

Summarizing FHIR® to OMOP Transformation Exceptions using Generative AI

Ron Sweeney, Hannah Kimura, Qi Li
Integration Required, InterSystems, InterSystems

Background

As organizations increasingly utilize standards-based data formats like FHIR® and seek to harness the power of analytics through OMOP, the need for robust, scalable transformation support has never been greater. However, the transformation from FHIR to OMOP is inherently complex—requiring deep domain knowledge, precision, regional understanding and the ability to handle a wide range of edge cases and evolving data standards.

Like most exploring Generative AI for opportunities to bolster efficiencies and elevate support throughput, we used it to extend knowledge of FHIR®, OHDSI, Terminology Mappings and Transformation logic to apply to the transformation runs for speed in iteration and support. For this we built a Retrieval-Augmented Generation system coupled with a TEXT to SQL implementation for specialized support designed to enhance the transformation pipeline accuracy and accelerate resolution of issues that arise during FHIR®-to-OMOP conversions.

Methods

The Open Source [Vanna.ai](#)¹ seemed to fit the bill for the use case and our environment and we could modify it and quickly get up and running with the examples provided. We implemented it as a Python Notebook coupled with a Git repository as a Runbook to Operations.

Vanna.ai is an open-source Python framework that enables natural language querying of SQL databases through Retrieval-Augmented Generation (RAG). You first “train” a Vanna model on your database schema, documentation, for example SQL queries, and then ask it questions in plain English, which it reliably translates into SQL for execution. Vanna was a fantastic fit for the FHIR® and OMOP Common Data Model by inspecting the FHIR® SQL Projected tables which includes the OperationOutcome⁵ and the OMOP CDM itself. [Vanna.ai](#) works by implementing a local vector database (ChromaDB) for similarity, connecting to any number of supported databases (OMOP/FHIR SQL), and an LLM/Model of choice (Chat GPT/Gemini).

Though out of the box it delivered results we could use, it fell short of anything outside of what it learned from the tables it inspected, so we implemented several RAG calls for increased focus on the components, data and logic of the FHIR to OMOP transformation.

We loaded into the model Chapters 4-7 from The Book of OHDSI⁴ to give the model a good understanding of OMOP core principals. We loaded FHIR documentation, specifically explanations of the FHIR® resource types utilized by OMOP, so the model understood FHIR® resources. We loaded FHIR-to-OMOP mappings, so the model understood which FHIR resource mapped to which OMOP table(s). And finally, we loaded specialized knowledge regarding edge cases that need to be understood for FHIR-to-OMOP transformations that we learned in practice. For detailed understanding of the RAG calls in python, inspect the notebook in the Github Repository.

Minimizing Hallucinations

A core customization to [Vanna.ai](#) was to include setting the temperature for the LLM dynamically (depending on the task), creating different prompts based on if the question was conceptual or one that required sql generation, and adding a check for hallucinations in the SQL generated by the LLM.

Temperature controls the randomness of text that is generated by LLMs during inference. A lower temperature essentially makes those tokens with the highest probability more likely to be selected; a higher temperature increases a model's likelihood of selecting less probable tokens. For tasks such as generating SQL, we want the temperature to be lower to prevent hallucinations. Hallucinations are when the LLM makes up something that doesn't exist. In SQL generation, a hallucination may look like the LLM querying a column that doesn't exist. This renders the query unusable, and throws an error. Thus, we edited the `generate_sql` function to change the temperature dynamically. The temperature is between 0 - 1. For questions deemed to be conceptual, we set the temperature to be 0.6, and for questions that require generating sql, we set the temperature to be 0.2. Furthermore, for tasks such as generating sql, the temperature is 0.2, while for tasks such as generating graphs and summaries, the default temperature is 0.5. We decided on 0.5 for the graph and summary tasks, because they require more creativity.

