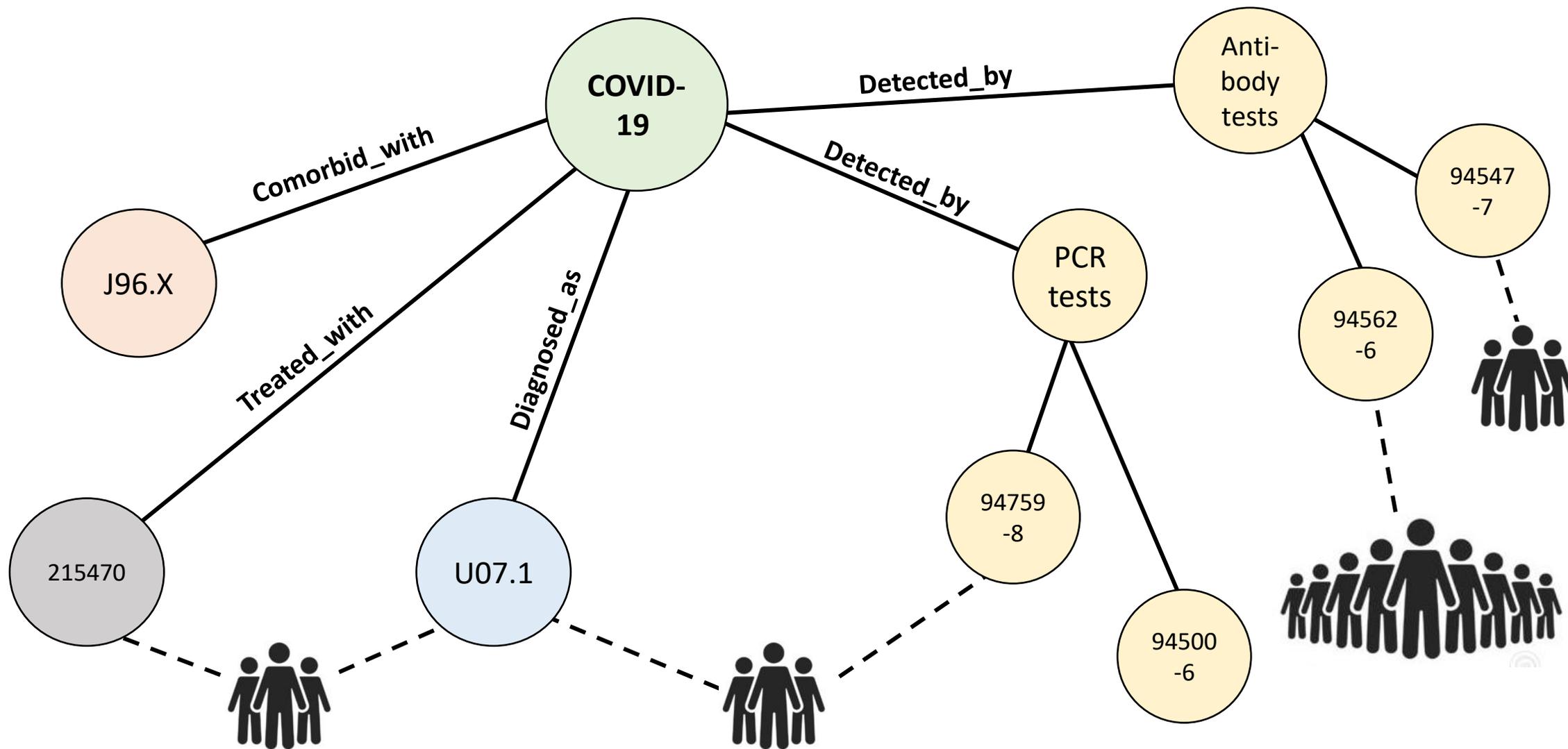


In an (ontologically) perfect world...



