German Medical Data Sciences: Bringing Data to Life
R. Röhrig et al. (Eds.)

doi:10.3233/SHTI210077

Accessing OMOP Common Data Model Repositories with the i2b2 Webclient – Algorithm for Automatic Query Translation

Raphael W. MAJEED [a,c,1], Patrick FISCHER [b], and Andreas GÜNTHER [a]

[a] *Universities Giessen and Marburg Lung Center (UGMLC), German Centre For Lung Research (DZL), Justus-Liebig University Giessen, Germany*
[b] *Institute for Medical Informatics, Justus-Liebig University Giessen, Germany*
[c] *Institute of Medical Informatics, Medical Faculty of RWTH AachenUniversity, Germany*

Abstract. In the era of translational research, data integration and clinical data warehouses are important enabling technologies for clinical researchers. The OMOP common data model is a wide-spread choice as a target for data integration in medical informatics. It's portability of queries and analyses across different institutions and data are ideal also from the viewpoint of the FAIR principles. Yet, the OMOP CDM lacks a simple and intuitive user interface for untrained users to run simple queries for feasibility analysis. Aim of this study is to provide an algorithm to translate any given i2b2 query to an equivalent query which can then be run on the OMOP CDM database. The provided algorithm is able to convert queries created in the i2b2 webclient to SQL statements which can be executed on a standard OMOP CDM database programmatically.

Keywords. Data integration, standardization, factual databases, i2b2, OMOP

1. Introduction

1.1. Scientific Background

In the era of translational research, data integration and clinical data warehouses are important enabling technologies for clinical researchers. "Data sharing, integration and annotation are essential to ensure the reproducibility of the analysis and interpretation of the experimental findings." [1]

While there are several commercial data warehouse software solutions on the market, attractive software tools for non-profit university research in the bio-medical field are "Informatics for Integrating Biology & the Bedside" (i2b2)[2], the tranSMART data warehouse [3] and the "Observational Health Data Sciences and Informatics" (OHDSI) OMOP Common Data Model (CDM)[4].

I2b2 uses a star-schema database model with four dimensions/tables and includes a web-based query tool which can be easily used by medical professionals without IT-background to define patient cohorts, find patient counts and perform simple analyses like breakdowns by age, gender, diagnosis and compare cohorts (figure 1).

[1] Corresponding Author, R.W.Majeed, E-mail: Raphael.Majeed@chiru.med.uni-giessen.de

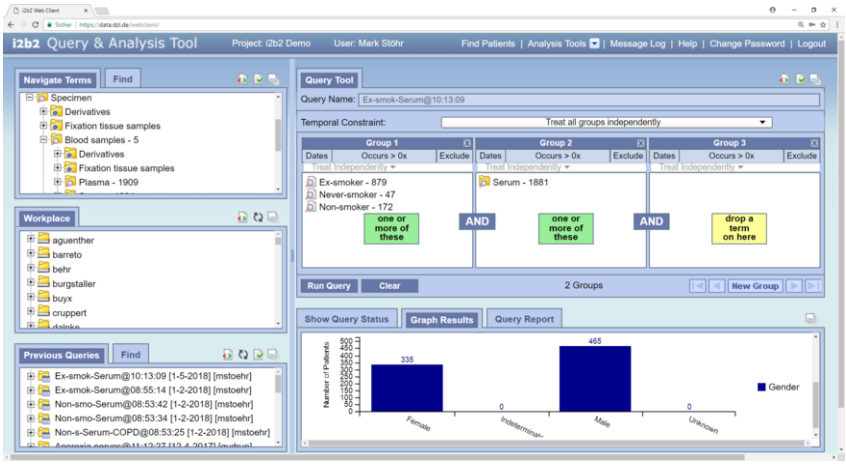

Figure 1. i2b2 web frontend.

tranSMART uses database schema similar to i2b2 but with a heavier focus on data from basic research and high-dimensional omics data [3]. Its web-client allows more advanced statistical analysis as compared to the i2b2 webclient [5]. This added functionality negatively impacts usability and makes it less intuitive to use for untrained users. Although advanced statistical functionality is provided, the frontend "does not check whether even basic preconditions for statistical tests are fulfilled" [5]. For research networks, the comprehensive statistical functionality can discourage users from sharing their data for fear of unasked data usage Recently, a second web-frontend, "GlowingBear" was added [6]. Compared to the classic tranSMART frontend, it provides a more modern design, but has very limited analysis functionality.

