PHUSE APAC 2026 – Paper Submission

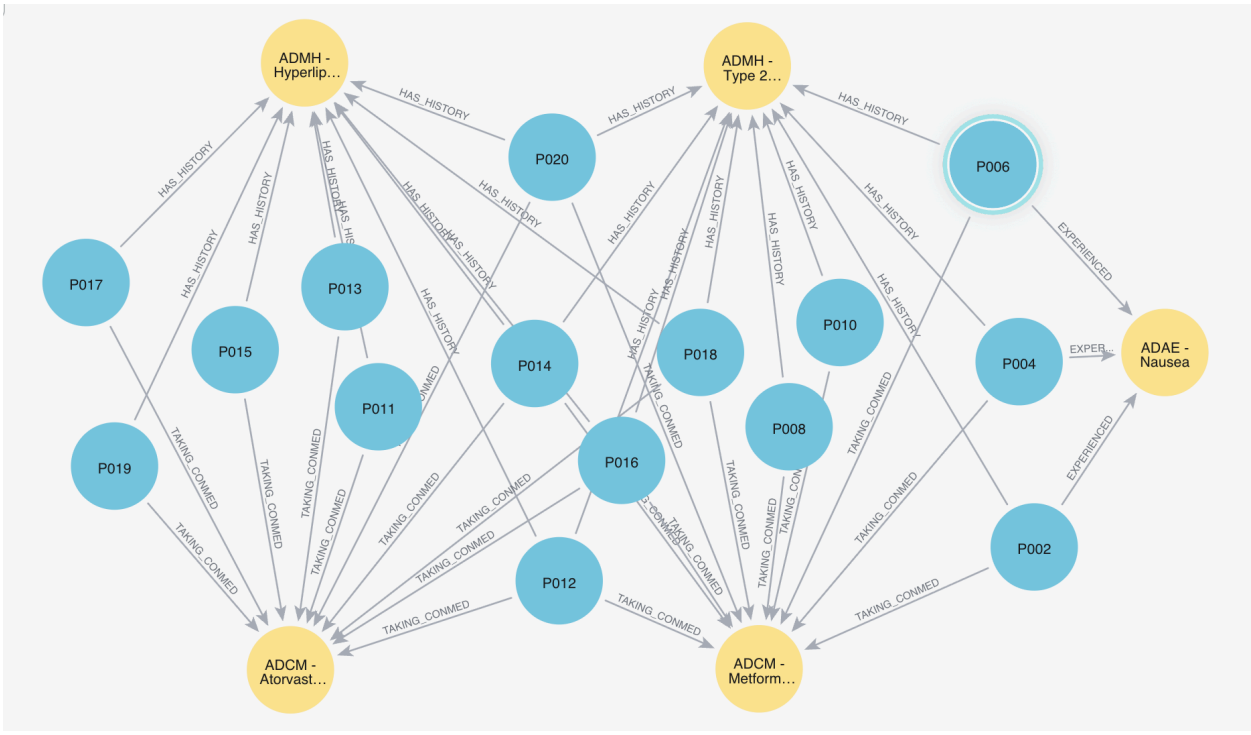
Presentation Number: ET15

Presentation Title: Unveiling Powerful Patient Data Insights Through Graph Database

Authors: Sonal Dedhia, Naresh Akhade, Saama

Abstract:

By unifying fragmented ADaM domains and unstructured trial protocols into a knowledge graph, organizations can move beyond simple keyword search to multi-hop clinical reasoning. This presentation leverages **Neo4j Graph Data Science, GraphRAG** and **AI agents** to automate complex tasks like patient-to-trial matching and cross-domain safety signal detection with 100% data traceability. Using **Neo4j Bloom**, clinical teams gain a transparent, visual "window" into the data, ensuring that high-stakes medical insights are both explainable and grounded in verified study data.



1. Graph Data Science:

The Strategic Value of Graph Data Science in Clinical Trials

- **Cross-Domain Intelligence:** Instantly connect ADaM domains (ADSL, ADAE, ADCM, etc.) without complex SQL joins, enabling 360-degree patient safety views.
- **Predictive Safety Signaling:** Use Link Prediction to identify potential drug-drug interactions before they impact a larger cohort.
- **Operational Efficiency:** Reduce trial recruitment time by using Node Similarity to find "ideal" patients across historical EHR data.
- **Standardization & Compliance:** Native support for CDISC standards ensures that graph insights are traceable and ready for regulatory submission.

Example 1:

To demonstrate the power of Neo4j GDS in a clinical context, let's look at a **Patient Similarity** example. This query identifies patients who are "mathematically similar" based on their shared medical conditions and biomarkers.

The Scenario

We want to find a cohort of patients similar to a specific "Candidate Patient" to see if they might be eligible for the same clinical trial.

