

Exploring Efficient and Scalable OMOP CDM Workflows by Leveraging dbt-synthea

Markian Hromiak
Georgia Institute of Technology

Aradhya Rajanala
Georgia Institute of Technology

Jacob S. Zelko
JuliaHealth

1 Introduction and Background

The OMOP CDM has established itself as an invaluable global asset within health informatics research. However, accessing it remains a pain point contingent upon hardware infrastructure that can process terabytes worth of patient data. While community endeavors (such as Eunomia) successfully mock small, limited OMOP CDM data bases, they do not give trainees an accurate sense of working with more realistic large data bases. Hence, exploring scalability in terms of practical performance and compute limitations can often constrain translating prototype analyses to production solutions. Building upon dbt-synthea, DuckDB, and Synthea [5], we introduce data management and processing workflows that may more closely mirror realistic OMOP CDM analysis scenarios while lowering barriers to access across a variety of constraints as a result of collaboration on dbt-synthea.

2 Methods

2.1 Synthetic Patient Generation

To create a synthetic patient database, we used the synthetic data generator, Synthea, to synthesize ≈ 1 million patients each with 3 year histories.

2.2 Using dbt-synthea as an ETL Process from Synthea to OMOP CDM

dbt is an open-source data transformation tool which prioritizes SQL-first approaches to data modeling and the development of robust data transformation workflows. [8] `dbt-synthea` is an extract-transform-load (ETL) project based on `dbt` which converts Synthea synthetic data into the OMOP CDM and acts as an expansion upon tools like ETL-SYNTHEA within the context of the OHDSI community. [6] While it is still under active development, we demonstrated the maturity of `dbt-synthea`'s ability to facilitate the ETL of large Synthea datasets using the following procedure:

1. Prepare the `dbt-synthea` environment to consume Synthea data.
2. Create a local DuckDB database to house ETL'd data.
3. Stage OHDSI vocabularies from ATHENA and Synthea data for ETL.
4. Execute the `dbt-synthea` ETL process.
5. Perform post-processing and optimization.

2.3 Interfacing with OMOP CDM Databases via DuckDB

DuckDB is a lightweight database management system that focuses on optimization in online analytical processing (OLAP) and uses in-process SQL without needing an explicit server startup [7]. It additionally features an expansive open-source ecosystem of community maintained extensions. Notably, the `httpserver` extension transforms a database into an OLAP HTTP server, enabling web-to-database interfacing. [4] While, `httpserver` offers this functionality, it does not provide clean integration with many commonly used scripting languages.

2.4 httpserver Interfaces

To address this, we introduce interface packages written in Python (DuckDBServer.py) and Julia (DuckDBServer.jl) that provide client-server architecture as shown in 2.4. These interfaces enable workflow automation for server management, extension installation, SSH authentication, and client HTTP requests, introducing minimal overhead.

DuckDBServerWrapper Class Diagram V0.2.0

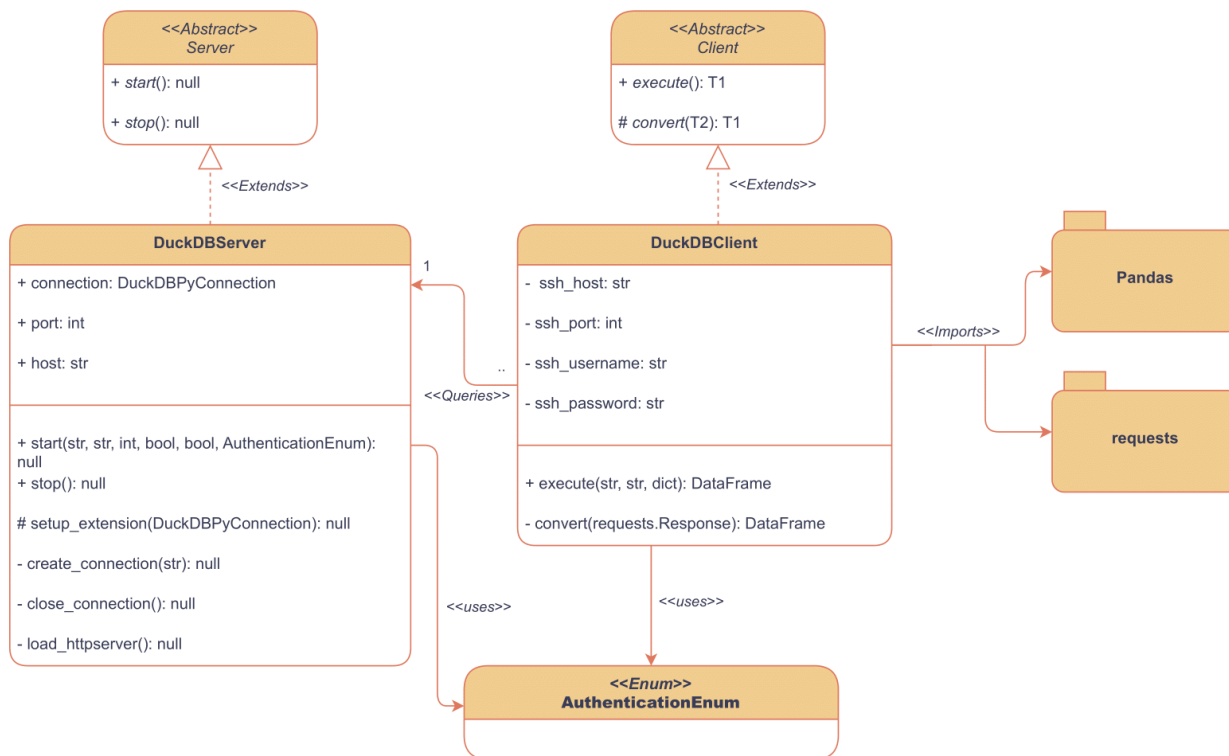


Figure 1: The current class diagram for DuckDBServer.py. The architecture is still under development as we continue to identify potential engineering requirements in workflows using such interfaces.