OHDSI OMOP CDM is a relational database schema with 16 detailed clinical data tables, 10 standardized vocabulary tables and additional tables for health system, economics, metadata, etc. It is being used extensively throughout the field of medical research from integration of longitudinal data [7] to analysis of treatment pathways [8] and nation-wide initiatives [9]. The OHDSI OMOP tool for defining patient cohorts and queries is the Atlas user interface. Its overall usability needs improvements [10] and in our experience is difficult to use for untrained users and unsuitable for basic and quick feasibility queries. Additional user query tools have been developed for OMOP CDM repositories, e.g. PatientExploreR for visualization of patient clinical history [10], ROMOP for interfacing health record data with R [11].

Basic interoperability on the database level between i2b2 and OMOP CDM has been shown by Klan et.al. [12], transferring data from i2b2 to OMOP via specialized ETL software, although the solution is limited to a specific data set and information model (PCORnet). The feasibility of connecting OMOP data to i2b2 has been shown by the same group by extending i2b2 to support multiple data tables [13].

A different approach on interoperability between different data warehouse systems can be achieved via query translation algorithms. Fette et.al. implemented translation of queries from i2b2 to openEHR AQL and vice versa [14] while Mate et.al. showed the feasibility to translate between i2b2 queries and CentraXX database queries [15].

1.2. Objective of the study

The OMOP CDM is a wide-spread choice as a target for data integration in medical informatics. It's portability of queries and analyses across different institutions and data are ideal also from the viewpoint of the FAIR principles. Yet, the OMOP CDM lacks a simple and intuitive user interface for untrained users to run simple queries for feasibility analysis.

Aim of this study is to provide an algorithm to translate any given i2b2 query to an equivalent query which can be run on the OMOP CDM database.

1.3. Essential functionality in the i2b2 webclient

The i2b2 webclient serves three essential functions which need to be implemented to be usable for querying foreign (non-i2b2) repositories.

First, *query management functionality* allows the user to view a list of previous queries, display information about each query (name, date, user and execution status) as well as the ability to execute, rename and delete queries.

Second *metadata browsing functionality* allows the user to navigate through a hierarchy of clinical terminologies and data elements. This metadata is used to discover data available in the data warehouse and as selection criteria for query construction. For the webclient, the relevant metadata for data elements is a display name, concept/terminology code and hierarchical path/id.

Third, *query definition functionality* allows using metadata elements to be combined and restricted to define patient cohorts. Restrictions can be applied to data elements e.g. by limiting numeric values of lab measurements or selecting one of several allowed values of a value set. Combination of data elements is done by dragging concepts in one of several panels. All elements in a panel are combined via logical OR operation and the panels themselves are combined via AND with one another. With both logical operators, the query definition panels make sure that an i2b2 query is always defined in conjunctive normal form. This normalization is important for the query translation algorithm. The XML representation of an i2b2 query is shown in figure 2.

```
<ns4:request xmlns:ns4="http://www.i2b2.org/xsd/cell/crc/psm/1.1/"
                                xsi:type="ns4:query_definition_requestType">
        <query_definition>
                <query_name>String -Etceter@22:19:05</query_name>
                <query_timing>ANY</query_timing>
                <panel>
                        <panel_number>1</panel_number>
                        <invert>0</invert>
                        <panel_timing>ANY</panel_timing>
                        <total_item_occurrences>1</total_item_occurrences>
                        <item>
                                <item_name>Female</item_name>
                                <item_key>\Demographics\Gender\F</item_key>
                        </item>
                        <item>
                                <item_name>Unknown</item_name>
                                <item_key>\Demographics\Gender\U</item_key>
                        </item>
                </panel>
                <panel>
                        <panel_number>2</panel_number>
                        <invert>0</invert>
                        <panel_timing>ANY</panel_timing>
                        <total_item_occurrences>1</total_item_occurrences>
                        <item>
                                <item_name>COVID-19</item_name>
                                <item_key>\Diag\COVID-19</item_key>
                        </item>
                </panel>
        </query_definition>

        <result_output_list>
                <result_output priority_index="9" name="patient_count_xml"/>
        </result_output_list>
</ns4:request>
```

Figure 2. XML representation of an i2b2 query, reduced to essential content.