The Cypher Code

This example uses the **Node Similarity** algorithm. It assumes a graph where (:Patient)-[:HAS_CONDITION]->(:Condition).

Cypher

```
None
// Step 1: Project the graph into memory
CALL gds.graph.project(
  'clinicalTrialGraph',
  ['Patient', 'Condition'],
  'HAS_CONDITION'
);
```

```
// Step 2: Run Similarity and Stream results
CALL gds.nodeSimilarity.stream('clinicalTrialGraph')
YIELD node1, node2, similarity
RETURN
  gds.util.asNode(node1).patientId AS PatientA,
  gds.util.asNode(node2).patientId AS PatientB,
  similarity
ORDER BY similarity DESC
LIMIT 10;
```

What this code does for the Trial

- **The Projection:** It creates a temporary, high-speed map of which patients have which conditions.
- **The Similarity Logic:** It calculates a **Jaccard Score** (0.0 to 1.0). If Patient A and Patient B share 9 out of 10 conditions, they receive a high similarity score.
- **The Clinical Value:** Instead of manually filtering by 50 different criteria, the algorithm automatically surfaces the most "relevant" patients for recruitment based on their entire medical profile.

Visualizing the Result

The output is a **weighted network**.

Patient A (Target)	**Patient B (Match)**	**Similarity Score**	**Shared Indicators**
P-102	P-554	**0.92**	Type 2 Diabetes, Hypertension, BMI > 30
P-102	P-881	**0.85**	Type 2 Diabetes, Sleep Apnea

P-102	P-203	**0.78**	Hypertension, High Cholesterol

Example 2: Predictive Adverse Event Modeling

The Goal: Predict potential side effects or drug-drug interactions *before* they occur in a clinical trial.

In a graph, we connect **Drugs**, **Targets** (Proteins/Genes), and **Side Effects**. If Drug A and Drug B both hit similar protein targets, GDS can predict a "hidden" link between them that indicates a high risk of a specific adverse event.

The Cypher Code: Pipeline Training

This example shows how GDS builds a machine learning model to predict these missing links.

Cypher

```
None
// 1. Create a specialized ML pipeline
CALL gds.alpha.ml.linkPrediction.create('adverseEventPipeline');

// 2. Add features (how the model "looks" at the nodes)
// Here, we use Common Neighbors to see if drugs share targets
CALL gds.alpha.ml.linkPrediction.addFeature('adverseEventPipeline',
'commonNeighbors', {
  nodeProperties: []
});

// 3. Train the model to predict the "INTERACTS_WITH" relationship
CALL gds.alpha.ml.linkPrediction.train('clinicalGraph', {
  pipeline: 'adverseEventPipeline',
  targetRelationshipType: 'INTERACTS_WITH',
  modelName: 'drugInteractionModel'
});
```

- **Safety:** The AI flags high-risk patients who are taking medications that have never been formally tested together.
- **Data Enrichment:** It fills in the gaps of known medical knowledge using the existing structure of the graph.

2. Graph RAG

In the context of clinical trials, Neo4j GraphRAG acts as a "connective tissue" across the massive, fragmented landscape of medical data. For clinical trials, where accuracy is non-negotiable, GraphRAG offers several distinct advantages.

In a typical clinical environment, data is siloed in PDFs (study protocols), tabular databases (patient records), and hierarchical standards (CDISC, MedDRA, SNOMED). Traditional RAG treats these as isolated text chunks, but Neo4j GraphRAG treats them as a **unified knowledge network**.

1. Multi-Hop Reasoning for Complex Queries

Traditional AI search (Vector RAG) finds text that "looks" like the question. GraphRAG, however, can traverse multiple relationships to answer complex medical questions.

- **The Advantage:** It can answer queries like: *"Which patients on Study X with a history of hypertension experienced a Grade 3 AE after their third dose?"* * **How it works:** Instead of searching for the words "hypertension" and "Grade 3," GraphRAG follows the physical links in Neo4j from the Patient to their MedicalHistory, through the Dosing schedule, and finally to the AdverseEvent node.

2. Global Context and Summarization

- **The Advantage:** GraphRAG can summarize themes across an entire trial. It can answer: *"What are the top three safety themes observed across all sites in the EU?"*
- **Hierarchical Clustering:** Neo4j allows for "Community Detection" (e.g., Leidien or Louvain algorithms). GraphRAG uses these pre-computed clusters to provide a holistic view of the data that individual text snippets cannot capture.

3. Grounding and Hallucination Reduction

One of the biggest risks in using LLMs for clinical data is "hallucination" (making up facts).

- **The Advantage:** GraphRAG "grounds" the LLM in a structured Knowledge Graph. When the AI generates an answer, it is forced to use the nodes and relationships present in your **CDISC-compliant graph**.

