

Use of Event Combined Table to Simplify ETL into OMOP CDM

PRESENTER: **Don Torok**

INTRO:

ETL into OMOP CDM is hard. Using a combined event table makes it easier. ETL code writes to the same location regardless of the domain of the 'Maps to' target concept. Afterward a simple set of views or queries deconstructs the combined event table into Conditions, Procedures, Drug Exposures, etc.

Getting Started

The *event combined* table is created by identifying all the common and unique attributes of the OMOP event tables. All event tables share several common attributes such as primary key, the person id, visit and visit detail links, start date time, mapped to concept, source value, etc. Then there are attributes unique to the specific tables such as number of refills or days supply from Drug Exposure.

Two additional attributes, *default_domain_id* and *domain_id*, are added and will be used to determine the target table when extracting from the *event combined* table into the specific OMOP event tables. The *default_domain_id* is determined by the source data. This represents the OMOP table the ETL would write to if it did not have to apply the OHDSI mapping rules. The *domain_id* is filled with the domain id of the mapped to target concept.

I found the easiest way to identify the common and unique attributes of the OMOP event tables is to annotate each table and its attributes on a single Excel worksheet. This work was done using OHDSI CDM version 5.3

The ETL

The transform code that reads from the source data and writes into the *event combined* table is deterministic. The ETL code reads from a source attribute and writes to the same attribute in *event combined* regardless of the vocabulary mapping. The ETL code tries to map the source concept, but this is only to populate the *domain* attribute in the *event combined* table, no additional logic as a result of the mapped-to domain is required.

To create the views that represent the OMOP CDM tables work backwards using the Excel workbook definition of the *event combined* table. For each view, the 'where clause' determines if a row belongs to the target table by the *domain_id*, if it is defined, otherwise the *default_domain_id*

RESULTS

Using an intermediate table that is a composite of all the columns of the event tables simplifies the ETL process. It allows the ETL code to disregard the OHDSI mapping rules and eliminates the need to write redundant procedures for each possible target table.

OMOP ETL Made Easier

	procedure_occurrence	Drug Exposure	condition_occurrence	measurement	observation	Common	data type2
1	procedure_occurrence_id	drug_exposure_id	condition_occurrence_id	measurement_id	observation_id	event_primary_id	INTEGER
2	person_id	person_id	person_id	person_id	person_id	person_id	INTEGER
3	procedure_concept_id	drug_concept_id	condition_concept_id	measurement_concept_id	observation_concept_id	event_concept_id	INTEGER
4	procedure_date	drug_exposure_start_date	condition_start_date	measurement_date	observation_date	event_start_date	DATE
5		drug_exposure_end_date	condition_end_date			event_end_date	DATE
6	procedure_datetime	drug_exposure_start_datetime	condition_start_datetime	measurement_datetime	observation_datetime	event_start_datetime	DATETIME
7		drug_exposure_end_datetime	condition_end_datetime			event_end_datetime	DATETIME
8	procedure_source_value	drug_source_value	condition_source_value	measurement_source_value	observation_source_value	event_source_value	VARCHAR(50)
9	procedure_source_concept_id	drug_source_concept_id	condition_source_concept_id	measurement_source_concept_id	observation_source_concept_id	event_source_concept_id	INTEGER
10	procedure_source_vocabulary_id	drug_source_vocabulary_id	condition_source_vocabulary_id	measurement_source_vocabulary_id	observation_source_vocabulary_id	event_source_vocabulary_id	varchar(20)
11	procedure_type_concept_id	drug_type_concept_id	condition_type_concept_id	measurement_type_concept_id	observation_type_concept_id	event_type_concept_id	INTEGER
12	visit_occurrence_id	visit_occurrence_id	visit_occurrence_id	visit_occurrence_id	visit_occurrence_id	visit_occurrence_id	INTEGER
13	visit_detail_id	visit_detail_id	visit_detail_id	visit_detail_id	visit_detail_id	visit_detail_id	INTEGER
14	provider_id	provider_id	provider_id	provider_id	provider_id	provider_id	INTEGER
15	quantity	quantity				quantity	FLOAT
16		stop_reason	stop_reason			stop_reason	VARCHAR(20)
17	modifier_concept_id				qualifier_concept_id	qualifier_concept_id	INTEGER
18					unique_device_id	unique_device_id	VARCHAR(50)
19	modifier_source_value				qualifier_source_value	qualifier_source_value	VARCHAR(50)
20		refills			refills	refills	INTEGER
21		days_supply			days_supply	days_supply	INTEGER
22		sig			sig	sig	CLOB
23		effective_drug_dose			effective_drug_dose	effective_drug_dose	FLOAT
24		dose_unit_concept_id			dose_unit_concept_id	dose_unit_concept_id	INTEGER
25		lot_number			lot_number	lot_number	VARCHAR(50)
26		route_concept_id			route_concept_id	route_concept_id	INTEGER
27		route_source_value			route_source_value	route_source_value	VARCHAR(50)
28		dose_unit_source_value			dose_unit_source_value	dose_unit_source_value	VARCHAR(50)
29				value_as_number	value_as_number	value_as_number	FLOAT
30				value_as_concept_id	value_as_concept_id	value_as_concept_id	INTEGER
31				operator_concept_id	operator_concept_id	operator_concept_id	INTEGER
32				unit_concept_id	unit_concept_id	unit_concept_id	INTEGER
33				unit_source_value	unit_source_value	unit_source_value	VARCHAR(50)
34				value_source_value	value_as_string	value_as_string	VARCHAR(50)
35				range_low	range_low	range_low	FLOAT
36				range_high	range_high	range_high	FLOAT
37					domain	domain	VARCHAR(20)
38					default domain	default domain	varchar(20)
39			condition_status_concept_id		event_status_concept_id	event_status_concept_id	INTEGER
40			condition_status_source_value		event_status_source_value	event_status_source_value	VARCHAR(50)
41			condition_origin		condition_origin	condition_origin	varchar(10)

Advantages

- Code is deterministic
- No need to repeat code based on target domain
- Easier to compare source to Event Combined table.

How Domain and Default Domain Works

The default domain is which table the row would go into if there was no mapping. The domain is where the source code is mapped to. COALESCE(domain, default_domain) is where the data will end up in the CDM

Decompose

```
CREATE view Procedure
(
  ,procedure_occurrence_id
  ,person_id
  ,procedure_concept_id
  ,procedure_date
  ,procedure_datetime
  ,procedure_source_value
  ,procedure_source_concept_id
  ,procedure_source_vocabulary_id
  ,procedure_type_concept_id
  ,visit_occurrence_id
  ,visit_detail_id
  ,provider_id
  ,quantity
  ,modifier_concept_id
  ,modifier_source_value
)
AS
SELECT
  event_primary_id
  ,person_id
  ,event_concept_id
  ,event_start_date
  ,event_start_datetime
  ,event_source_value
  ,event_source_concept_id
  ,event_source_vocabulary_id
  ,event_type_concept_id
  ,visit_occurrence_id
  ,visit_detail_id
  ,provider_id
  ,quantity
  ,qualifier_concept_id
  ,qualifier_source_value
FROM event_combined;
```



Take a picture to download the full paper



Don Torok,
Mark Khayter,
Misha Bronshvayg