3 Results

Using the Synthea data generator, we produced 218 GB of synthetic patient data with 1 million living patients and 114,357 dead patients. Because `dbt-synthea` supports DuckDB, we were able to explore several optimization approaches which greatly reduced data burden. For example, we dropped several staging tables from the `dbt-synthea` ETL and, using DuckDB’s native compression, copied the processed data into a final, space-optimized version. This resulted in a 92% decrease in space – from 218GB’s to 18GB’s of data – without loss of functionality.

The packages we have developed can interface with the OMOP CDM database in a variety of ways. One of the simplest examples – querying data from the database – is shown in listing 2. Furthermore, using `DuckDBServer.jl`, we can compose this package with other in JuliaHealth tools to generate ATLAS-based cohort definitions; the first steps to doing this are shown in 2. [9] Finally, with `DuckDBServer.py` we were able to create programs which streamlined database hosting and access, demonstrating ways to integrate with Python ecosystem tools for analysis and visualization.

```

using DuckDBServer
using DuckDB

start_server("~/data/synthea_1M_3YR.duckdb")
host = "http://10.0.0.121:8080/"
body = "SELECT * FROM dbt_synthea_dev.person LIMIT 3;"
df = execute(host, body)

```

Listing 1: Querying a DuckDB database using DuckDBServer.jl. For this example, this lightweight server is configured to run on a Raspberry Pi 4 computer.

```

from duckdbserverwrapper import *

#Done on the hosting side
server : DuckDBServer = DuckDBServer()
server.start("myDatabase.db")

#Created by users to access the hosted DB. Password optional if using X-API-Key header
client : DuckDBClient = DuckDBClient("ssh_host", "ssh_port", "username", "password")
data : DataFrame = client.execute("SELECT * FROM dbt_synthea_dev.person LIMIT 3;", \
    {"X-API-KEY": "GPBurdell"})

```

Listing 2: Querying a DuckDB database using DuckDBServer.py. The server can be hosted at and accessed from anywhere as long as an endpoint URL is exposed (i.e. a website).

4 Conclusion

With the flexibility enabled by dbt-synthea, we illustrate how pairing traditional OMOP CDM workflows with DuckDB and interface packages can enable scalable workflows that could more closely mimic realistic analysis scenarios. Furthermore, we highlight how, given the scalability of dbt-synthea, individuals can now build a variety of different software tools that leverage the high performance offered by DuckDB. In future work, dbt-synthea can be further optimized and enable interoperability with OHDSI tools such as Data Quality Dashboard to produce data sets that are both efficient and high quality. [10] Finally, we plan to continue development of lightweight interface packages within Python, Julia, and R to allow further analysis workflows to take place both on local and remote servers. In conclusion, alongside ongoing collaborative efforts, we plan to continue lowering barriers to access by providing a means for hosting more realistic but space-efficient OMOP CDMs on resource limited devices, easy set-up and access to OMOP CDM databases on servers, and compatibility with ongoing OHDSI efforts.

5 Acknowledgments

We would also like to thank Katy Sadowski for her work in dbt-synthea and support in enabling such workflows as we demonstrate here to be built.

References

- [1] DeFalco, F. et al. (no date) Standard dataset manager for Observational Medical Outcomes Partnership Common Data Model Sample datasets, Standard Dataset Manager for Observational Medical Outcomes Partnership Common Data Model Sample Datasets. Available at: <https://ohdsi.github.io/Eunomia/>.
- [2] Hallinan, Christine Mary et al. "Seamless EMR data access: Integrated governance, digital health and the OMOP-CDM." *BMJ health & care informatics* vol. 31,1 e100953. 21 Feb. 2024, doi:10.1136/bmjhci-2023-100953

- [3] Keloth, Vipina K et al. “Representing and utilizing clinical textual data for real world studies: An OHDSI approach.” *Journal of biomedical informatics* vol. 142 (2023): 104343. doi:10.1016/j.jbi.2023.104343
- [4] Quackscience (no date) Quackscience/duckdb-extension-httpserver: Duckdb HTTP API server and query interface in a community extension, GitHub. Available at: <https://github.com/quackscience/duckdb-extension-httpserver> (Accessed: 2025).
- [5] Walonoski, J. et al. (2017). Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record. *Journal of the American Medical Informatics Association*. 1-9. 10.1093/jamia/ocx079.
- [6] Sadowski, K. (2024) DBT for OMOP phase I: DBT-Synthea, OHDSI. Available at: <https://www.ohdsi.org/2024showcase-124/>.
- [7] *Why duckdb* (no date) DuckDB Foundation. Available at: https://duckdb.org/why_duckdb (Accessed: 27 June 2025).
- [8] *dbt (data build tool)* (no date) dbt Labs. Available at: <https://www.getdbt.com>(Accessed: 1 July 2025).
- [9] Datseris, G., & Zelko, J. S. (2024). Physiological signal analysis and open science using the Julia language and associated software. *Frontiers in Network Physiology*, 4, 1478280.
- [10] Blacketer, C., Voss, E. A., DeFalco, F., Hughes, N., Schuemie, M. J., Moinat, M., & Rijnbeek, P. R. (2021). Using the data quality dashboard to improve the EHDEN network. *Applied Sciences*, 11(24), 11920.