# Reality Check

- In our non-ontologically-perfect world, a COVID-19 computable phenotype looks more like [this](#).
- This covers all the concepts, but:
  - Must be translated into a number of different data models and dialects
  - Must be continually manually updated to stay current
- What would it take to be able to write *one* query script that:
  - Thoroughly covers all the concepts with the fewest possible codes?
  - Can be run at multiple institutions with few (or no) local changes?
  - Results in a consistently, accurately defined cohort?

# Reality Check

- What would it take to be able to write *one* query script that:
  - Thoroughly covers all the concepts with the fewest possible codes?
  - Can be run at multiple institutions with few (or no) local changes?
  - Results in a consistently, accurately defined cohort?

clinical ontologies

interoperable data  
models/technologies

clinical ontologies



# Building Blocks

---

Literature, Current Practices, Existing  
Technologies



# Computable Phenotyping

- *Computable phenotypes* are algorithms used to query EHR data to identify patient cohorts with conditions or events relevant to some use case.
- Important for research, but difficult to share across institutions in a machine-readable way.
- Rely heavily on standard codesets (ICD-10, LOINC, RxNorm, CPT, etc.)
- The ideal computable phenotype is shareable, data model agnostic, semantically unambiguous, machine readable, and publicly accessible.

# The advantage presented by ontologies



Current state: exact code matches,  
from all over the hierarchy

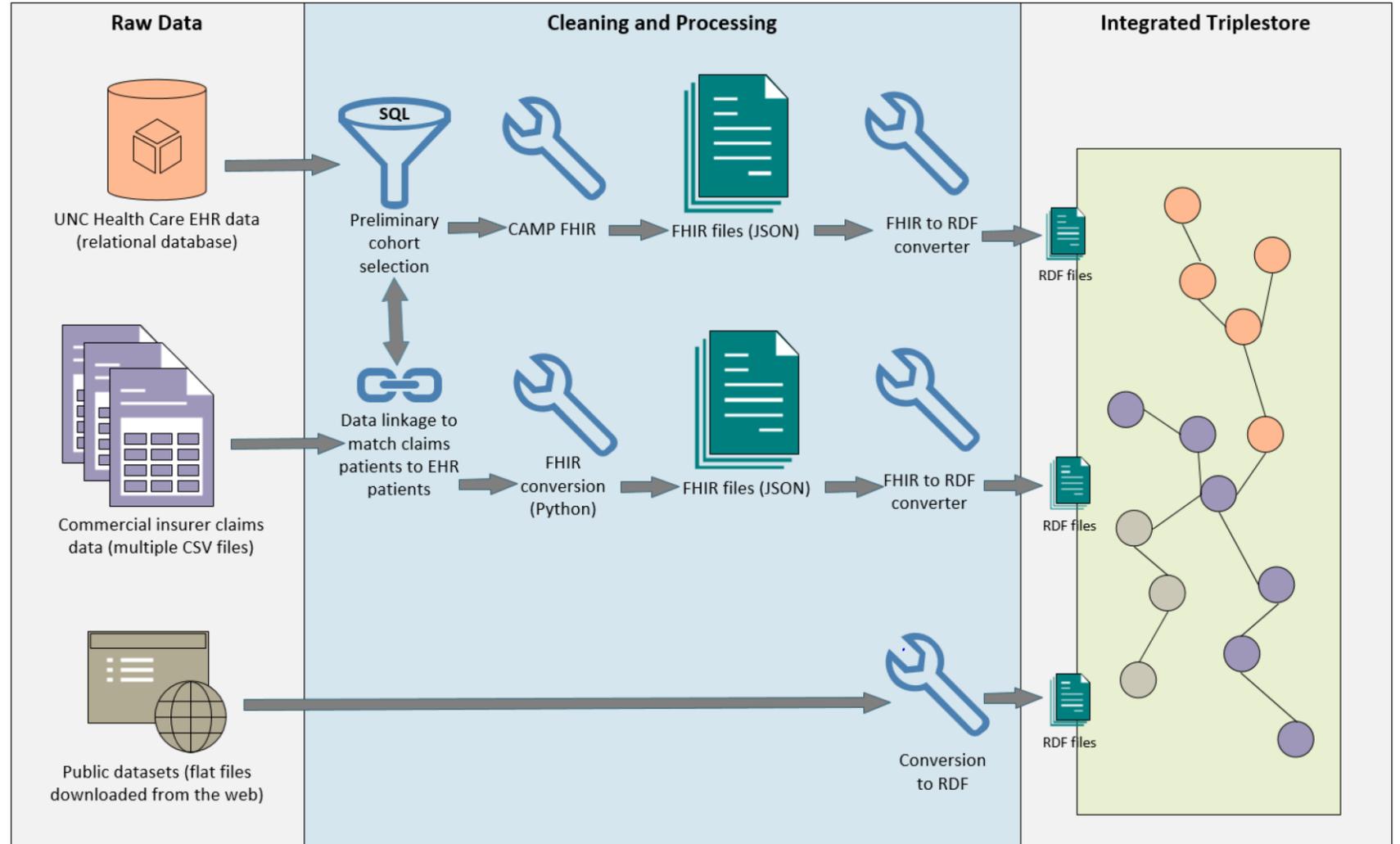


Desired state: conceptual  
matches, powered by ontologies

# The Opioid Triplestore

Existing clinical dataset from a prior project\*; a triplestore containing EHR, claims, and community-level data for post-surgical patients who were prescribed opioids.

\*Collaborative project; see Acknowledgements slide for list of contributors and funding information





# Comparing Phenotype Performance: Ontology versus ICD-10-CM codeset

- Consider two computable phenotypes: (1) depression, and (2) rheumatoid arthritis.
- Objective is to compare the “coverage” of a computable phenotype defined by a list of ICD-10-CM codes (from PheKB) against a phenotype defined by a single concept (along with its child concepts) from the SNOMED or HPO ontology.
- Measuring the sensitivity and specificity of the phenotype methods is not possible in this study, as there is no way to know who the “true” cases are—we can only know which patients are identified by each phenotyping method.
- What is critical to measure, then, is the degree to which the two phenotyping methods overlap, as well as the degree to which they diverge.

