# Standardizing Electronic Health Records to Improve Medical Research and Analysis

Alyssa S. Huque, Intern, Aaron Z. Berger, Intern, William R. Hersh, M.D., Steven D. Bedrick, Ph.D., Aaron M. Cohen, M.D., Steven R. Chamberlin, N.D.

Department of Medical Informatics and Clinical Epidemiology School of Medicine, Oregon Health and Science University

## Background

The passage of the HITECH Act incentivized medical institutions to digitize health records. However, with this transition, there were difficulties standardizing data in electronic health records (EHRs). As such, today there are many inconsistencies in the information captured by various institutions[1]. Observational Health Data Sciences and Informatics (OHDSI), an interdisciplinary group, has extended the Observational Medical Outcomes Partnership (OMOP) Common Data Model (CDM) to improve the quality of EHRs across instutitions[1]. The OMOP CDM standardizes the information captured by electronic health records allowing for analytics and research across multiple, disparate data sources[2]. Standardizing the data in EHRs can accelerate research, improve collaborative efforts and support clinical practice and administrative claims[2].

## Objective

To develop and implement a software program that maps OHSU's EHR data stored in XML files to new XML files that are formatted and organized by the standards set by the OMOP CDM.

## Methods

1. Parse the OHSU XML source files
2. Store the EHR patient data into a list
3. Add all lists to a queue
4. Remove one list at a time from the queue
5. Index the list and assign the information to the OMOP field
6. Print a new XML file that follows the OMOP CDM standards

## Future Directions

- Refine OHSU to OMOP mappings to ensure minimal loss of data.
- Increase the efficiency of software to handle a few million data records.
- Create relational linkages of multiple XML files for a given patient.
- Implement standardized codes used in the OMOP CDM _concept_id fields

## Acknowledgements and References

*This project was supported by the OHSU Department of Medical Informatics and Clinical Epidemiology and National Library of Medicine Grant: 2T15LM007088, Research Training in Biomedical Informatics and Data Science at OHSU.*

*[1]"OMOP Common Data Model." OHDSI, www.ohdsi.org/data-standardization/the-common-data-model/. Accessed 26 Jun. 2019.*

*[2]Voss EA, Makadia R, Matcho A, Ma Q, Knoll C, Schuemie M, DeFalco FJ, Londhe A, Zhu V, Ryan PB. Feasibility and utility of applications of the common data model to multiple, disparate observational health databases. J Am Med Inform Assoc. 2015 May;22(3):553-64. DOI: 10.1093/jamia/ocu023. Epub 2015 Feb 10. PubMed PMID: 25670757; PubMed Central PMCID: PMC4457111.*

## Steps 1 and 2

```python
def demographics(data_set):
    '''parses the xml file for the demographics tables of patients and
    stores only the information that maps to the OMOP CDM in a list'''
    base_path = os.path.dirname(os.path.realpath(__file__)) # path to this file
    xml_file = os.path.join(base_path, "C:\\Users\\huque\\Desktop\\Data\\{}".format(data_set))
    tree = et.parse(xml_file) # parses xml file
    root = tree.getroot()

    # creates a list for each DATA_RECORD
    collected_data = []
    for d in root.findall('DATA_RECORD'):
        collected_data.append([d.find(x).text for x in ['OHSU_PATIENT_ID',
            'BIRTH_DATE',
            'GENDER',
            'ETHNICITY',
            'RACE',
            'DEATH_DATE',
            'LAST_NAME',
            'FIRST_NAME',
            'CITY',
            'ADDRESS_LINE1',
            'ADDRESS_LINE2',
            'STATE',
            'ZIP',
            'COUNTY',
            'PRIMARY_CARE_PROVIDER']])

    return collected_data
```

## Steps 3 and 4

```python
def q_demographics(data_set):
    '''adds the list containing data records of demographics
    to a queue and continuously removes a list and prints an OMOP PERSON
    and LOCATION XML document until the queue is empty'''
    demographics_queue = Queue()
    for i in range(len(pohsu.demographics(data_set))):
        demographics_queue.enqueue(pohsu.demographics(data_set)[i]) # adds lists to queue
    while demographics_queue.isEmpty() != True:
        data_record = demographics_queue.dequeue() # removes list from queue
        pxml.print_demographics_PERSON_LOCATION(data_record) # prints OMOP xml document
```

## Step 5

```python
def demographics_PERSON_elements(root, collected_data):
    '''indexes the demographics list and maps the OHSU data fields to
    the OMOP structure for the PERSON table'''
    record = et.SubElement(root, 'DATA_RECORD')
    et.SubElement(record, "person_id").text = collected_data[0]
    et.SubElement(record, "gender_concept_id").text = None # map from Athena
    et.SubElement(record, "year_of_birth").text = collected_data[1][6:10]
    et.SubElement(record, "month_of_birth").text = collected_data[1][0:2]
    et.SubElement(record, "day_of_birth").text = collected_data[1][3:5]
    et.SubElement(record, "birth_datetime").text = collected_data[1]
    et.SubElement(record, "death_datetime").text = collected_data[5]
    et.SubElement(record, "race_concept_id").text = None # map from Athena
    et.SubElement(record, "ethnicity_concept_id").text = None # map from Athena
    et.SubElement(record, "location_id").text = None # unique ID from location table
    et.SubElement(record, "provider_id").text = None # from provider table
    et.SubElement(record, "care_site_id").text = None
    et.SubElement(record, "person_source_value").text = collected_data[7] + ' ' + collected_data[6]
    et.SubElement(record, "gender_source_value").text = collected_data[2]
    et.SubElement(record, "gender_source_concept_id").text = None
    et.SubElement(record, "race_source_value").text = collected_data[4]
    et.SubElement(record, "race_source_concept_id").text = None
    et.SubElement(record, "ethnicity_source_value").text = collected_data[3]
    et.SubElement(record, "ethnicity_source_concept_id").text = None


def demographics_LOCATION_elements(root, collected_data):
    '''indexes the demographics list and maps the OHSU data fields to
    the OMOP structure for the LOCATION table'''
    record = et.SubElement(root, 'DATA_RECORD')
    et.SubElement(record, "location_id").text = None # unique system generated ID
    et.SubElement(record, "address_1").text = collected_data[9]
    et.SubElement(record, "address_2").text = collected_data[10]
    et.SubElement(record, "city").text = collected_data[8]
    et.SubElement(record, "state").text = collected_data[11]
    et.SubElement(record, "zip").text = collected_data[12]
    et.SubElement(record, "county").text = collected_data[13]
    et.SubElement(record, "country").text = None # find from state
    et.SubElement(record, "location_source_value").text = None
    et.SubElement(record, "latitude").text = None
    et.SubElement(record, "longitude").text = None
```

## Step 6

```python
def print_demographics_PERSON_LOCATION(collected_data):
    '''takes a list of demographics from OHSU data records and
    reformats to a PERSON and LOCATION table that meets OMOP standards'''
    root = et.Element('LOCATION')
    xmlc.demographics_PERSON_elements(root, collected_data) # indexes list

    tree = et.ElementTree(root)
    tree.write(f"C:\\Users\\huque\\Desktop\\data\\PERSON_{collected_data[0]}.xml")

    root = et.Element('PERSON')
    xmlc.demographics_LOCATION_elements(root, collected_data) # indexes list

    tree = et.ElementTree(root)
    tree.write(f"C:\\Users\\huque\\Desktop\\data\\LOCATION_{collected_data[0]}.xml")
```