Paper RE06

Unlocking Healthcare Insights: Graph-Based Modeling and Analysis of Synthetic Patient Data

Panagiotakis Georgios, **Pfizer Inc.,** Athens, Greece
Kountouris Nikolaos, **Pfizer Inc.,** Athens, Greece
Puri Maria, **Pfizer Inc.,** Athens, Greece

ABSTRACT

In the pharmaceutical domain, **real-world data (RWD)** spans a large, intricate web of information drawn from electronic health records, patient registries, insurance claims, and other healthcare sources. Although traditional relational databases are well-established for structured analytics, their rigid table-based schemas often create challenges for handling the complexity and dynamically evolving nature of those data. This paper explores the use of graph theory & databases for modeling synthetic data that approximate real-world patient data, without breaching any regulatory policies such as Health Insurance Portability and Accountability Act (HIPAA), and the General Data Protection Regulation (GDPR). Graph-based modeling could act as an alternative tool by explicitly capturing the naturality within relationships to healthcare data, enabling scientists to perform analyses effectively. In this work, graph data science algorithms are being used for extracting "hidden" information from data. Three different tasks are being showcased; a step-by-step graph-based modeling of relational synthetic patient data, a comparison between graph databases (GDBs), and relational databases (RDBs), and a report on performance.

1. INTRODUCTION

Over the past decade, life-science has moved from a trickle of billing sheets and narrow registries to a flood of data. Nowadays, almost all hospitals in the US and circa four out of five office-based physicians have started using electronic health record systems, adding millions of encounter notes, prescribing events and other health-related information each day [2]. At the same time, insurance companies, digital devices, genomic pipelines, and even social media forums inject new details into the equation. This data is notable not only for its size, but also for its diversity. It covers everything from lab findings and diagnoses to multi-year treatment plans that include primary care, specialists, hospitals, and pharmacies. However, tight privacy policies such as HIPAA and GDPR continue to restrict the amount of information that may be used, leading to increasing interest in high-fidelity synthetic alternatives that mirror the statistical qualities of real-world data without exposing any individual information [3]. When it comes to analyzing data, traditional databases remain the primary tool for day-to-day tasks, but their restricted table-based schema makes it difficult for scientists to explore semantic information on all these many-to-many interactions. Research shows that SQL queries with few joins, often used in healthcare (e.g., tracking adverse event chains or drug-condition interactions), can significantly slow performance due to the time spent optimizing these queries. Table formats can obscure data relationships and patterns even when the information is visible. Graph theory has yet to be extensively studied and applied in this field. Graphs excel at preserving and representing the natural interactions of real-world data, allowing for greater visual efficiency in understanding the relationship between concepts and identifying patterns. In addition, graph data science algorithms like centrality, community detection, and path discovery can help with uncovering hidden patterns. In terms of performance, graph databases treat relationships as fundamental components, enabling traversal times to remain nearly constant. The remainder of this article is organized as follows: Section 2 begins with an introduction of what real-world and synthetic data are. Section 3 contains an introduction to graphs and graph databases. Section 4 presents a set of applied exercises using synthetic data produced by **Synthea** to evaluate graph databases. The section details three tasks: (1) a systematic approach to modelling a small sample of synthetic data, (2) the ingestion of a larger synthetic dataset into both relational and graph databases with an analysis of performance considerations, and (3) the application of graph data science algorithms to demonstrate their utility in uncovering latent insights.

2. Introduction to Real World Data & Synthetic Data

2.1 What is Real-World Data & Real-World Evidence?

Real-World Data (RWD) is defined as health-related information generated during ordinary clinical care and outside of traditional trials, including electronic health records (EHRs), insurance claims, registries, data from personal devices and others. The U.S. **Food and Drug Administration (FDA)** define RWD as "data relating to patient health status and/or the delivery of health care routinely collected from a variety of sources" and the **European Medicines Agency (EMA)** express it as observational data collected in real-world repositories. By analyzing RWD, we generate **real-world evidence (RWE),** which refers to clinical conclusions about a product's use, benefits, or risks in everyday practice that supplement efficacy results from randomized controlled trials (RCTs) [4][5][6][7].

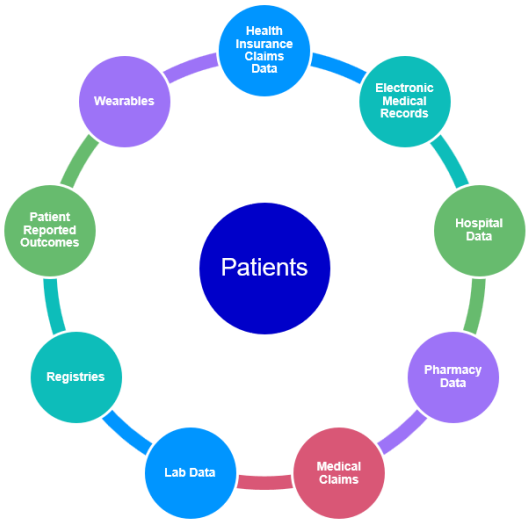

Figure 1: Overview of Real-World Data Sources

2.2 Introduction to Synthetic Data

Synthetic data refers to artificially generated information intended to statistically mirror the features and patterns present in real-world datasets, without risking the exposure of any real patient information [8]. When it comes to synthetic patient data, they encompass clinical or biological elements such as demographics, bio-signals (EEGs), lab results, and others. Their primary purpose is to safeguard patient identities, ensuring that synthetic information cannot be de-identified and linked back to real patients. There are multiple methods for generating synthetic datasets, such as: i) statistical simulations that sample from fitted distributions, ii) generative artificial intelligence (AI) models (e.g., Variational Autoencoders), iii) rule-based models (e.g., Synthea) that simulate life courses via probabilistic clinical state machines [9][10][11]. The motivation for using synthetic data is privacy preservation and regulatory compliance. Synthetic data in healthcare are increasingly recognized as a privacy-focused solution for research and development. This approach enables researchers to utilize realistic datasets for algorithm prototyping, training, testing, or profiling without compromising actual patient information. Although these data are valuable for research, there are often concerns regarding the quality and the rigor of training applied to the models that generate them. Concerns around fairness & bias arise when the original datasets that were used contained inherent biases, such as under-representation of certain groups, where such biases can be amplified in the synthetic data, leading to distorted distributions of characteristics across different subgroups. Privacy is another issue that can arise. The purpose of synthetic data is to closely mirror real-world data; however, this approach also raises concerns regarding the potential disclosure of actual patient information. Finally, fidelity is another topic that is often discussed; that is how efficient and successfully these models maintain the complexity and statistical properties of the original data [8][12][13][14][15].

2.3 What is Synthea?

For this paper, the synthetic data that will be used are generated from a tool called **Synthea**. Synthea is a rule-based synthetic data generator which has started becoming a benchmark across the healthcare domain for generating high-quality and clinically realistic but entirely synthetic patient datasets. It is free of cost and privacy restrictions that support both industry and academia. It works by modeling health conditions, e.g. diabetes as module states, which can be depicted as a flowchart of clinical states and transitions that patients can experience. Finally, it provides different output options with some of them but not limited to **FHIR** and **CSV** files [9][16].

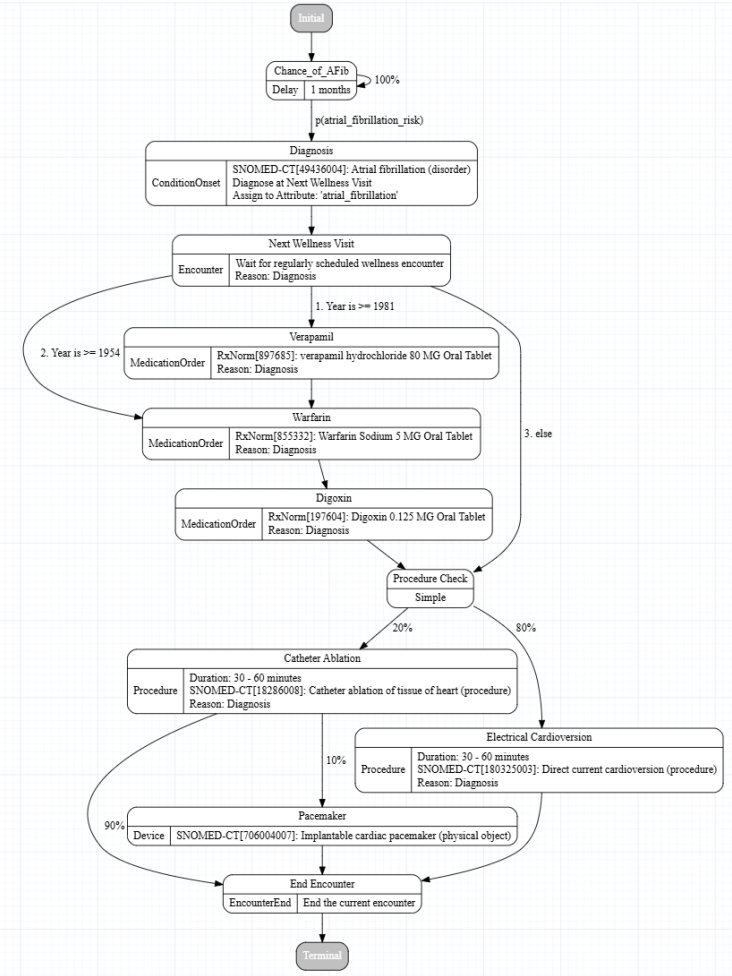

Figure 2: Atrial Fibrillation Module from Synthea
Synthea Generic Module Builder accessed 10/1/2025

3. A Graph Approach for RWD

3.1 What are graphs?

In discrete mathematics, graph theory studies structures called graphs consisting of **nodes** (also known as vertices, entities) and **edges** (also known as connections, relationships) between nodes. Graphs can represent systems like healthcare networks, where nodes stand for patients, providers, and conditions, and edges show connections such as procedures or referrals. Graphs have different properties and forms; they can be directed or undirected, having weighted or unweighted relationships, homogeneous or heterogeneous, including loops, etc.