# Mapping SNOMED to ICD-10-CM

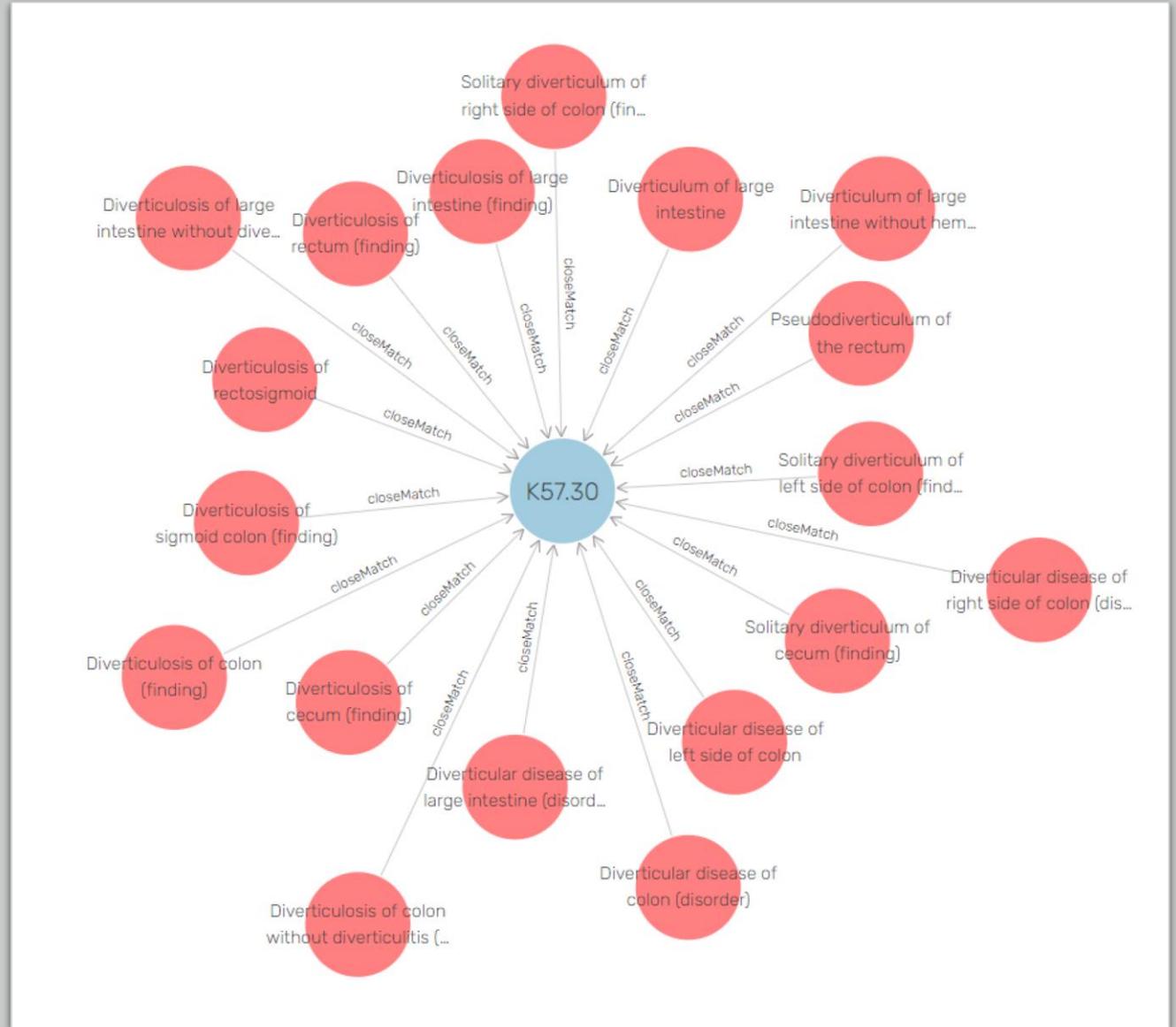
SNOMED Code	SNOMED description	Mapping Rule	Mapping Advice	ICD-10-CM code	ICD-10-CM descr.
100801000119107	Maternal tobacco use in pregnancy	IFA 10761391000119102   Tobacco use in mother complicating childbirth	IF TOBACCO USE IN MOTHER COMPLICATING CHILDBIRTH CHOOSE 099.334	099.334	Smoking (tobacco) complicating childbirth
100801000119107	Maternal tobacco use in pregnancy	OTHERWISE TRUE	ALWAYS 099.330	099.330	Smoking (tobacco) complicating pregnancy, unspecified trimester