- **Deterministic Retrieval:** Because the retrieval is based on actual graph paths, the facts fed to the AI are verifiable and exist in the source data, significantly increasing the reliability of the output for regulatory submissions.

4. Superior Explainability and Traceability

Being able to "show your work" to a regulator is vital.

- **The Advantage:** GraphRAG provides a clear **provenance trail**.
- **Visual Evidence:** Unlike "black box" vector searches, GraphRAG can return the exact sub-graph (the specific nodes and edges) used to generate an answer. You can present a visual map to an auditor showing exactly which patient records and which lab results led the AI to a specific conclusion.

5. Merging Structured and Unstructured Data

Clinical trials involve both structured tables (SDTM/ADaM) and unstructured text (Clinical Study Reports, Investigator Brochures).

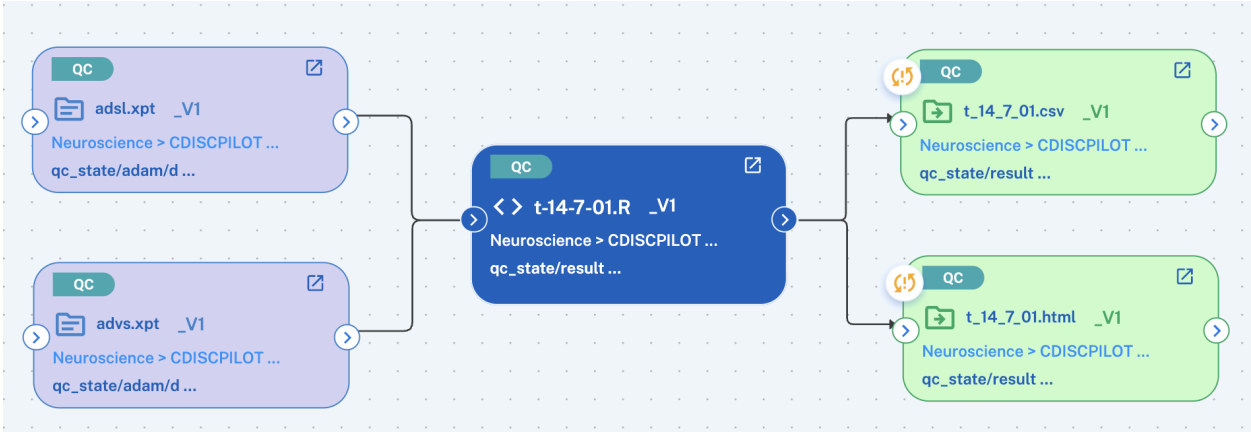
- **The Advantage:** GraphRAG is the "bridge." It allows you to store the structured CDISC data as nodes and link them directly to embeddings of the unstructured text.
- **Unified Intelligence:** An LLM can now query the graph to find a specific lab value (structured) and then instantly pull the related investigator's comment (unstructured) from a PDF report because they are linked by a DESCRIBED_IN relationship.

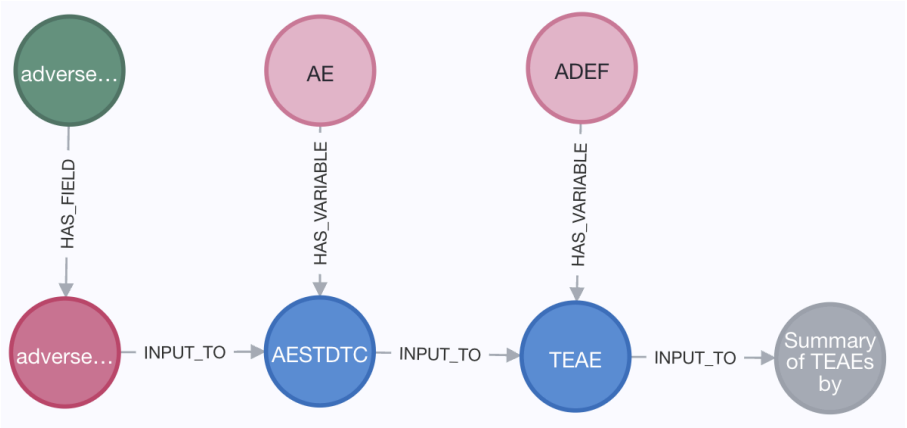
3. Bloom

1. Using the "Cypher Search" Bar

You can paste the query directly into the search bar at the top of Bloom.

- **How it works:** Bloom will execute the query and automatically zoom in on the specific nodes (Subject) and relationships (EXPERIENCED) returned.





2. Creating a "Perspective" (Saved Search)

This is the most powerful way to use GraphRAG reasoning in Bloom for non-technical clinical users (like Medical Monitors or Safety Reviewers).