1.4. Essential functionality of the OMOP CDM

OMOP CDM on the other side, stores data in a completely different information model and structure: clinical information is separated into domains *condition, measurement, specimen, observation, study, visit, person,* etc. Data in each topic is represented in a detailed relational table. Conceptual information is separated in a concept table linked to standard vocabularies.

1.5. Metadata model differences between i2b2 webclient and OMOP CDM

The OMOP CDM enforces the use of standard vocabularies and terminologies (e.g. LOINC, SNOMED-CT, ICD10, etc.). It is possible to use own terminologies, but doing so requires more effort. On the contrary, i2b2 does not need to be used with standard terminologies. There is only a single concept code string without specification of a coding system in i2b2. The most common way to use standard codes is by concatenating coding system and code with a colon ':' as separator. E.g. 'LOINC:8867-4' or ICD10CM:J10.

Metadata concept relations in i2b2 are purely hierarchical in the form of parent/child relationships. All metadata concepts need to be part of the hierarchy shown in the webclient. Querying patients using a metadata concept which contains children will always include patients associated with any child concepts in addition to patients with the actual concept. The OMOP CDM on the other hand allows a multitude of relations between concepts and also allows concepts without any relations.

2. Methods

For development of the query translation and execution algorithm, the current release of the i2b2 webclient version 1.7.12 is used as well as the current OMOP CDM release 5.2.2.

For input queries, we assume that the query has been created with the i2b2 webclient. The underlying metadata used by the webclient must be available to the algorithm, since the query definition does not include terminology codes (only hierarchical paths). In this metadata, all non-folder metadata items need to have standard concept codes assigned in combination with vocabulary info (e.g. 'LOINC:8867-4' in i2b2.base_code)

For simplicity, we choose not to cover value restrictions, logical inversions and timing variations in the algorithm. Instead, we focus on explaining the basic algorithm. As output of the algorithm, we calculate three different types of results: a patient list, patient count as well as a breakdown by gender.

3. Results

The query translation and execution algorithm requires additional tables in the OMOP CDM database. For distinction from the original CDM tables, these tables are prefixed with "iq_" (i2b2 query). The algorithm is arranged in four states. Algorithm steps are grouped by these states and described in the following sub-sections.

3.1. State A, Initialization and metadata

In the OMOP CDM database, the following additional tables are required for state A:

The table iq_query contains a unique *id* identifying each query, a *title* for display purposes, a *source* for syntactic representation of the original query definition as well as user and timestamp information. For *status* and *progress* reporting purposes, two more fields are introduced.

The table *iq_query_panel_item* contains information about each metadata element used in the query. A foreign key *query_id* references to the iq_query table. The item's panel number is stored in *panel_no* and its key in *item_key*. For execution, four more fields are introduced: A varchar i2b2_*concept_cd* for lookup of the i2b2 concept codes, two numeric columns *expansion_id* and *generated_by* for concept expansion (see state B) and a numeric omop_*concept_id* for mapping to corresponding OMOP CDM concepts.

The table *iq_result_type* is used to specify the result types which need to be calculated for a query. A foreign key *query_id* references to the iq_query table. The result type is stored in a varchar column *result_type*.

The algorithm starts in state A, takes as input a valid i2b2 query definition (see figure 2) and performs the following steps:

A.1: generate a new query id, e.g. from a database sequence

A.2: insert query metadata into the *iq_query* table and initialize status and progress fields.

A.3: insert panels and items into the *iq_query_panel_item* table (panel_no, item_key). The remaining fields are left empty and will be filled later

A.4: insert result types into the *iq_result_type* table.