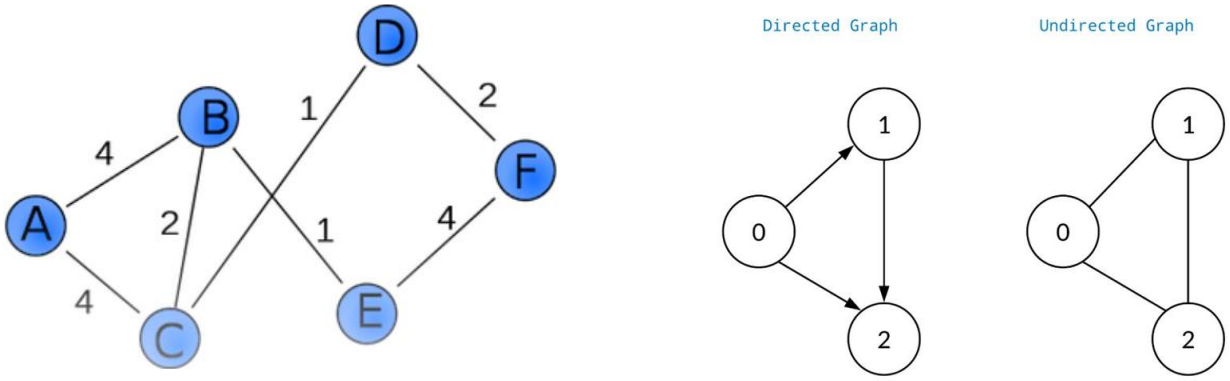

Figure 3: Illustration of a weighted graph, Directed vs. Undirected Graphs | Overview, Examples & Algorithms - Lesson | Study.com accessed 10/1/2025

Figure 4: Overview of Directed and undirected graphs, Graphs UMPIRE Cheat Sheet | CodePath Cliffnotes accessed 10/1/2025

When it comes to analyzing graph structures, there are algorithms that often focus on path analysis such as shortest path finding, community detection by identifying groups of entities, centrality algorithms that measure the importance of the nodes within the network and this can identify also nodes that act as bridges. Another important type is similarity algorithms that measure how similar nodes are based on their neighborhoods [1][17][18][19].

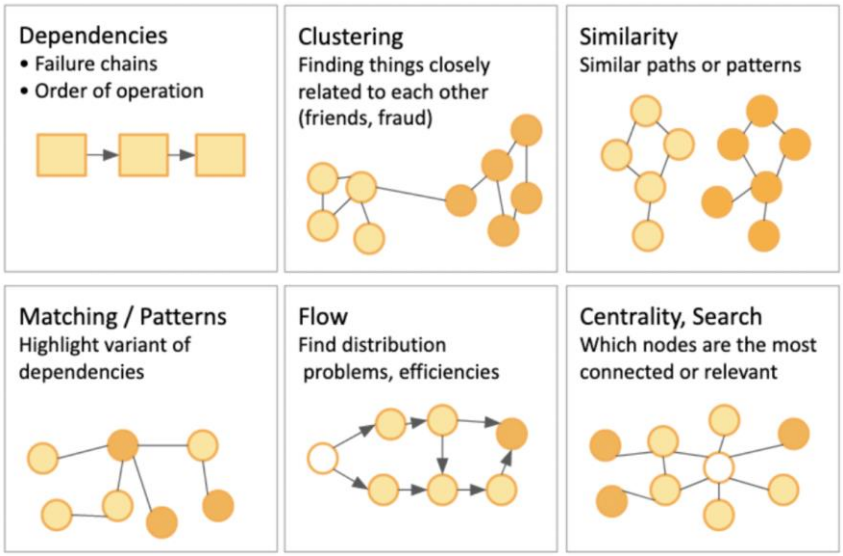

Figure 5: Different categories of graph algorithms, Data Science with Graphs - using knowledge graphs on the data before it reaches the ML phase - deepsense.ai accessed 10/1/2025

3.2 Introduction to Graph Databases

A **graph database** is optimized to store and query graph-structured data, data whose inherent structure is a network of entities and relationships. Unlike relational databases with rigid tables, graph databases represent information as nodes and edges and represent them as a **property graph or RDF model**. This makes them well-suited for heterogeneous, highly connected data such as healthcare records [20]. For example, one can store patients, doctors, diagnoses, and medications as nodes, and use edges to capture relationships such as “doctor X TREATS patient Y” or “patient Y HAS diagnosis Z”, with labels for characterizing nodes and properties for individual node information. These explicit relationships eliminate the need for expensive JOIN operations and allow traversing connections directly. Indeed, graph databases “retain data’s natural richness and complexity” by mapping real-world entities and their interactions without forcing them into tables [20]. Graph databases support specialized query languages for navigating and matching patterns in the graph. For instance, **Cypher** (used in Neo4j) allows declarative pattern matching (e.g. MATCH (patient)- [: HAS_CONDITION]->(diagnosis)) [21]. From 2024, **graph query language (GQL)** became the first new ISO graph database query language in 35+ years [20][21][22][23].

3.3 Introduction to Graph data modeling

Designing an effective graph model is both **art and science**. Graph data modeling is the process of organizing data into a graph database, with entities (nodes) and their connections (relationships) at its heart. Unlike relational databases, which store data in rigid tables with predefined schemas, graph databases natively model real-world relationships, making it easier to browse and analyze linked data. Graph modeling aims to optimize graph structure and enhance computational performance.

For effective graph data modeling and ingestion, the recommended basic steps are [24][25]:

- **Identifying business questions:** This is the first and most critical step: determining why graphs are needed in the first place. Most of the time, the questions we care about revolve around graph analytics or multi-modal data sources, but performance can be an issue, particularly in use cases that frequently require recurrent

queries or strong interconnected entities. Examples of business questions that can help determine if graphs are needed could be:
    - **Which providers serve as bridges?**
    - **What patient clusters emerge naturally?**
    - **What clusters of comorbidities conditions emerge?**
- Identify the main entities & relationships that represent business logic
- Load a small sample of data and test the defined questions.
- Iteratively add more data and optimize the schema
- Engage with **domain experts** from early steps
- **Optional:** when focusing on Graph Data Science modify the graph schema accordingly.

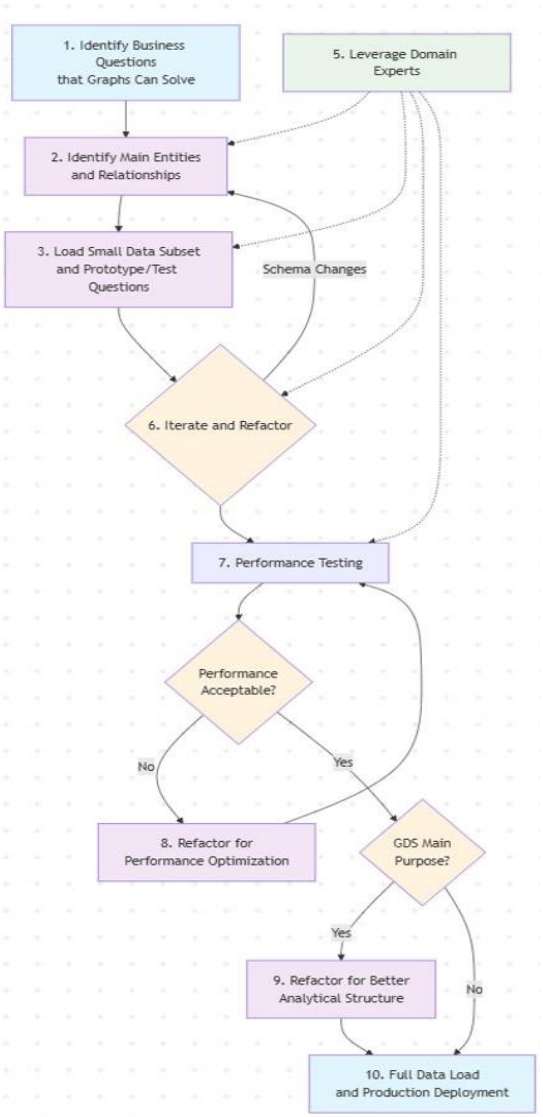

Figure 6: Graph Modeling Typical flow

Even if all the above steps have been followed, there might be performance degradation. In such cases the following steps can be implemented:

- Using **multiple labels** rather than properties. Ideally, each node should have less than ~10 properties.
- **High Cardinality** properties as nodes
- **Low Cardinality** properties as relationships
- Use **distinctively** relationships e.g. **avoid** relationships such as **is_a** or **type_of**
- Introducing **Intermediate nodes**
- If there are **list properties** convert them into **nodes**
- Utilize chain relationships next/previous e.g. **next_encounter**. This will eliminate the need for joining encounter nodes.
- **Avoid High Density nodes** - performance issues due to path explosion.