Context-dependent mapping

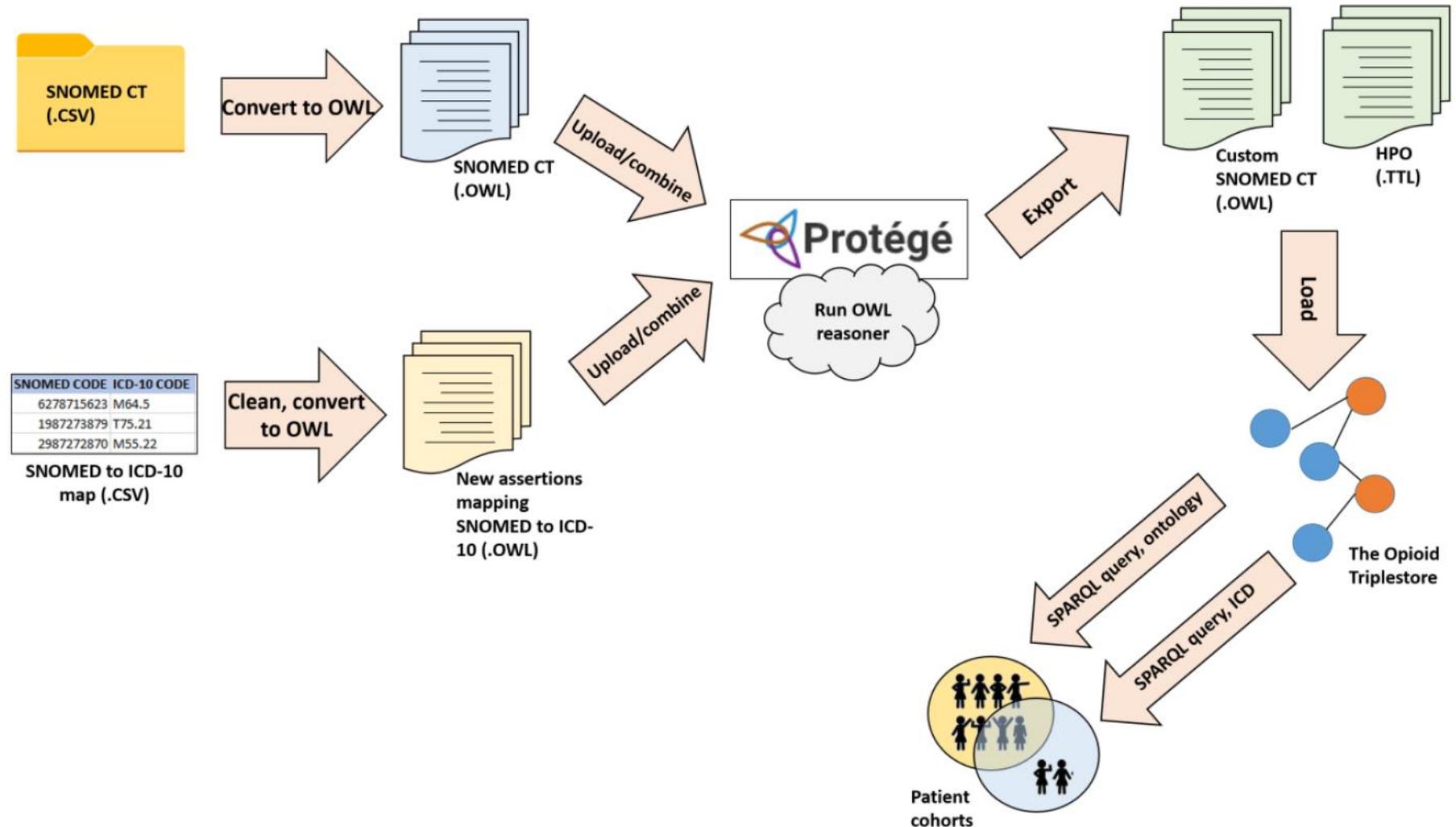
SNOMED CT Code	SNOMED CT description	Mapping Advice	ICD-10-CM code	ICD-10-CM description
10298002	Victim, motorcycle rider in vehicular AND/OR traffic accident	ALWAYS V29.9XX?   EPISODE OF CARE INFORMATION NEEDED   THIS IS AN EXTERNAL CAUSE CODE FOR USE IN A SECONDARY POSITION	V29.9XX?	Motorcycle rider (driver) (passenger) injured in unspecified traffic accident, episode of care unspecified

Required seventh digit mapping

# Mapping SNOMED to ICD-10-CM



# Transformation Pipeline



# Implementing the Pipeline in KNIME

- A. Documentation
- B. CSV to RDF/OWL converter
- C. FHIR to RDF converter
- D. Triplestore loader
- E. Data mover (optional)

**0: TriplestoreCreator\_public** Welcome to KNIME Analytics Platform

**Prerequisites for Running this Pipeline:**

1. Server
  - \* OS: Linux (Pipeline built using RHEL 7)
  - \* Minimum 8 GB RAM; 16 GB is better
  - If working on a remote server (i.e., an environment external to where this pipeline lives):
  - \* If 2FA is enabled, it must be set to "auto-push" to your device if you want to run Step 2 (which is optional).
  - \* You must be able to SSH to the machine.
2. Software dependencies
  - \* Python 3
  - \* Blazegraph (<https://github.com/blazegraph/database>).
  - Note:** Blazegraph must NOT be running when you are trying to load data.
  - \* FHIR-to-RDF conversion utility (<https://github.com/BD2KOhFHIR/fhir2rdf>)
3. Files
  - \* Clinical data in FHIR format (file(s))
  - \* Optional: Other data in CSV format (file(s))
  - \* Optional: Other data in JSON format (file(s))
  - \* Optional: Other data in RDF format (file(s))