3.2. State B: Item expansion and OMOP concept lookup

In the OMOP CDM database, the following additional database objects are required for state B:

From the metadata tree displayed in the i2b2 webclient which generated the query definition, the full item hierarchy with concept codes is needed. This can be accomplished by copying the underlying table used by the i2b2 webclient to produce the metadata view into the OMOP CDM schema (e.g. metadata.i2b2). Since the query definition only lists the hierarchical item_key, this table is used to lookup the corresponding concept codes as well as child relations.

A sequence *panel_item_expansion_seq* to generate numeric expansion ids. This sequence is used for the corresponding field in *iq_query_panel_item*.

The algorithm proceeds to state B and performs the following steps for the current query on the table *iq_query_panel_item*:

B.1: For rows with empty omop_concept_id and expansion_id (unmapped and unexpanded concepts), use the i2b2 metadata to lookup item_key and fill the i2b2_concept_cd.

B.2: For rows with empty omop_concept_id, empty expansion_id and filled i2b2_concept_cd, lookup and fill the corresponding OMOP concept id. For this step, the i2b2_concept_cd prefix (e.g. 'LOINC:' can be used to lookup the corresponding OMOP vocabulary and then find the exact code in the OMOP concept table. If the OMOP concept cannot be found for an i2b2_concept_cd, change to a failure state (non-standard i2b2 concept code or missing OMOP vocabulary) and abort the algorithm.

B.3: If there are rows with empty omop_concept_id and empty i2b2_concept_cd, repeat the following steps B.3.1 to B.3.3 for each row:

B.3.1: For the current row, assign a new expansion id (using sequence *panel_item_expansion_seq*).

B.3.2: Expand the item into the panel: Use its item_key to find all child entries in the i2b2 metadata and insert all children's item_key, i2b2_concept_cd into the panel. The generated_by field for these insertions is set to the current expansion id.

B.3.3: If an item could not be expanded because it did not produce child entries, change to a failure state (folder hierarchy without child concepts) and abort the algorithm.

B.4: Go to B.2 until omop_concept_id is filled for all entries.

B.5: All panel items have a valid omop_concept_id assigned. The relevant items for the query are those with empty *expansion_id* (all leaf nodes) The original panel items before expansion can be reconstructed by looking at rows with empty *generated_by*.

3.3. State C: Panel execution and combination (repeated for each panel)

In the OMOP CDM database, the following additional tables are required for state C:

The temporary table *iq_temp_concept_closure* is created for each query temporarily by the algorithm and used to derive the transitive closure (hull) of concepts for each OMOP concept. It contains numeric fields *omop_concept_id* and *derived_from* both referencing the original OMOP concept table and *domain_id* referencing OMOP domain.

Two temporary tables *iq_panel_result_a*, *iq_panel_result_b* for storing result patient ids. Both tables contain a single numeric column *result_id* referencing OMOP person ids.

The algorithm proceeds to state C and performs the following steps for the first panel of the current query.

C.1: Clear contents of table *iq_temp_concept_hull*.

C.2: From *iq_query_panel_item*, copy all *omop_concept_id* for current panel into *iq_temp_concept_closure*.

C.3: Build the closure of the OMOP concept descendant relationship on the set of panel concepts. This is done by inserting all descendant omop_concept_id obtained via the OMOP concept_ancestor table into *iq_temp_concept_closure*.

C.4: Lookup OMOP and fill *domain_id* for all concepts in *iq_temp_concept_closure* via the OMOP concept table.

C.5: For each supported domain (e.g. observation, condition, measurement, specimen, etc.) perform the following steps C.5.1-C.5.3:

C.5.1: Skip domain, if not used in *iq_temp_concept_closure*.

C.5.2: Construct query to insert patient_id into *qt_panel_result_a*. This query is specific to each domain. E.g. for measurement the OMOP table measurement is joined with *iq_temp_concept_closure* (limited to measurement domain) and the OMOP person table (this changes for panels other than the first). Resulting patient ids are added to *qt_panel_result_a*.