- **Avoid Bi-directional** relationships

4. A Practical Example

This paper introduces graph theory and includes practical tasks to help readers become familiar with its concepts. Specifically, three different tasks will be addressed:

- **Task 1:** Step-by-step transforming a dataset of relational synthetic patient data generated by **Synthea** into a graph.
- **Task 2:** Produce an expanded dataset of patients, import the data into both a relational database and a graph database, and evaluate the performance trade-offs between the two systems.
- **Task 3:** Define two example business questions that are mentioned above and apply graph-data-science algorithms to these questions and explore how and if graphs indeed help uncovering "hidden" information.

	Id	BIRTHDATE	DEATHDATE	FIRST	MIDDLE	LAST	RACE	ETHNICITY	GENDER	STATE
0	29a4eca4-d77c-84c5-3a2c-5e3a9444492f	2022-07-04	NaN	Stephan	NaN	Hermiston	white	nonhispanic	M	Massachusetts
1	c7f16b77-ca5a-a58f-5759-3fbfb3ed627e	2024-11-16	NaN	Bill	NaN	Hahn	white	nonhispanic	M	Massachusetts
2	277f8b94-131b-ed59-3042-6ab418c9fe7c	2022-12-31	NaN	Leigh	NaN	Volkman	white	nonhispanic	M	Massachusetts
3	8ca41df2-7a24-2a39-e65f-d729f9217f9c	2006-02-17	2018-02-24	Edward	Jordon	Tillman	white	nonhispanic	M	Massachusetts
4	9bbb9c38-8a5a-c1b7-c1f5-67b5f4f16a09	2009-10-24	NaN	Carleen	NaN	Rowe	white	nonhispanic	F	Massachusetts

Figure 7: Snapshot of the patient synthetic table

	Id	START	STOP	PATIENT	ORGANIZATION	PROVIDER	PAYER	ENCOUNTERCLASS	CODE	DESCRIPTION	BASE_ENCOUNTER_COST	TOTAL_CLAIM_COST
0	fa6bfaa0-e141-dc59-23a0-ea084ef7ec26	2022-12-31T00:40:11Z	2022-12-31T00:55:11Z	277f8b94-131b-ed59-3042-6ab418c9fe7c	817a1428-2045-3e89-9fa2-ef7d287d6fa5	685d04c7-dc44-347c-8130-723524d6792a	26aab0cd-6aba-3e1b-ac5b-05c8867e762c	wellness	410620009	Well child visit (procedure)	136.8	347.38
1	ae49bf52-6c2c-9272-a681-2cec66504951	2022-07-04T17:06:04Z	2022-07-04T17:21:04Z	29a4eca4-d77c-84c5-3a2c-5e3a9444492f	db60692f-cbef-315b-9fce-38a7f015c482	b7ba653d-db19-3382-b57f-deadf17aaa18	b046940f-1664-3047-bca7-dfa76be352a4	wellness	410620009	Well child visit (procedure)	136.8	563.08
2	f98e2946-4a95-29d3-c702-2bfeff9c6bb6	2024-11-16T06:14:20Z	2024-11-16T06:29:20Z	c7f16b77-ca5a-a58f-5759-3fbfb3ed627e	d3085315-9893-34c7-8889-3a31dc17c2b0	224244b5-fb3a-3893-ad62-4e1f5b7ed5d4	e03e23c9-4df1-3eb6-a62d-f70f02301496	wellness	410620009	Well child visit (procedure)	136.8	347.38
3	7a396b4e-c0bf-a9dc-8d31-81f9a293c20f	2024-12-21T06:14:20Z	2024-12-21T06:29:20Z	c7f16b77-ca5a-a58f-5759-3fbfb3ed627e	d3085315-9893-34c7-8889-3a31dc17c2b0	224244b5-fb3a-3893-ad62-4e1f5b7ed5d4	e03e23c9-4df1-3eb6-a62d-f70f02301496	wellness	410620009	Well child visit (procedure)	136.8	272.80
4	56464876-94e9-ba06-5dae-4a33ee29b62e	2022-08-08T17:06:04Z	2022-08-08T17:21:04Z	29a4eca4-d77c-84c5-3a2c-5e3a9444492f	db60692f-cbef-315b-9fce-38a7f015c482	b7ba653d-db19-3382-b57f-deadf17aaa18	b046940f-1664-3047-bca7-dfa76be352a4	wellness	410620009	Well child visit (procedure)	136.8	705.54

Figure 8: Snapshot of the encounter synthetic table

	START	STOP	PATIENT	PAYER	ENCOUNTER	CODE	DESCRIPTION	BASE_COST	PAYER_COVERAGE	DISPENSES	TOTALCOST	REASONCODE	REASONDESCRIPTION
0	2024-01-19T09:02:14Z	NaN	277f8b94-131b-ed59-3042-6ab418c9fe7c	26aab0cd-6aba-3e1b-ac5b-05c8867e762c	85970e06-ebbb-64ea-bcdc-22a5f1781fb7	1014676	cetirizine hydrochloride 5 MG Oral Tablet	382.20	0.00	18	6879.60	NaN	NaN
1	2024-01-19T09:02:14Z	NaN	277f8b94-131b-ed59-3042-6ab418c9fe7c	26aab0cd-6aba-3e1b-ac5b-05c8867e762c	85970e06-ebbb-64ea-bcdc-22a5f1781fb7	1870230	NDA020800 0.3 ML Epinephrine 1 MG/ML Auto-Injector	105.37	0.00	18	1896.66	NaN	NaN
2	2008-01-08T20:43:46Z	2008-01-22T20:43:46Z	8ca41df2-7a24-2a39-e65f-d729f9217f9c	df166300-5a78-3502-a46a-832842197811	99040467-a0e9-e7bb-b9d3-112c43731d22	308192	Amoxicillin 500 MG Oral Tablet	2.51	0.00	1	2.51	NaN	NaN
3	2008-01-08T20:43:46Z	2008-01-22T20:43:46Z	8ca41df2-7a24-2a39-e65f-d729f9217f9c	df166300-5a78-3502-a46a-832842197811	99040467-a0e9-e7bb-b9d3-112c43731d22	198405	Ibuprofen 100 MG Oral Tablet	87.32	37.32	1	87.32	NaN	NaN
4	2013-05-22T20:43:46Z	2013-06-21T20:43:46Z	8ca41df2-7a24-2a39-e65f-d729f9217f9c	df166300-5a78-3502-a46a-832842197811	c9be1cbf-faf8-90ca-d3b2-a6fc4293d77e	313820	Acetaminophen 160 MG Chewable Tablet	45.86	0.00	1	45.86	NaN	NaN

