this topmost node.
- Level 1 nodes correspond to the primary OMOP CDM types:
  - Conditions
  - Demographics
  - Drugs
  - Drug Outcomes of Interest
  - Health Outcomes of Interest
  - Procedures
  - Observations
- Level 2 nodes are the unique concept classes for the highest level nodes for each type found in concept\_ancestor
- All subsequent nodes follow the parent child order defined in concept ancestor
  - First find all concepts that are a parent, but not a child
  - Next, walk down the concept ancestor tree inserting along the way until the parent child relationship list is exhausted

**c. Insert Institutional Vocabularies Not Included in OMOP (ICD-9, CPT-4, RxNorm, LOINC)**

The relationships found in `concept_ancestor` account for roughly 50% of all concepts in found in the OMOP concept table. In order to query the remaining concepts, each institution would need to manually add the missing concepts based on the institutional hierarchies they use. Some vocabulary based scripts have been included to outline the basic transformation process for including them in the tree (CPT-4, ICD-9, LOINC, RxNorm), however, the hierarchical structure associated with these vocabularies has been intentionally left out.

#### **d. Insert Demographics**

There is no default hierarchy included in OMOP for organizing the demographic concepts. Although there are only four types of demographic attributes, navigating this data can be difficult given the number of elements for each type (in particular, location codes). To help with this, a simple organization structure is created to make navigation easier.

The demographic tree is largely taken from the i2b2 demo metadata included with the i2b2 software with a few enhancements:

- The structure used for finding patient ages is taken directly from demo I2B2 table
- The Race/Ethnicity and Gender folders use the concept names directly from the OMOP concept table
- OMOP only includes the first 3 digits of the zip code for each location code. To give some order to these codes, the i2b2 zip code hierarchy was used by removing the city names folders from the hierarchy and remapping the 3 digit zips to the appropriate states they are found in.

#### **e. Insert DOI and HOI**

Two specialized tables are included in OMOP for organizing the DOI and HOI concepts: `DOI_ancestor` and `HOI_ancestor`. Both tables only go down one level; each DOI is broken down into the individual drug concepts that were used to create its definition while only the first HOI definition for each type can be expanded to see the individual condition concepts that were included for it. A general “Drug Outcomes of Interest” and “Health Outcomes of Interest” folder is included on the first level of the hierarchy to contain both types.

#### **f. Check Integrity of Tree**

Once the data for each vocabulary is added to `OMOPMetaData` table, it is necessary to recheck the tree for missing nodes and invalid folder paths. Not all branches and leaves are created cleanly when importing from various sources. For instance, some paths from `concept_ancestor` are incomplete, so there are missing folders. Additionally, some concepts imported from other sources may have folders instead of leaves.

The integrity script traverses the entire ontology tree and tries to correct or fix problems along the way. It does this by:

1. First, set the visual attributes of all entries in the table to folder
2. Next, for each level of the hierarchy (there are 21), the `c_fullname` column is used to if the existing node has children somewhere else in the table. If no children exist for the entry, then the visual attributes of the concept are set to a leaf, otherwise they remain a folder.

#### **g. Insert MetaData XML**

Within the OMOP CDM, each entity type is modified by additional types or definitions. These types are described below:

- Drug Era – Persistence window
- Drug Exposure – Exposure Type
- Condition Occurrence – Occurrence Type (First condition, second condition, etc.)
- Condition Era – Persistence window
- Observation – Observation Type
- Procedure – Procedure Type

In order to include this information within i2b2, each type is added to the underlying data as an enumerated value that can be set when querying that particular type. For instance, drug eras can have a persistence window of 0 or 30 days.

For this to work within i2b2, the metadata table must be modified with the list of enumerated values for each type. This script adds the appropriate xml values for each type based on the various reference tables included in the OMOP CDM.

#### **h. Create Indexes**

Indexes are added to the table for faster performance while navigating the tree in the Ontology cell.

#### **i. Create and Index Trimmed MetaData Table (Optional)**

Because of the enormous size of the data included in the metadata table (~8 million entries), users may experience indexing problems or poor performance when using the metadata in the i2b2 workbench. To overcome this, a script is included to decrease the bytes of data that gets stored in the table by shortening the path names for each concept. This is achieved by using the following process:

For each folder in the metadata table:

1. Order the children of that folder by name
2. Assign each child an integer value sequence number
3. Convert that integer value to base 32 string (a-z, 0-9)
4. Substitute the folder's readable name in the `c_fullname` column for newly calculated base 32 string

As an example, the entry for SNOMED entry for Raynaud's Syndrome gets converted from :

```
\OMOP\Conditions\System Organ Class\Vascular disorders\Arteriosclerosis, stenosis, vascular  
insufficiency and necrosis\Peripheral vasoconstriction, necrosis and vascular insufficiency\Raynaud's  
phenomenon\Raynaud's syndrome\
```

To:

```
\1\1\2\1\3\4\1\
```

The length of the entry goes from 224 to 17. When performed on all of the entries in the table, the overall physical size of the table decreases significantly.