**The Basic List:**

1. Gather the data you wish to load, in CSV, RDF, and FHIR/JSON format, in a single directory. (Or, a single local directory and a single remote directory, if you can't keep sensitive data locally.) Subdirectories to keep things organized are fine. The important part is to know where everything is.
2. Convert any CSV files to RDF using the yellow box to the right.
3. Generate a shell script that you'll use to convert your FHIR/JSON files to RDF files using Step 1a (first green box below). Set aside.
4. Generate a shell script that you'll use to load all of your data, including the converted FHIR files, into your blazegraph instance using Step 1a. Set aside.
5. If working on a remote server, generate both shell scripts to the remote server. (If you're working on a local machine, you can run your shell scripts manually, but you should still generate them to the blazegraph directory.)
  - The script that loads Blazegraph files.
  - The FHIR conversion script.
6. Run Step 3 regardless of whether you're on a local machine or a remote server. The end result is a loaded triplestore.

**Optional : Convert CSVs to RDF (TTL) or OWL files for later triplestore ingest**

Python Edit Variable → Python Source (Transform CSV to TTL) → Python Source (Convert CSV to OWL)

Configure input CSV

First, edit the configuration for the CSV file and adjust the additional parameters. Then, run the TTL or OWL conversion in one of those formats.

You can execute this part of the pipeline conversion even if you don't want to run the rest of the pipeline. This step is independent.

**Step 1a:** Generate a shell script to convert your FHIR files to RDF.

Python Edit Variable → Python Source

Configure input FHIR (JSON) files → Generate shell script for FHIR to RDF conversion

**Step 1b:** Generate a shell script to load your RDF files into blazegraph.

Python Edit Variable → Python Source

Configure input RDF files → Generate shell script for loading into blazegraph

**Step 2 :** Upload your shell script(s) from Steps 1(a) and 1(b) to the blazegraph directory on your remote server. (Alternatively, skip this step and manually upload your shell script(s) to this directory.)

The data you want to load needs to either be accessible from the remote server, or uploaded to the remote server. You can use this same step to upload your data to the remote server, if desired—just switch in your filenames accordingly.

If your remote server uses multi-factor auth, make sure it is set to push a request to your phone on login attempt, rather than waiting for manual input from you.

SSH Connection → Upload

Configure your SSH connex to your remote server

List Files → Upload

No changes needed

Enter the blazegraph directory on your remote server to upload your shell script(s)

```
cd (blazegraph directory)
sed -i -e 's/\r$/' (your shell script)
chmod u+x (your shell script)
./(your shell script)
```

(The sed step gets rid of characters that KNIME uses in the shell scripts it outputs that will cause your shell script not to execute. The chmod is necessary if you have permissions issues once you upload your shell script to your remote server.)

At this point, depending on what script you're running, either blazegraph or blazegraph will be able to load your data. If you're using blazegraph, you may need to wait a significant amount of time.

Pipeline available for download at <https://github.com/empfff/clinical-tripleizer>

# Experimental Steps (Each Phenotype)

1. Execute a SPARQL query in the Opioid Triplestore using the ICD-10-CM codeset from the PheKB phenotype to identify matching patients.
2. Execute a SPARQL query in the Opioid Triplestore using a single ontology concept (SNOMED or HPO\*), as well as any child nodes, to identify matching patients.

```
?snomednode skos:closeMatch ?icdnode .  
?snomedsubnode rdfs:subClassOf* ?snomednode .  
?snomedsubnode skos:closeMatch ?icdsubnode .
```