Another customization that helped substantially with hallucinations was to ensure we trained the model exactly once per use. At first, we repeatedly trained on top of the data, and ended up with really terrible results without any predictability to the focus of the generated responses . For this we ensured that we recreated the Vector Database required by [Vanna.ai](#), the training data, and counted on the incremental SQL table information to add subsequent timely value to the model. We managed to get the Notebook to run in under 10 minutes until it was ready to be prompted.

Results

To illustrate what we ended up with, we took an example service ticket description and worked the problem as usual with the implementer to conclusion, then turned around and prompted our implementation of [Vanna.ai](#) with the problem and came remarkably close to our previous conclusion with two subsequent prompts. What problems were encountered (Figure 1) a summary of those problems (Figure 2) , and what a possible remediation plan would look like to alleviate those problems (Figure 3) .

You

We created a test fhir data set that was used for a prior pharma project and sent through our omop pipeline. There were several conversion issues and conversion warnings reported in the UI. we need to review the warnings and issues to determine a starting point to fixing the problems, please provide a detailed summary of the warnings and issues and information on how to fix them and supply a list of the affected patient ids



WARNING_ID	ISSUE_ID	TARGET_FIELD	WARNING_MESSAGE
1	1	Measurement.UnitConceptId	Unrecognized or missing system
2	2	Measurement.UnitConceptId	Unrecognized or missing system
3	3	Measurement.UnitConceptId	Unrecognized or missing system
4	5	Measurement.UnitConceptId	Unrecognized or missing system
5	6	Measurement.UnitConceptId	Unrecognized or missing system
6	8	Measurement.UnitConceptId	Unrecognized or missing system
7	11	Measurement.UnitConceptId	Unrecognized or missing system
8	12	Measurement.UnitConceptId	Unrecognized or missing system
9	13	Measurement.UnitConceptId	Unrecognized or missing system
10	14	Measurement.UnitConceptId	Unrecognized or missing system

CSV

 Download

Figure 1. Prompting for Issues



Figure 2. Prompting for Issues with Summary



Your conversion warnings and issues fall mainly into two categories: unit concept recognition problems in measurements, and missing race/ethnicity data for persons. Here's a detailed explanation and guidance on how to address each:

1. Measurement.UnitConceptId warnings with "Unrecognized or missing system":

- Explanation: The OMOP CDM requires measurement units to be mapped to standard concepts, typically from the UCUM (Unified Code for Units of Measure) vocabulary or other standard vocabularies. The warning "Unrecognized or missing system" means that the units in the FHIR `valueQuantity` are either not recognized because their coding system is missing or the unit codes themselves are not mapped to a standard OMOP concept.

- Impact: Without proper unit concept mapping, measurement values cannot be accurately interpreted or compared across datasets, reducing data quality and analytic validity.

- Common units affected: %, nmol/l, U/l, mg/l, fl, g/l, pg, umol/l, E9/l, mm/h, ug/g, ug/l.

- Resolution steps:

- Verify that the source FHIR data includes the unit code and the coding system (e.g., UCUM) in the `valueQuantity.unit` and `valueQuantity.system` fields.

- Ensure the transformation logic supports and recognizes these unit codes and their systems. For example, "nmol/l" should be coded with the UCUM system "<http://unitsofmeasure.org>".

- If units are missing the system or use non-standard codes, consider enhancing the ETL to normalize or map them to standard UCUM codes.

- Update or extend your OMOP vocabulary mappings with missing units if they are legitimate but not yet included.

- Review the InterSystems OMOP transformation documentation on Measurement unit mapping for specific guidance on supported units and system handling.

2. Person.RaceConceptId and Person.EthnicityConceptId warnings with "No source data found":

- Explanation: These warnings indicate that the FHIR data did not provide race or ethnicity information in the expected extensions, so the transformation cannot assign a corresponding OMOP concept ID.

- Impact: Missing demographic data can limit stratified analyses and bias interpretation, especially in US datasets where race/ethnicity are important covariates.

- Resolution steps:

- Confirm whether the source FHIR data includes race and ethnicity extensions compliant with the US Core or other relevant profiles.

- If the data source legitimately lacks this information (e.g., European data where race/ethnicity is not collected), these warnings can be safely ignored.