You can create a **Search Template** where the parameters are dynamic. For example, instead of hardcoding "Nausea," you can use a placeholder.

3. Visualizing ADaM Patterns in Bloom

In a clinical context, Bloom is often used to find "Patient Journeys." Because ADaM domains like **ADAE** and **ADCM** are time-bound, you can use Bloom's **Timeline** feature alongside your Cypher query.

Why this matters for GraphRAG:

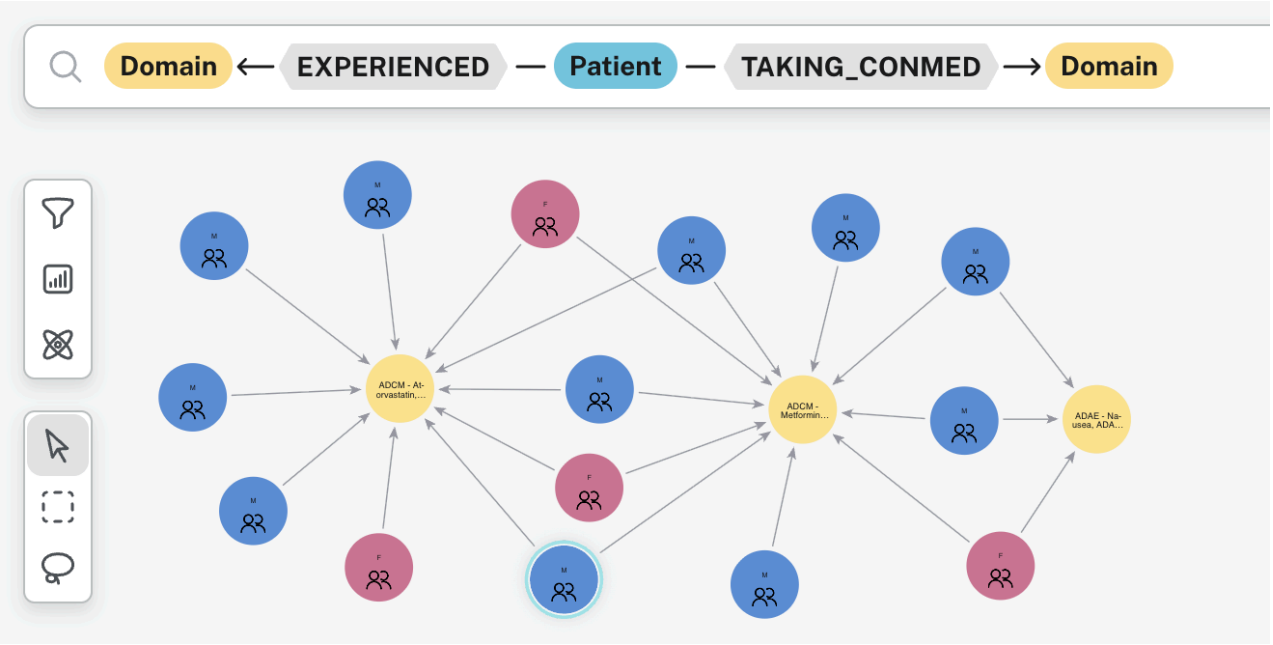
When your RAG system identifies a trend (e.g., *"There is a correlation between Drug X and Nausea"*), a clinical lead can open Bloom to:

1. **Verify the Path:** Visually see the connection between the ADSL Subject, the ADAE event, and the ADCM (concomitant meds) they were taking at that exact time.
2. **Slicing Data:** Use the "Filter" pane to instantly hide "Mild" events and focus only on "Severe" cases across the entire ADaM graph.

4. Setting it up for ADaM Domains

To make your ADaM data "Bloom-ready," ensure your graph has these three things:

1. **Category Colors:** Color-code ADSL (blue), ADAE (red), and ADLB (green) for instant recognition.
2. **Scene Actions:** Set up a deep link so that clicking a Subject node opens their full profile in your clinical data management system.
3. **Rule-based Styling:** Automatically make the AdverseEvent node larger or glow if the AESER (Serious Adverse Event) flag is 'Y'.



4. Conclusion

Graph transforms clinical trial data into a dynamic knowledge graph, offering unparalleled **traceability and performance** for complex CDISC standards. By prioritizing relationships, it enables **real-time safety signal detection** and seamless integration across disparate data silos. When coupled with **GraphRAG**, it provides a grounded, explainable AI framework that eliminates hallucinations and supports deep, multi-hop medical reasoning. Ultimately, adopting a graph-first approach empowers pharmaceutical researchers to move beyond static tables to **actionable, interconnected insights** for faster regulatory submissions.