Figure 9: Snapshot of the medication synthetic table

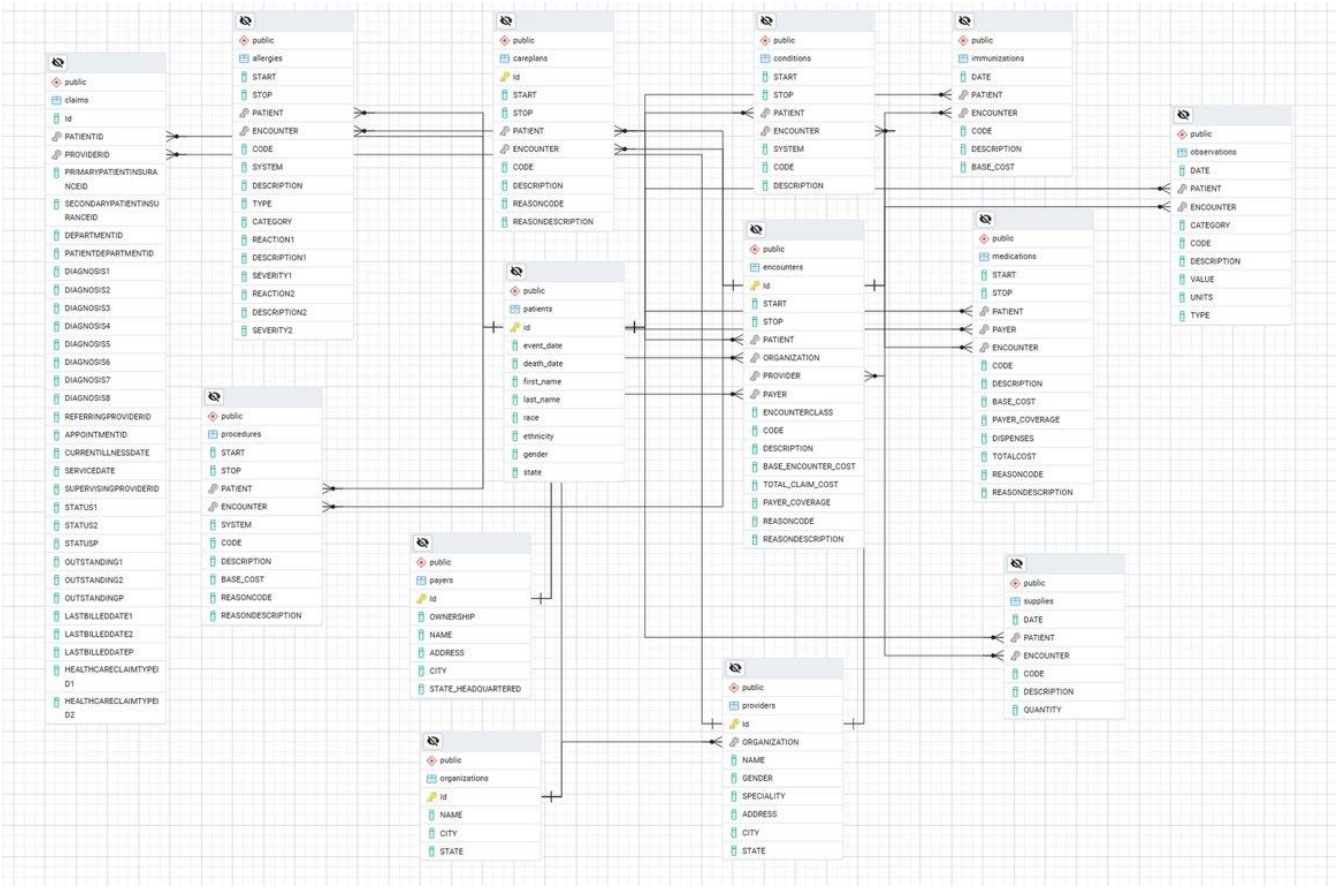

Figure 10: Synthea's relational schema. **Note** that we did not use all the tables generated but only the above ones for all the tasks below.

Before continuing, there are some clarifications that the audience should know.

- **All the data** used and displayed were generated by Synthea, thus they are "**fake"/synthetic** data and **free of cost and privacy restrictions.**
- Our objective is not to establish new scientific findings, but to examine graph modeling concepts and algorithmic methodologies.
- The databases that were used for all the experiments were **PostgreSQL** (as the RDB) and **Neo4J** (GDB).

4.1 Task 1: Step by step modeling

The first practical task is to showcase how synthetic patient data can be modeled as a graph. We mentioned in the modeling part that the first and most critical step should be regarding answering business questions to establish whether graphs are required. However, in this task, we simply aim to transform relational data into a graph structure.

4.1.1 Step 1: Ingesting the data

We generated a synthetic dataset for around **~100 patients** in **CSV** format. To convert the relational data into a graph structure we need to follow some steps:

- **De-Normalization** of the RDB into a lower normal form; this step is done often in analytics to reduce joins between tables.
- **Table names become labels** in our graph, where a label is an assigned type to each node e.g., Patient labels describe nodes that are referring to patients.
- **Rows** within the tables become entities
- The joins between the tables become relationships between the nodes.

After applying the first step we end up having a graph with:

- **Number of Entities**: 260.240
- **Number of Relationships**: ~505.000

```
'allergies.csv',
'careplans.csv',
'claims.csv',
'claims_transactions.csv'
'conditions.csv',
'devices.csv',
'encounters.csv',
'imaging_studies.csv',
'immunizations.csv',
'medications.csv',
'observations.csv',
'organizations.csv',
'patients.csv',
'payers.csv',
'payer_transitions.csv',
'procedures.csv',
'providers.csv',
'supplies.csv']
```

Figure 11: Files that Synthea outputs in CSV

```
Patients Table Shape ->  (127, 10)
Payers Table Shape ->  (10, 22)
Providers Table Shape ->  (289, 13)
Organizations Table Shape ->  (289, 11)
Encounters Table Shape ->  (11083, 15)
Careplans Table Shape ->  (549, 9)
Conditions Table Shape ->  (6374, 7)
Immunizations Table Shape ->  (2644, 6)
Medications Table Shape ->  (8539, 13)
Observations Table Shape ->  (144724, 9)
Procedures Table Shape ->  (29819, 10)
Claims Table Shape ->  (19622, 31)
Allergies Table Shape ->  (143, 15)
Symptoms Table Shape ->  (31179, 9)
Supplies Table Shape ->  (4849, 6)
```

Figure 12: For ~100 patients datasets table shapes

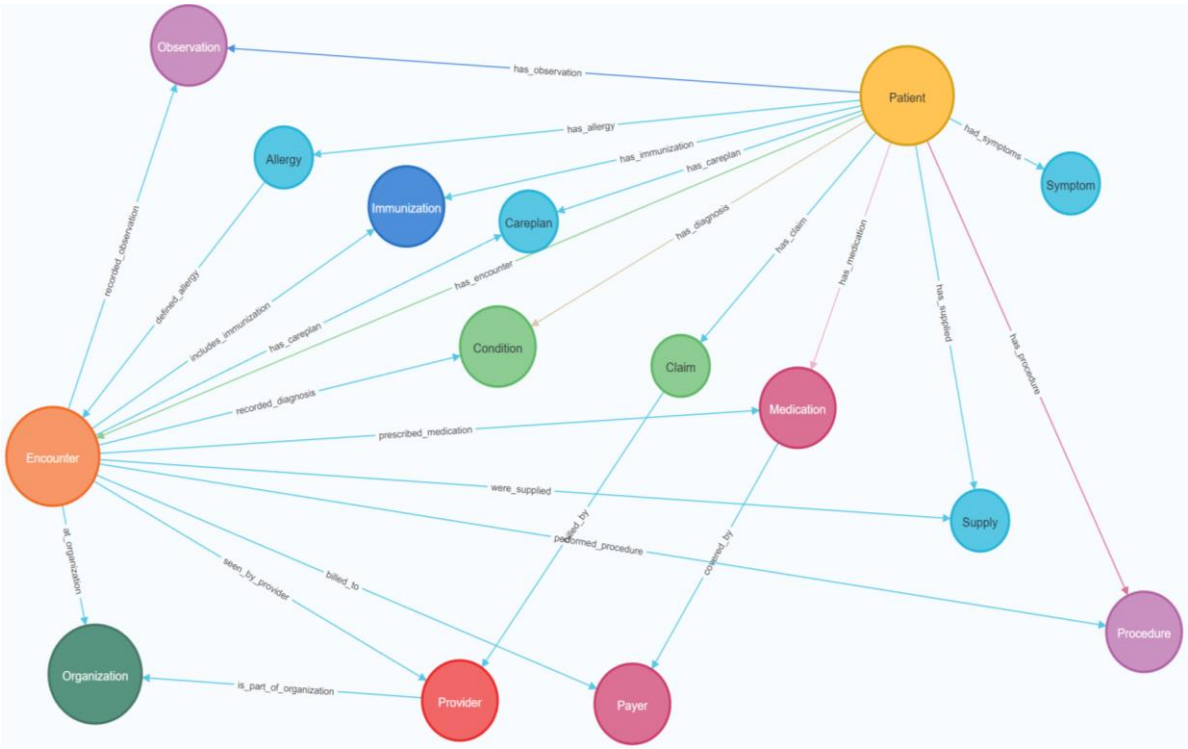

Figure 13: Snapshot of the Meta-graph (our current graph structure) for step 1

4.1.2 Step 2: De-duplication/normalization of entities

The strength of the graphs is to connect entities with each other and to create a "complete" network of information. In our approach, our graph consists of patients and their own information (e.g. patients with diabetes, having their own diabetes condition entities). To normalize this, we kept one node of health conditions that is shared across

patients, providers, and other entities. Entity normalization is an important step that is often implemented on graph analytics for using GDS algorithms such as centrality and community detection.
After the entity normalization we end up with:

- **Number of Entities**: 64.179 (Approximately **~75\% less** than the original)
- **Number of Relationships**: ~505.000

The number of relationships remains the same with no connection removed, just pointed to one health condition instance.