C.5.3: *qt_panel_result_a* now contains patient ids for all panel items ORed (first panel)

C.6: For each additional query panel, perform the following steps C.6.1-C.6.2:

C.6.1: Move content of *qt_panel_result_a* to *qt_panel_result_b*

C.6.2: Repeat steps C.1 to C.5.3 for next panel (2..n), but replace in C.5.2 the person table join with a join of *qt_panel_result_b*. This causes the resulting patient set of the next panel to be AND-combined with the previous panel results.

C.7: *qt_panel_result_a* now contains patient ids matching all query criteria from all panels. Drop table *qt_panel_result_b*, which is no longer needed.

3.4. State D: Result compilation

In the OMOP CDM database, the following additional tables are required for state D:

The table *iq_result_patient* is used to store the calculated patient set. It contains columns *query_id* referencing the *iq_query* table and numeric *patient_id*.

The table *iq_result_breakdown* is used to store calculated breakdowns. For this purpose, columns *query_id*, *breakdown_type*, *category* and numeric *value* are used.

The algorithm proceeds to state D and performs the following steps:

D.1: Copy all *patient_id* from *qt_panel_result_a* to the table *iq_result_patient* assigned to the current query id. This persists the resulting patient set/cohort into the database.

D.2: To store the resulting patient count, calculate the number of rows in *qt_panel_result_a* and store the result into a single row in *iq_result_breakdown* with breakdown_type of 'patient_count'.

D.3: To generate a categorical breakdown, join the table *qt_panel_result_a* with the desired table by *person_id* and count distinct values. E.g. for gender breakdown, join with the OMOP person table and count the distinct values of gender_concept_id. The resulting counts are stored in *iq_result_breakdown*.

4. Discussion

The provided algorithm is able to convert queries created in the i2b2 webclient to SQL statements which can then be executed on a standard OMOP CDM database. Due to the way how metadata/concept hierarchies are handled in i2b2, the algorithm requires metadata hierarchies to be added to the OMOP CDM schema. The most feasible way to create queries for this algorithm is by using the i2b2 webclient. It has been shown, that the webclient can be used for this purpose without a full i2b2 data warehouse installation [16].

The provided algorithm was tested by manually performing the described steps on a PostgreSQL 12.2 database using OMOP CDM version 5.2.2. To implement the algorithm, a programming language is needed. Depending on the database management system (DBMS) used for the OMOP CDM, the algorithm can be directly implemented in the embedded procedural programming language of the DBMS. For example, with Oracle and PostgreSQL, implementations can be written in PL/SQL or PL/pgSQL respectively. For Microsoft SQL Server, Transact-SQL or .NET Framework CLR can be used for stored procedures.

We chose to present the algorithm in its abstract form, because the OMOP CDM is built to run on several different DBMS and users rarely use all OMOP domains/relations in a single implementation. While the algorithm is tailored to executing i2b2 query definitions on OMOP-CDM data warehouses, can also be modified to work on other clinical databases or information models.

Translation of queries between i2b2 and other databases had been shown feasible by other groups [14-15]. The presented algorithm extends this interoperability on the query level to also include translation between i2b2 and OMOP CDM.

4.1. Limitations

For simplicity, the algorithm does not cover value restrictions, logical inversions and timing variations. These features can be added into the algorithm by implementers, depending on their need for this functionality. Implementations of the algorithm may add these features by adding columns and additional SQL code. Implementations without these features can choose to reject queries when users try to use unsupported functionality.

The algorithm works with all OMOP CDM tables which can be directly derived from OMOP concept domains. This includes the OMOP domains observation, condition, measurement, specimen, procedure, drug and visit. Other domains like e.g. unit, route, gender, race, provider, geography are not supported by our algorithm for patient selection. To allow use of these domains in queries, code specific to the desired additional domain can be added after step C.3.

5. Conclusion

The OMOP common data model is a wide-spread choice as a target for data integration in medical informatics, but lacks user friendly general purpose query tool. The provided algorithm can be implemented on top of OMOP CDM repositories and enables the use of the widely used i2b2 query tool web frontend for simple queries and analyses. Our abstract algorithm is independent of specific DBMS and can also be modified to work on other clinical databases or information models.

Conflict of Interest

The authors state that they have no conflict of interests.