- If race/ethnicity data should be present, verify the extensions' structure and paths to ensure the ETL extracts them correctly.

- Consider enhancing your ETL to handle alternative representations or extensions of race/ethnicity if your data source uses different profiles.

- If possible, collaborate with data providers to improve race/ethnicity capture at the source.

Summary:

- Unit mapping issues primarily arise from missing or non-standard unit system codes in FHIR measurement values. Fixing this requires ensuring proper unit coding and updating your transformation or vocabulary mappings.

- Race and ethnicity warnings are due to missing data in FHIR extensions; address this by verifying data presence, ETL extraction logic, and acknowledging regional differences in data collection.

By addressing these two areas, your OMOP conversion quality and completeness will improve significantly.

Figure 3. Prompting How to Remediate

Limitations & Future Directions

While this project demonstrates the potential of Generative AI to accelerate and clarify FHIR-to-OMOP transformation troubleshooting, several limitations must be acknowledged.

Alignment with OHDSI Conventions

One challenge is that GenAI outputs are not inherently constrained to OHDSI community conventions such as THEMIS or consensus decisions from working groups. Although our implementation incorporated knowledge sources including *The Book of OHDSI* and the HL7 FHIR-to-OMOP Implementation Guide to ground the model, there remains a risk of divergence. Future work will include a validation layer that explicitly cross-references GenAI-generated recommendations against existing THEMIS conventions and community-adopted mappings to improve consistency and validity.

Structural Differences Between FHIR and OMOP

The Vulcan FHIR-to-OMOP working group calls out that some structural differences between the two models make certain mappings inherently ambiguous or infeasible. Our approach surfaces these issues more rapidly, but it cannot resolve cases where no community consensus exists. Future iterations of this work could flag these “irreconcilable mappings” more clearly and direct implementers to open OHDSI working group discussions where collective resolution is needed.

Future Directions

Looking forward, this solution could be expanded beyond FHIR-to-OMOP transformations to other ETL contexts within OHDSI and generically represent a workflow to ETL remediation. The real work here that would avoid divergence would be contributing to existing OHDSI projects with exception handling that is capable of being consumed in a RAG based Generative AI workflow.

Conclusion

With an evolving target of being able to take the text from anybody reporting an issue related to the transformation run, explain the warnings or errors, and subsequently provide a plan to remediation, we successfully managed to build a repeatable process using Generative AI tooling on exceptions that occur when transformation FHIR Data to OMOP in *minutes* which what would have been *days* if not weeks with back and forth communication patterns.

For organizations or implementors of FHIR to OMOP Transformation pipelines, the use of [Vanna.ai](#) to triangulate and focus on the moving parts coupled with Generative AI to save time and increase accuracy of the ETL.

References

1. vanna-ai.vanna [Internet]. GitHub; 2023–. [cited 2025 Jul 1]. Available from: <https://github.com/vanna-ai/vanna>
2. **Hanna Kimura**. Transforming with Confidence: A GenAI Assistant for InterSystems OMOP [Internet].

InterSystems Developer Community; 2025 Jun 23 [cited 2025 Jul 1]. Available from: <https://community.intersystems.com/post/transforming-confidence-genai-assistant-intersystems-omop>

3. **Ron Sweeney.** OMOP Odyssey - Vanna AI (The Underworld) [Internet]. InterSystems Developer Community; 2025 Mar 31 [cited 2025 Jul 1]. Available from: <https://community.intersystems.com/post/omop-odyssey-vanna-ai-underworld>
4. Hripcsak G, Duke JD, Shah NH, Reich CG, Huser V, Schuemie MJ, Suchard MA, Park RW, Wong IC, Rijnbeek PR, et al. *The Book of OHDSI*. Observational Health Data Sciences and Informatics; 2021.
5. HL7 International. *FHIR to OMOP FHIR IG* [Internet]. Version 0.1.0, continuous build; generated 2025 Jun 12 [cited 2025 Jul 1]. Available from: <https://build.fhir.org/ig/HL7/fhir-omop-ig/>