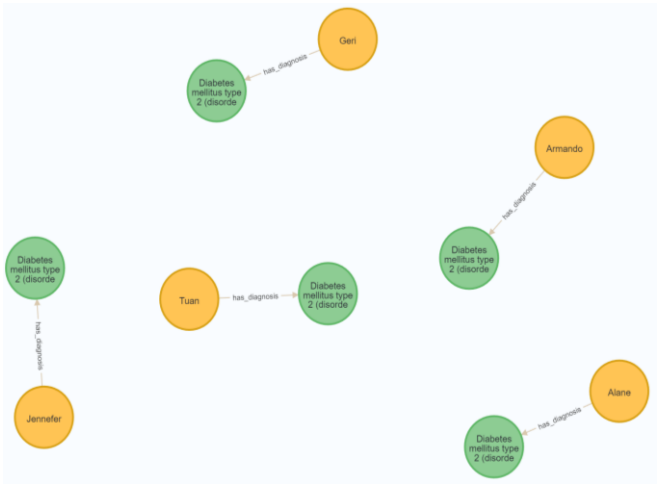

Figure 14: Conditions set before entity normalization

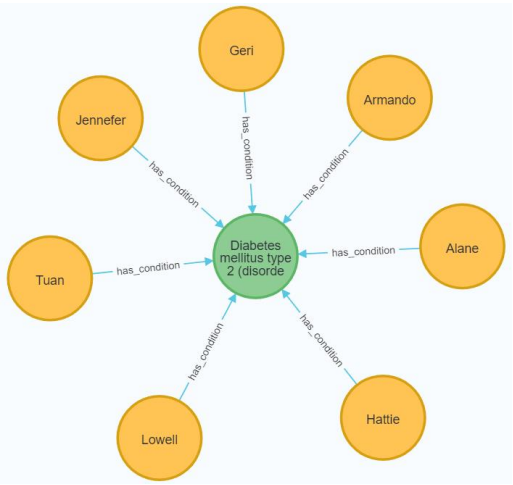

Figure 15: Normalized concepts after applying step2

4.1.3 Step 3: Removing "same" relationships

This step is optional, but we implemented it for illustration purposes. We noticed that we had some relationship with the "same" meaning, such as:

- Patient - **has_condition** -> Condition
- Patient - **has_encounter** -> Encounter - **includes_condition** -> Condition

This applies not just to conditions, but also to immunizations and procedures; however, some observations lack encounter-related details. Since these relationships share the same meaning, we identified several reasons to remove one of them:

- Improve the path explosion problem by reducing the density to the health condition nodes
- Since graph databases excel at traversal into almost constant time, there was not any execution time difference for traversing directly to condition or passing through encounter first.
- Reducing redundant relationships results to **memory improvements.**

To preserve encounter data, we removed direct links from patients to health conditions. After this step the graph looks like the following:

- **Number of Entities:** 64.179
- **Number of Relationships:** ~ 316.000

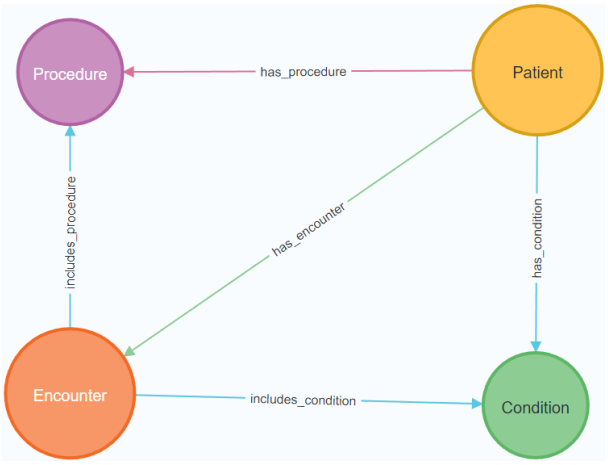

Figure 16: Overview of Semantically "same" information

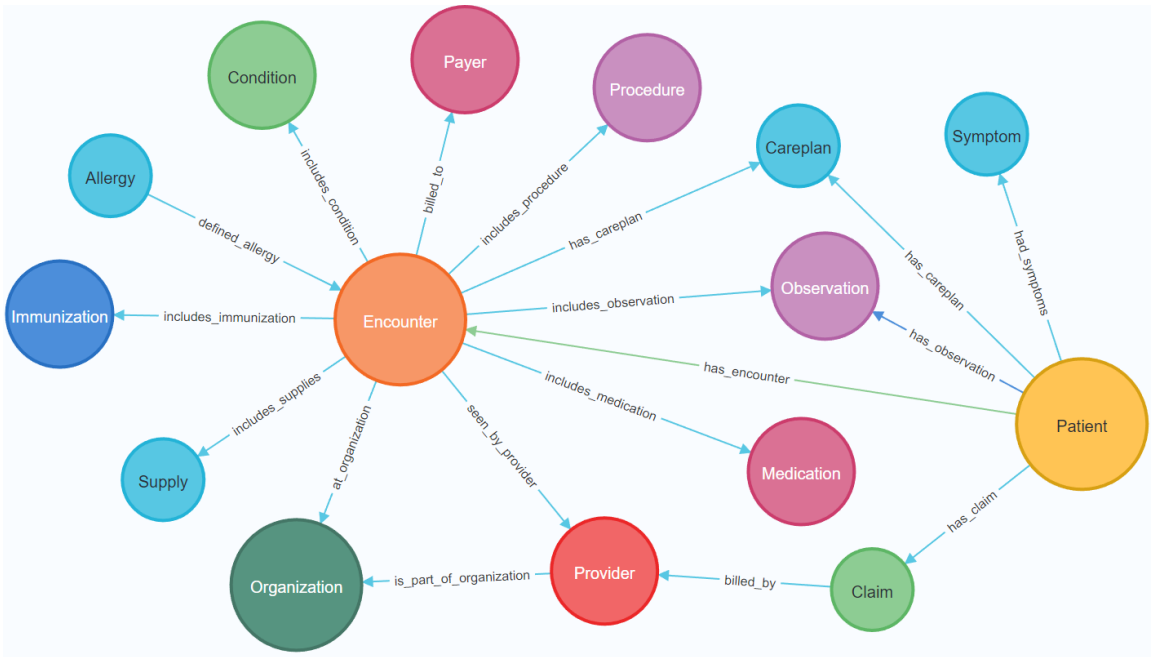

Figure 17: Meta-graph after applying step 3

4.1.4 Step 4: Introducing semantically sequential relationships

Real-world data often requires examining interactions between encounters. For this reason, we added three new relationships between encounters and patients:

- Encounter - **next_encounter** -> Encounter
- Patient - **has_first_encounter** -> Encounter
- Patient - **has_last_encounter** -> Encounter

This gave us the ability to traverse through encounters. Finally, the graph modified as:

- **Number of Entities:** 64.179
- **Number of Relationships:** 330.818

Figure 18: Sequence of encounters by adding new relationships.

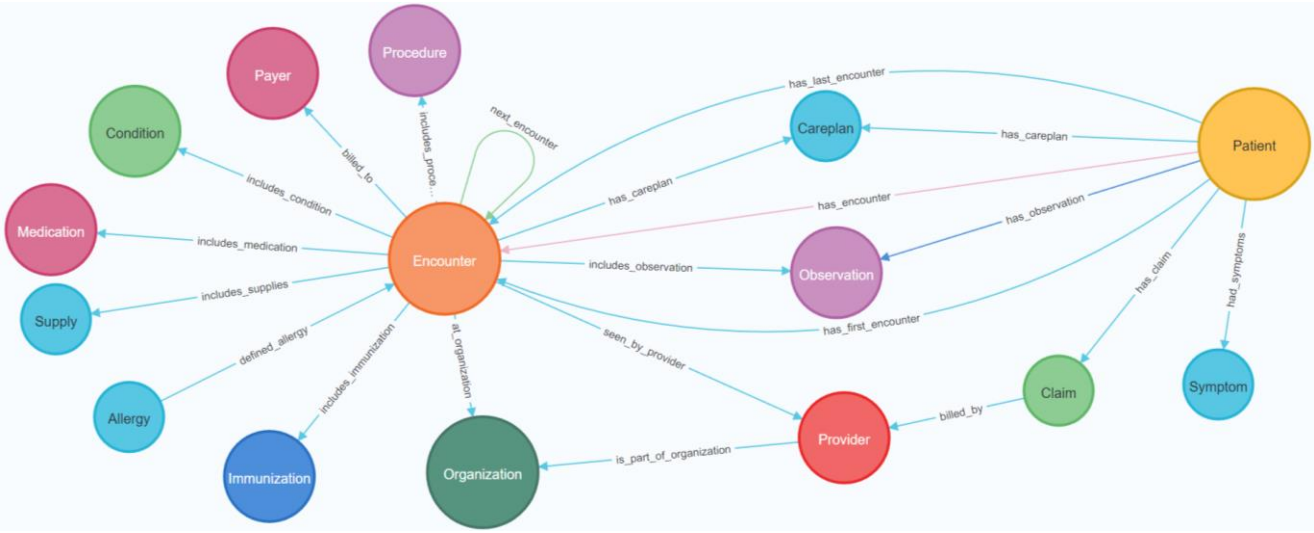

Figure 19: Meta-graph after applying step 4

4.2 Task 2: Performance comparison

We conducted a brief experiment comparing performance tradeoffs between **PostgreSQL** and **Neo4J**. Both databases were installed and configured on the same machine using similar machine resources. Further details about the experiment are provided below.

- The synthetic dataset generated was for **~80.000 patients** and size of **(~27 GB**)
- The output generated by Synthea is presented in a relational format that closely resembles real-world database structures; therefore, we maintained this format for consistency with relational systems.
- We retained the graph structure from Task 1, making two changes: converting certain properties to **labels** and adding **intermediate nodes** to preserve information.