References

[1] Lapatas V, Stefanidakis M, Jimenez RC, Via A, Schneider MV. Data integration in biological research: an overview. Journal of Biological Research-Thessaloniki. 2015 Dec;22(1):9.

[2] Murphy SN, Weber G, Mendis M, Gainer V, Chueh HC, Churchill S, Kohane I. Serving the enterprise and beyond with informatics for integrating biology and the bedside (i2b2). Journal of the American Medical Informatics Association. 2010 Mar 1;17(2):124-30. Please strictly adhere to the specifications of IoS Press.

[3] Athey BD, Braxenthaler M, Haas M, Guo Y. tranSMART: an open source and community-driven informatics and data sharing platform for clinical and translational research. AMIA Summits on Translational Science Proceedings. 2013;2013:6.

[4] FitzHenry F, Resnic FS, Robbins SL, Denton J, Nookala L, Meeker D, Ohno-Machado L, Matheny ME. Creating a common data model for comparative effectiveness with the observational medical outcomes partnership. Applied clinical informatics. 2015;6(03):536-47.

[5] Christoph J, Knell C, Naschberger E, Stürzl M, Maier C, Prokosch HU, Sedlmayr M. Two Years of tranSMART in a University Hospital for Translational Research and Education. IneHealth 2017 May 12 (pp. 70-79).

[6] Cirillo E. Glowing Bear officially launched. The Hyve. 2018. http://blog.thehyve.nl/blog/glowing-bear-officially-launched (last accessed 2020-07-18)

[7] Lima DM, Rodrigues-Jr JF, Traina AJ, Pires FA, Gutierrez MA. Transforming two decades of ePR data to OMOP CDM for clinical research. Stud Health Technol Inform. 2019 Aug 21;264:233-7.

[8] Zhang X, Wang L, Miao S, Xu H, Yin Y, Zhu Y, Dai Z, Shan T, Jing S, Wang J, Zhang X. Analysis of treatment pathways for three chronic diseases using OMOP CDM. Journal of medical systems. 2018 Dec 1;42(12):260.

[9] Maier C, Lang L, Storf H, Vormstein P, Bieber R, Bernarding J, Herrmann T, Haverkamp C, Horki P, Laufer J, Berger F. Towards implementation of OMOP in a German university hospital consortium. Applied clinical informatics. 2018 Jan;9(01):054-61.

[10] Glicksberg BS, Oskotsky B, Thangaraj PM, Giangreco N, Badgeley MA, Johnson KW, Datta D, Rudrapatna VA, Rappoport N, Shervey MM, Miotto R. PatientExploreR: an extensible application for dynamic visualization of patient clinical history from electronic health records in the OMOP common data model. Bioinformatics. 2019 Nov 1;35(21):4515-8.

[11] Glicksberg BS, Oskotsky B, Giangreco N, Thangaraj PM, Rudrapatna V, Datta D, Frazier R, Lee N, Larsen R, Tatonetti NP, Butte AJ. ROMOP: a light-weight R package for interfacing with OMOP-formatted electronic health record data. JAMIA open. 2019 Apr;2(1):10-4.

[12] Klann JG, Joss MA, Embree K, Murphy SN. Data model harmonization for the All Of Us Research Program: Transforming i2b2 data into the OMOP common data model. PloS one, 2019; 14(2), e0212463.

[13] Klann JG, Phillips LC, Herrick C, Joss MA, Wagholikar KB, Murphy SN. Web services for data warehouses: OMOP and PCORnet on i2b2. Journal of the American Medical Informatics Association, 25(10), 1331-1338.

[14] Fette G, Kaspar M, Liman L, et al. Query Translation Between AQL and CQL. Stud Health Technol Inform. 2019;264:128-132. doi:10.3233/SHTI190197

[15] Mate S, Vormstein P, Kadioglu D, et al. On-The-Fly Query Translation Between i2b2 and Samply in the German Biobank Node (GBN) Prototypes. Stud Health Technol Inform. 2017;243:42-46.

[16] Majeed RW, Xu T, Stöhr MR, Röhrig R. Li2b2-Façade: Simulation of i2b2 Data Warehouse Server and Client for Interaction with Other Systems. Studies in health technology and informatics. 2017;245:1275-.