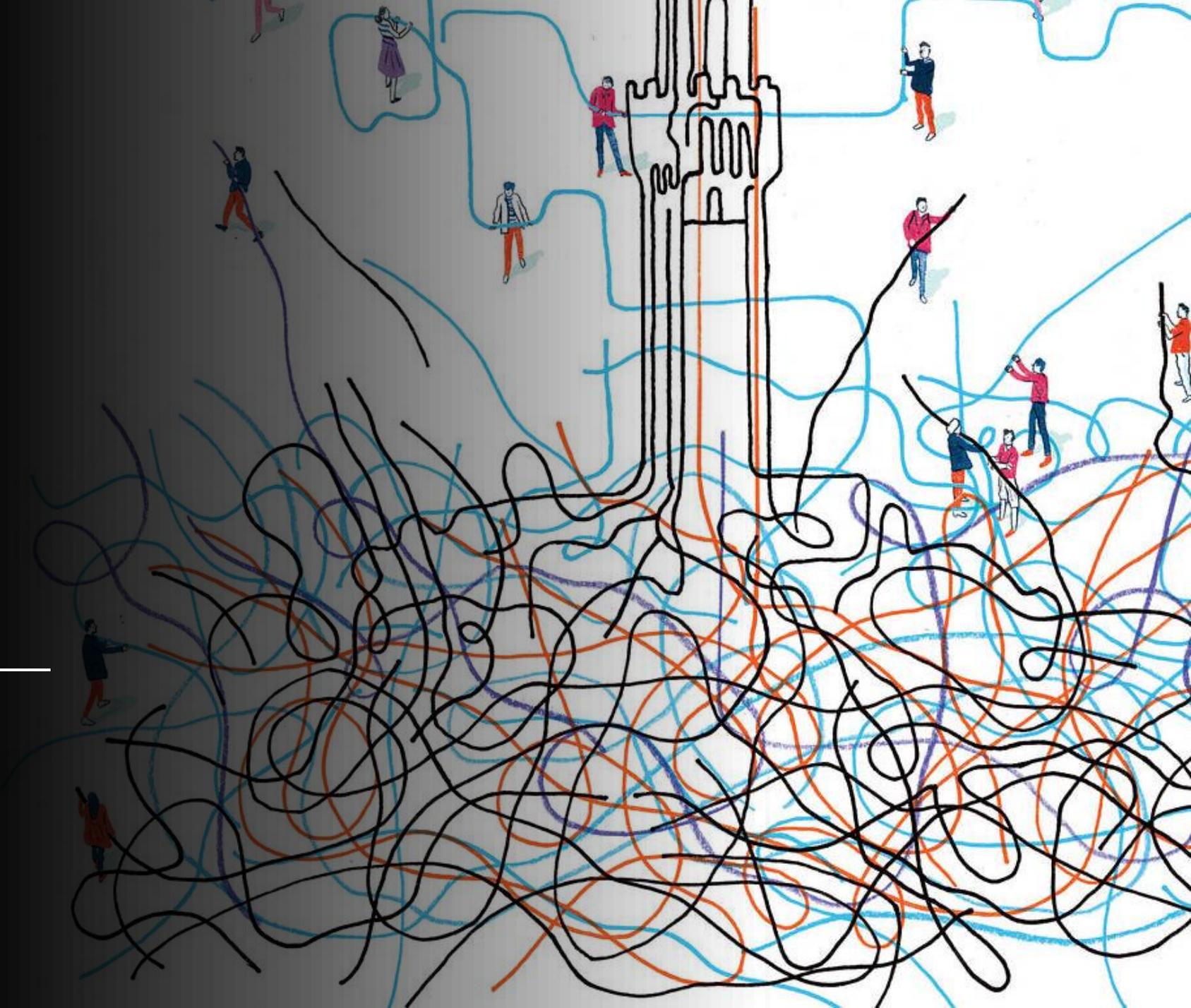
3. Compare the ICD-10-CM codes in the subgraphs defined by #1 and #2.
4. Compare the patients captured in the subgraphs defined by #1 and #2

\*Because HPO maps to SNOMED, though the queries differed, the results were the same.



# Results

---



# Depression

- In general, SNOMED/HPO is picks up more patients than the PheKB phenotype, with a broader definition of "depression."
- SNOMED/HPO codeset included all but one of the PheKB codes (F43.21, adjustment disorder), and added more.
- SNOMED/HPO includes several bipolar codes; PheKB phenotype specifically excludes them.
- SNOMED/HPO graph contains two "problem codes": T81.89X (complications from procedure) and B94.9 (sequelae of unspecified disease).