- The queries were chosen based on multi-join information between tables to reflect real-world scenarios where most of the time we need to aggregate information from multiple tables such as patients, conditions/diagnoses, procedures, etc.
- For both the ingestion phase (data loading into each database) and the retrieval phase (querying data to answer questions), each query **was executed ten times under cold and warm cache conditions**. After that, we reported the mean execution time for each query on each cache, and then we computed the mean time from both caches.
- **Note** that this comparison was an exploratory performance assessment rather than a statistically rigorous performance evaluation. In addition, the queries established were intended solely for assessing execution time, rather than reflecting real-world results.

The graph created for the ~80.000 patients consisted of:

- **Number of Entities:** ~35.000.000
- **Number of Relationships:** ~235.000.000

The meta-graph is displayed below.

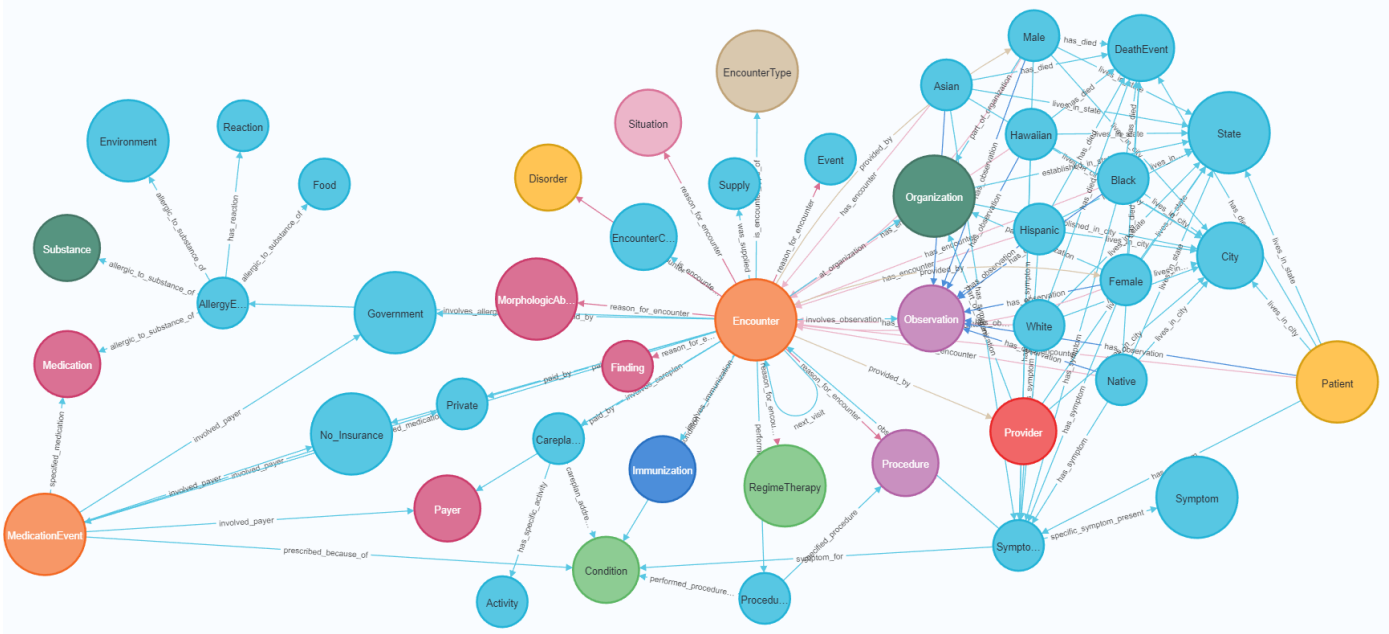

Figure 20: Meta-graph of Task 2

Several metrics were evaluated, including the following:

- **Data ingestion time.**
- **Memory Utilization** for data storage at each database.
- **Query Execution time** on the following questions:
  - **Question 1:** Identify patients who experienced **fever within 14 days after an immunization** and subsequently received an antipyretic **within 7 days**.
  - **Question 2:** Among patients with a **death event**, determine the **most frequent care pathways** in the **60 days** preceding death**.**
  - **Question 3:** Identify **pairs** of providers who tend to **co-treat the same patients for the same symptoms** within a **30-day** window**.**

From the above research questions, it is evident that each case necessitates a minimum of two or more joins. For example, the **first question** (**"Identify patients who experienced fever within 14 days after an immunization and subsequently received an antipyretic within 7 days.**") requires referencing the following tables: **patients, encounters, immunizations, medications,** and **symptoms**, which underscores the inherent complexity present in real-world healthcare analytics.

Use Case	RDB (PostgreSQL)	GDB (Neo4J)
Data Ingestion Time	**22 min**	~2.1 hours
Memory Utilization	36 GB	**32 GB**
Question 1	38.7 sec	**16.25 sec**
Question 2	17.5 sec	**10.5 sec**
Question 3	2.2 min	**15.5 sec**

Table 1: Result's table of the performance comparison between the relational and graph database. The metrics reported were regarding the ingestion time, memory utilization for storing the data to each database and a set of defined questions that approximate real-world research questions.

The above table results demonstrate that the **ingestion** time was **~x6 slower** in the Graph database. This is expected as the aim is to define the graph's structure such that it meets the requirements (or the research questions to be answered), for retrieving information at the latter steps. On the other hand, the execution time (**after the ingestion**) was **~x4 times** faster on GDB when compared to RDB. These results are consistent with existing literature, which suggests that as the data grows in size and complexity, the **faster the graphs perform** with differences reaching even **~60+ times faster than RDBs.**

4.3 Task 3: Applying Graph Data Science

The last task involved defining example business questions where graphs can reveal hidden information and applying graph data science algorithms to answer them. The questions are:

1. What **clusters** of conditions tend to **co-occur** across patients?
2. Which providers act as **chokepoints** in referral pathways?

Prior to utilizing graph data science algorithms, it was necessary to establish the appropriate relationships required for these algorithms to operate effectively. First, we established a weighted co-occurrence relationship between conditions, using pair frequency as the weight. For the provider network, we defined referrals as cases where patients switched providers within a short time frame, indicating Provider A referred to Provider B. While these definitions are simplified, they serve our purpose. Finally, we applied a community detection algorithm (**Louvain**) to the first question in order to find groups of conditions that co-occur. We also applied a centrality algorithm (**Betweeness**) to find bridges within the providers network by stating that high betweenness score suggests bridges that connect our network and that tend to introduce delays within the patient treatment flow for patients passing through those providers.

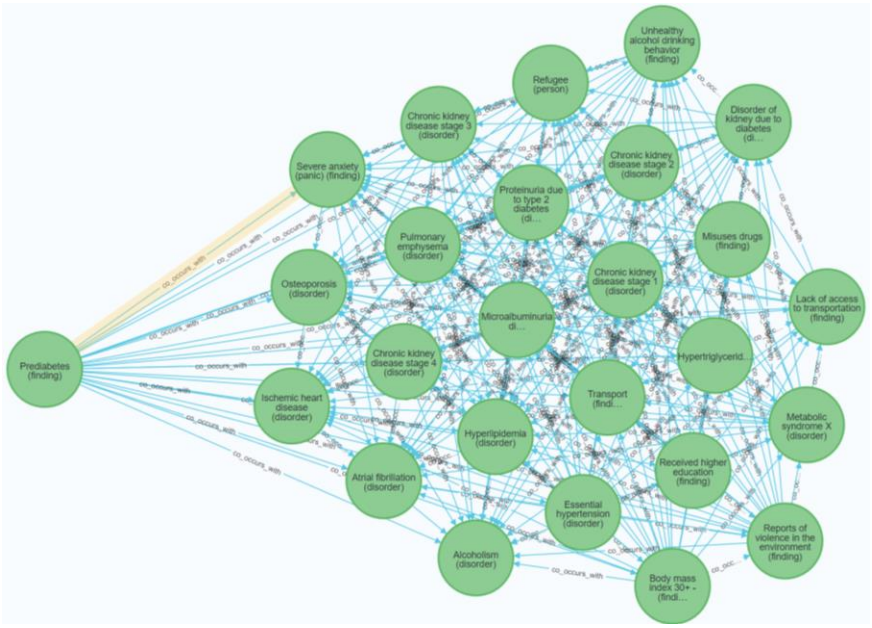

Figure 21: Created **co-occur** relationship between conditions.

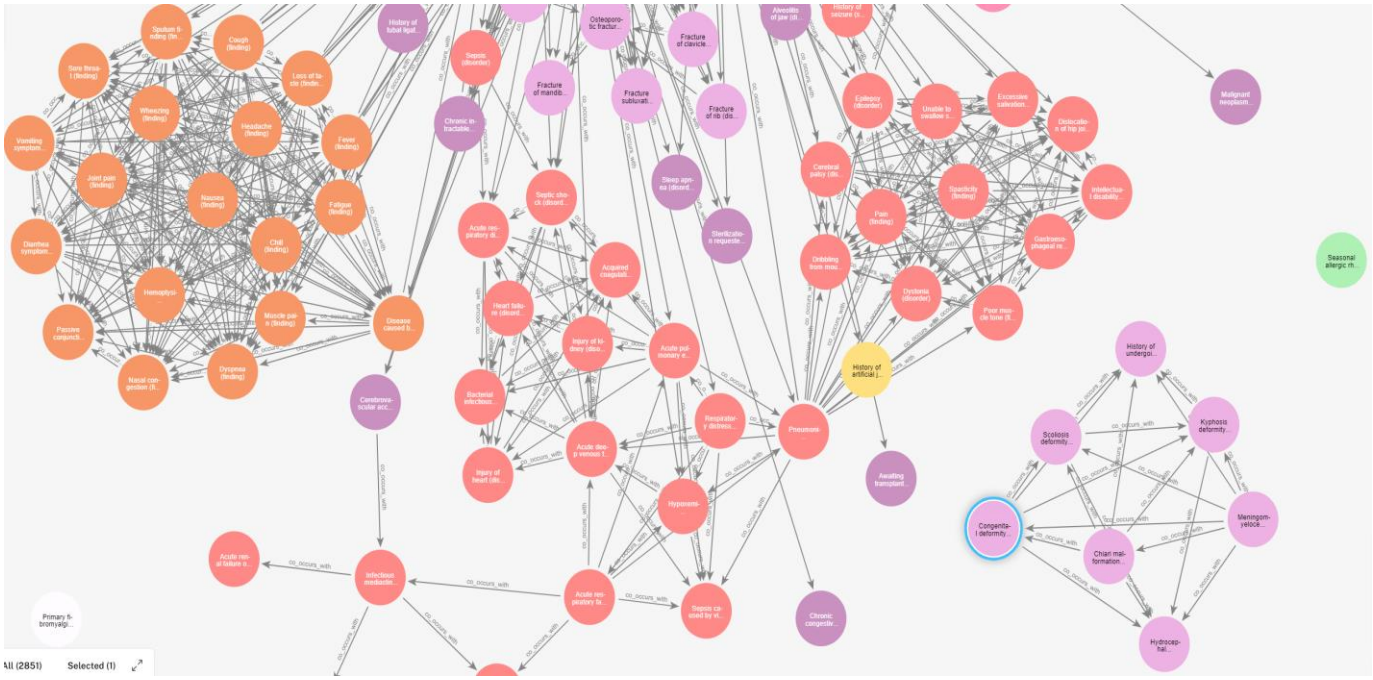

Figure 22: Snapshot from different (colored) groups of conditions after applying **Louvain** community detection algorithm.

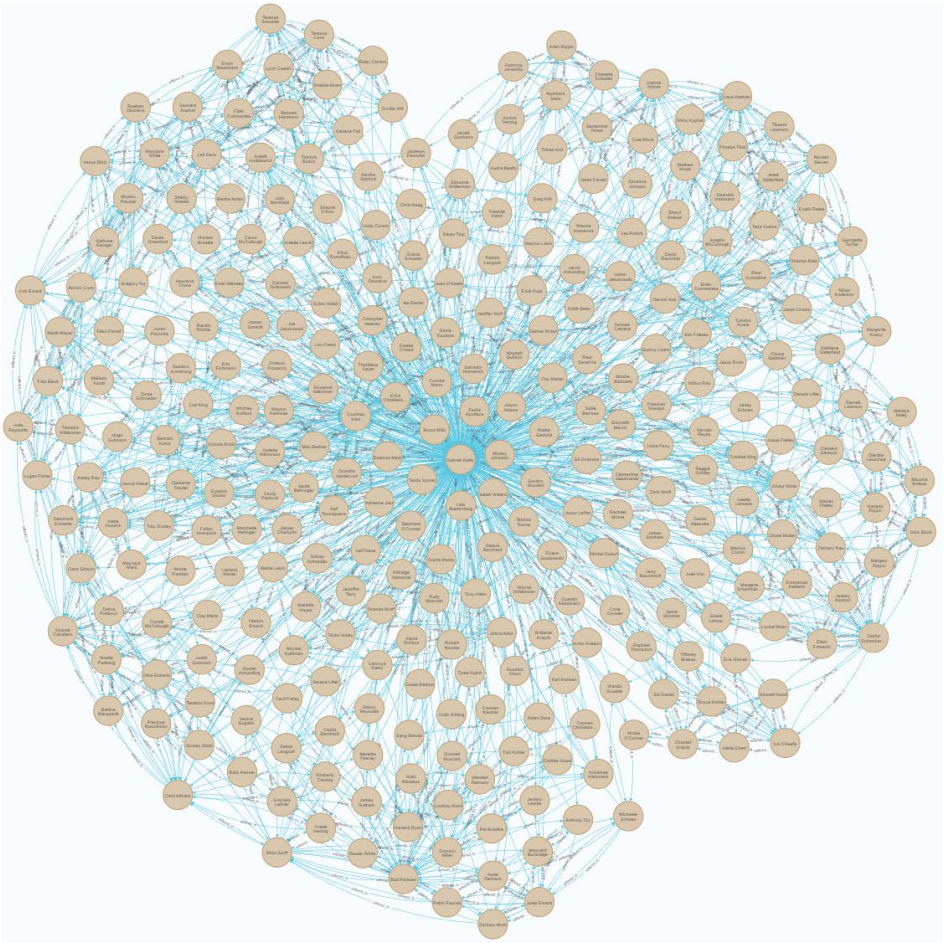

Figure 23: Providers referral network flow.

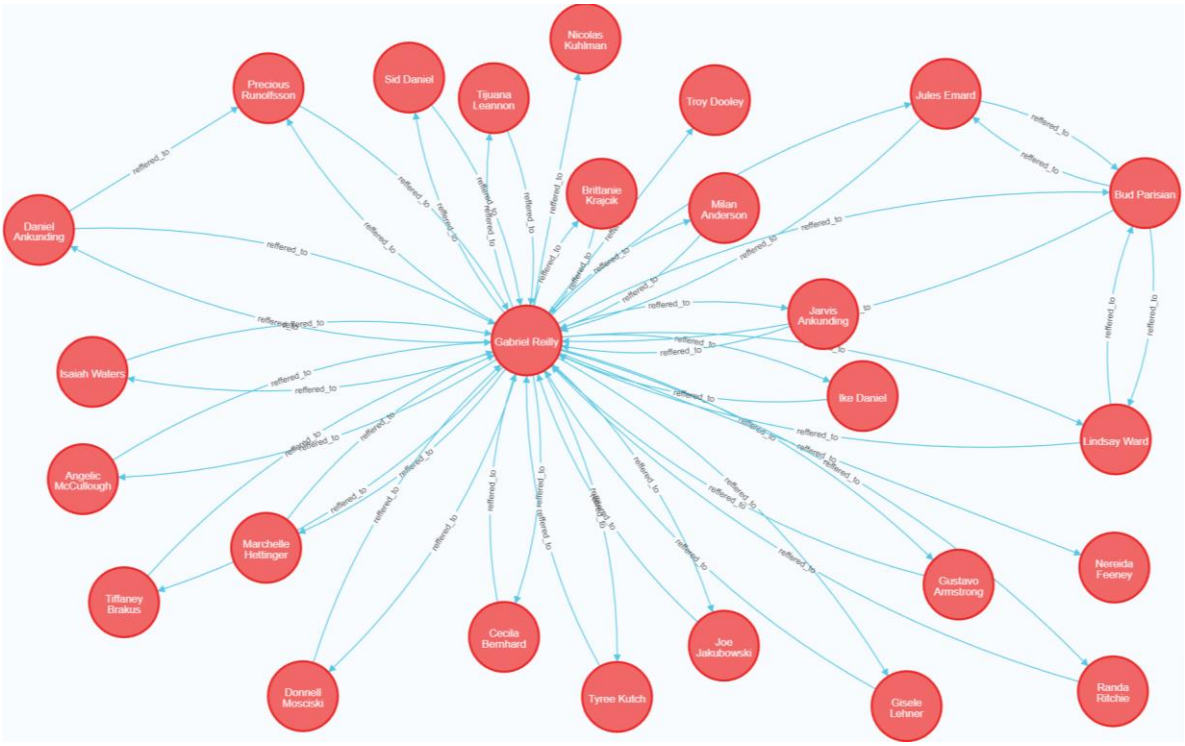

Figure 24: Snapshot of the same provider's network (zoomed in)

In summary, both questions reveal information that traditional relationship methods struggle to capture. For example, in the first question, we can see **groups** formed by conditions (after applying the clustering algorithm), if the label domain is too large to identify visually, adding a property to each entity can specify its group. Same for the second query, even for **~300 providers** on **figure 23** it is not that easy to spot providers with a big volume of incoming or outcoming relationships, imagine having millions of these nodes, thus we could apply the same approach and add the betweenness score into the nodes and query by descending order the most important nodes with the highest scores.

	provider	score
1	"Gabriel Reilly"	1272233.81822999
2	"Courtney Kihn"	29378.421248573144
3	"Bud Parisian"	25932.835528117175
4	"Cortez Price"	24256.21033923793
5	"Ana Luisa Gallardo"	17282.83680187145
6	"Terrie Ratke"	16852.69970083501