# Depression, continued

**Table 2.** Overlaps and differences between the cohorts defined by the PheKB and SNOMED/HPO depression queries against the Opioid Triplestore

Query	# patients found	# patients found by this query <i>only</i>	# patients found by both queries
PheKB	1,739	35	1,704
SNOMED/HPO	2,029	325	1,704

Of the "problem codes," B94.9 did not match with any actual patients in the Opioid triplestore. T81.89X? would have matched 47 more patients if the ? had been treated as a wildcard.



# SNOMED Correction (9/20 edition)

## Parents

-  Rheumatoid arthritis (disorder)
-  Seronegative arthritis (disorder)

 **Seronegative  
rheumatoid arthritis  
(disorder)**

SCTID: 239792003

239792003 | Seronegative rheumatoid  
arthritis (disorder) |

*en* Seronegative rheumatoid arthritis  
*en* Seronegative rheumatoid arthritis  
(disorder)



Pathological process →  
Autoimmune process  
Finding site → Joint structure  
Associated morphology →  
Rheumatic inflammation

Has interpretation → Negative  
Interprets → Rheumatoid factor  
measurement

# Rheumatoid Arthritis, continued

- SNOMED/HPO graph also contains two "problem codes": H20.10 (chronic iridocyclitis) and M25.80 (other specified joint disorders). Only the latter matched patients in the triplestore.

**Table 6.** Overlaps and differences between the cohorts defined by the PheKB and SNOMED/HPO queries

Query	# patients found	# patients found by this query <i>only</i>	# patients found by both queries
PheKB	166	30	136
SNOMED/HPO	142	6	136



# Discussion & Conclusion

---



# Ontologies as a Remedy for Semantic Ambiguity

**Research Question 1:** Can clinical ontologies (specifically SNOMED CT and the Human Phenotype Ontology) offer a less semantically ambiguous mechanism to perform computable phenotyping than reimbursement-focused terminologies (specifically ICD-10-CM)?

- ICD-10-CM serves an important purpose. As it is unlikely for its hierarchy to be modified, SNOMED/HPO provide a valuable augmentation, if not an alternative.
- The superset of "Google-able" or already known ICD-10-CM codes and those in the SNOMED/HPO subgraph may be a good starting point for researchers trying to choose the right codes for their phenotype.
- Using the superset fills in gaps in both SNOMED and ICD-10-CM.
- I would argue that allowing researchers to keep or pitch codes from a longer list is preferable to not providing the choice.

# Semantic Web as a Driver of Interoperability

**Research Question 2:** Does the use of semantic web technologies and standards (e.g., Resource Description Framework (RDF), triplestores, the Web Ontology Language) in tandem with ontologies improve the interoperability prospects of computable phenotypes and the underlying clinical data?

The infrastructure for this project addressed three interoperability challenges:

- **Challenge:** Inconsistent data models
- **Addressed By:** FHIR, RDF
  
- **Challenge:** Differing coding practices
- **Addressed By:** Ontologies
  
- **Challenge:** Integration with non-EHR data
- **Addressed By:** RDF

# Limitations

- ICD-10-CM presents a bit of a catch-22—if it is the only diagnosis code type that will ever be attached to a patient, any phenotype method will eventually need to connect to ICD-10-CM. ("You're only as good as your worst ICD-10-CM code.")
- Our two phenotypes performed very differently. If I ran two more, I suspect they would each have their own idiosyncracies. This is a common issue in phenotyping studies.

---

# Acknowledgments

---

I want to thank my collaborators and co-authors on the Opioid Triplestore project:

- Robert Bradford
- Marshall Clark
- Dr. Jim Balhoff
- Dr. John Preisser
- Rujin Wang
- Kellie Walters
- Dr. Matt Nielsen

The project was funded by a TraCS Institute pilot grant.

Thanks also to Dr. Ashok Krishnamurthy, my advisor and mentor.

Thank you!