Figure 25: Betweenness score for top 5 "important" nodes

5. CONCLUSION & DISCUSSION

Real-world data has become increasingly prominent throughout the pharmaceutical sector. However, privacy policies and regulatory requirements present significant challenges for researchers aiming to apply innovative techniques, particularly artificial intelligence. Additionally, the substantial costs associated with acquiring real-world datasets can amount to millions of dollars each year. Therefore, synthetic data is created to mimic real-world data while remaining entirely artificial. Synthetic data are free of cost and privacy restrictions, enabling cross collaboration and experimentation across industry and academia. In this paper, we presented and utilized the Synthea tool for generating synthetic patient data that looks very similar to real-world data. Synthea generates data using modular disease-progression state simulators by using public epidemiology statistics, yielding synthetic populations where distribution (prevalences, incidence rates, etc.) approximate real-world patterns. When it comes to storing and analyzing real-world data, relational methods remain the primary technology, but challenges do still exist, the rigorous form from storing data as tables introduce difficulties such as observing hidden information (hidden clusters, long path interactions etc.), performance limitations, and difficulties in interpreting data semantics. Having all this, in this paper we introduced graph theory and graph databases from both a theoretical and practical perspective. We started by covering step-by-step modeling, performance considerations, and graph data science approaches to uncover clusters and key nodes. For the step-by-step modeling, we showcased how each graph "schema" is being built by following an iterative evolving approach. Graphs offer both strengths and limitations due to their schema-less structure and inherent flexibility; while they allow for straightforward modeling of datasets tailored to specific research questions, this same flexibility can also lead to the introduction of inconsistencies within the graph. Although performance was not the primary focus of our study, we formulated specific research questions and utilized a dataset comprising approximately 80,000 patients to evaluate and compare the effectiveness of both databases on a larger scale. We found that ingestion in GDB was **six times slower** than in RDB, including data loading, indexing, and preprocessing. Although this is a significant difference, it is expected and not critical because graphs require extra overhead to create edges. In most cases, the ingestion task happens periodically (maybe every 3,6,9,12 months). Conversely, the query execution time following data ingestion was substantially lower—**over four times shorter**—in the GDB than in the RDB. Several studies have further indicated that this disparity tends to widen as the volume of data increases. This finding is quite important, since writing queries to retrieve information is a significant part of a statistical programmer's job. Even with approximately 80,000 patients, some queries require several seconds to minutes to be executed. Considering real-world data (RWD) databases often contain millions of patients and billions of records, query performance becomes an even more significant challenge on a scale. In addition, real-life projects may require multiple queries, which can save a lot of time when executed ~x4 times faster. Finally, the last task was focused on showcasing graph data science algorithms, to extract "hidden" information. Real-world data often contains complex relationships that are not immediately apparent. Graph data science helps uncover these by analyzing paths, clusters, key nodes, and node similarities. This paper introduces a method for detecting hidden clusters and a centrality algorithm to identify providers who may delay patient treatment. In conclusion, while graph-based methods may not yet be as mature as relational approaches, they represent a promising avenue for extracting and analyzing patient information and hold significant potential **to enhance patient outcomes**.

REFERENCES

[1] G. Panagiotakis and N. Kountouris, "Introducing graph structure in real-world data analyses." Poster presented at the PHUSE EU Connect 2024, Strasbourg, France, November 2024.

[2] Office of the National Coordinator for Health Information Technology, "National Trends in Hospital and Physician Adoption of Electronic Health Records (Health IT Quick-Stat)." Health IT - Quick Stat, 2022. Accessed 6 Aug 2025.

[3] J. M. Mendes, A. Barbar, and M. Refaie, "Synthetic data generation: a privacy preserving approach to accelerate rare disease research," Frontiers in Digital Health, vol. 7, p. 1563991, Mar. 2025. Published March 18, 2025.

[4] U.S. Food and Drug Administration, "Real-world evidence." Science and Research – Special Topics, June 2025. Accessed 6 Aug 2025.

[5] U.S. Food and Drug Administration, "Center for biologics evaluation and research center for drug evaluation and research real-world evidence." Science Research – Real-World Evidence, FDA, June 2025. Content current as of June 9, 2025; accessed August 22, 2025.

[6] U.S. Food and Drug Administration, "Considerations for the use of real-world data and real-world evidence to support regulatory decision-making for drug and biological products." FDA Guidance for Industry, Aug. 2023. Final guidance issued August 2023; accessed August 22, 2025.

[7] A. Dang, “Real-world evidence: A primer,” Pharmaceutical Medicine, vol. 37, pp. 25–36, Jan. 2023. Epub January 5, 2023; published 2023.

[8] Global Alliance for Genomics and Health, “GDPR brief: when are synthetic health data personal data?” GA4GH News Events — GDPR Brief, April 2024. accessed 6 Aug 2025.

[9] The MITRE Corporation, “Synthea™: Synthetic patient generator.” https:// syntheticheal th.github.io/synthea/, 2017. Accessed 6 Aug 2025.

[10] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: Synthetic minority over-sampling technique,” Journal of Artificial Intelligence Research, vol. 16, pp. 321– 357, June 2002. Published June 2002. 18

[11] I. Diouf, J. Grimes, M. J. O’Brien, H. Hassanzadeh, D. Truran, H. Ngo, P. Raniga, M. Lawley, D. C. Bauer, D. Hansen, S. Khanna, and R. Reguant, “An approach for generating realistic Australian synthetic healthcare data,” Studies in Health Technology and Informatics, vol. 310, pp. 820–824, Jan. 2024. Published January 25, 2024.

[12] A. Gonzales, G. Guruswamy, and S. R. Smith, “Synthetic data in health care: A narrative review,” PLOS Digital Health, vol. 2, p. e0000082, Jan. 2023. Published January 6, 2023.

[13] J. Chen, D. Chun, M. Patel, E. Chiang, and J. James, “The validity of synthetic clinical data: a validation study of a leading synthetic data generator (synthea) using clinical quality measures,” BMC Medical Informatics and Decision Making, vol. 19, p. 44, Mar. 2019. Published March 14, 2019.

[14] K. Bhanot, M. Qi, J. S. Erickson, I. Guyon, and K. P. Bennett, “The problem of fairness in synthetic healthcare data,” Entropy, vol. 23, p. 1165, Sept. 2021. Published September 4, 2021.

[15] M. Nisevic, D. Milojevic, and D. Spajic, “Synthetic data in medicine: Legal and ethical considerations for patient profiling,” Computational and Structural Biotechnology Journal, vol. 28, pp. 190–198, 2025.

[16] The MITRE Corporation, “Synthea synthetic data overview.” FHIR® for Research documentation, 2023. Accessed 6 Aug 2025.

[17] J. Schrodt, A. Dudchenko, P. Knaup-Gregori, and M. Ganzinger, “Graph-representation of patient data: a systematic literature review,” Journal of Medical Systems, vol. 44, p. 86, Mar. 2020. Published March 12, 2020.

[18] J. Roemer, “Graphs in healthcare: Improving patient outcomes with graph algorithms.” Neo4j Blog, June 2020. Published June 24, 2020; accessed 22 Aug 2025.

[19] Wikipedia contributors, “Graph (discrete mathematics),” 2025.

[20] G. Belani, “Using graph databases to model patient journeys and clinical relationships.” Datafloq blog, July 2025. Published July 1, 2025; accessed 22 Aug 2025.

[21] PuppyGraph Blog, “What are graph query languages?.” PuppyGraph Blog, May 2025. Published May 9, 2025; accessed 22 Aug 2025.

[22] Wikipedia contributors, “Graph query language.” Wikipedia, The Free Encyclopedia, Aug. 2025. Last edited 14 Aug 2025; accessed 22 Aug 2025.

[23] J. Reed, “How knowledge graphs help pharma companies with data visualization.” IQVIA Blog, June 2023. Published June 19, 2023; accessed August 22, 2025.

[24] Memgraph, “Data modeling: Introduction to graph data modeling,” 2025. Product documentation.

[25] Neo4j, Inc., Neo4j Documentation, 2025.

[26] J. Newberry, “Populating a snomed ct property graph with synthetic patient data.” Medium blog, Mar. 2022. Published March 31, 2022; accessed August 24, 2025.

[27] J. A. M. Stothers and A. Nguyen, “Can neo4j replace postgresql in healthcare?,” AMIA Joint Summits on Translational Science Proceedings, vol. 2020, pp. 646–653, may 2020. eCollection 2020. ©2020 AMIA – All rights reserved.

[28] T. Sikhinam, “How does the performance of a graph database such as Neo4j compare to the performance of a relational database such as Postgres?,” Course project report p06, University of Washington, Seattle, WA, 2020. CSE D516, Autumn 2020

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Author Name: Panagiotakis Georgios
Email: (Georgios.Panagiotakis@pfizer.com)

Co-Author Name: Kountouris Nikolaos
Email: (Nikolaos.Kountouris@pfizer.com)

Co-Author Name: Puri Maria
Email: (Maria.Puri@pfizer.com)