Meaningful Datatypes: Ontologically-Sound Dependent Type Systems for Data Science

Riley Moher
21.04.2022
University of Toronto

The Role of Data Science

- Data Science is increasingly important and valuable
- Data science drives important decisions
- Good data science begins with an understanding of the data
- Understanding data is critical

What’s in a (Data) Type?

- Data is representative of real-world phenomena
- Data is represented using simple datatypes
- Datatypes do not typify the real world
- This gap left by datatypes is significant and unsolved

Outline

1. Why are datatypes problematic for data science?

2. What are my contributions to solving this problem?

3. What is the significance of these contributions?

4. What are the directions for future work?

Outline

1. **Why are datatypes problematic for data science?**
2. What are my contributions to solving this problem?
3. What is the significance of these contributions?
4. What are the directions for future work?

Outline

1. **Why are datatypes problematic for data science?**
   1. **Datatype Problem Classes**
   2. **The gaps in current work**
2. What are my contributions to solving this problem?
3. What is the significance of these contributions?
4. What are the next steps?

Outline

1. Why are datatypes problematic for data science?
   I. **Datatype Problem Classes**
   II. The gaps in current work

2. What are my contributions to solving this problem?

3. What is the significance of these contributions?

4. What are the directions for future work?

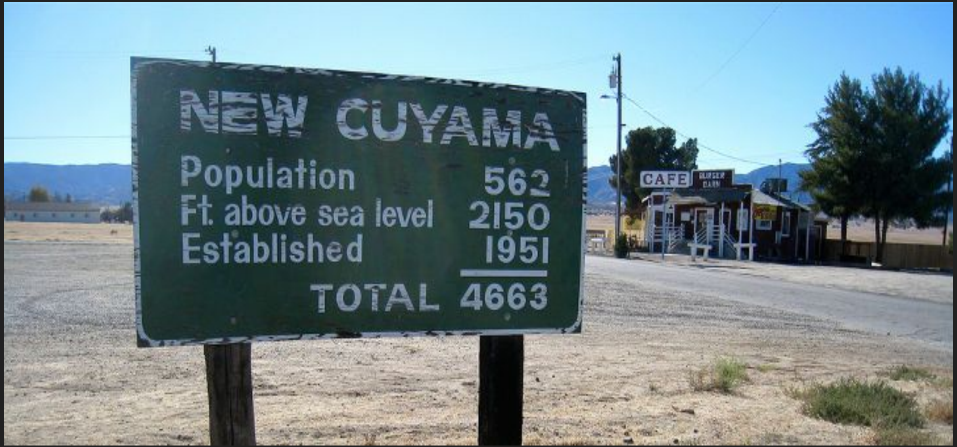
Datatype Problems

The Real World

- Implicit rules
- Complex concepts and relationships

Simple Datatypes

- Numbers are numbers*
- String OR Integer OR Float OR …
- Same datatype represents many different concepts



Datatype Problem Classes

1. Time
2. Mereology
3. Provenance

Datatype Problem Classes

1. **Time**
2. Mereology
3. Provenance

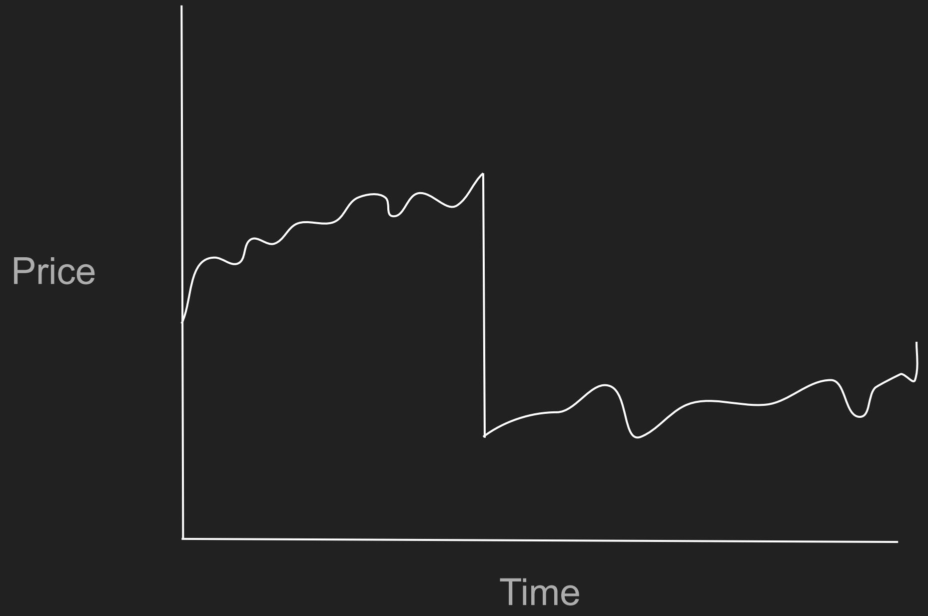
Datatype Issues: Time

- Time is common and important in most data

Datatype Issues: Time

- Time is common and important in most data

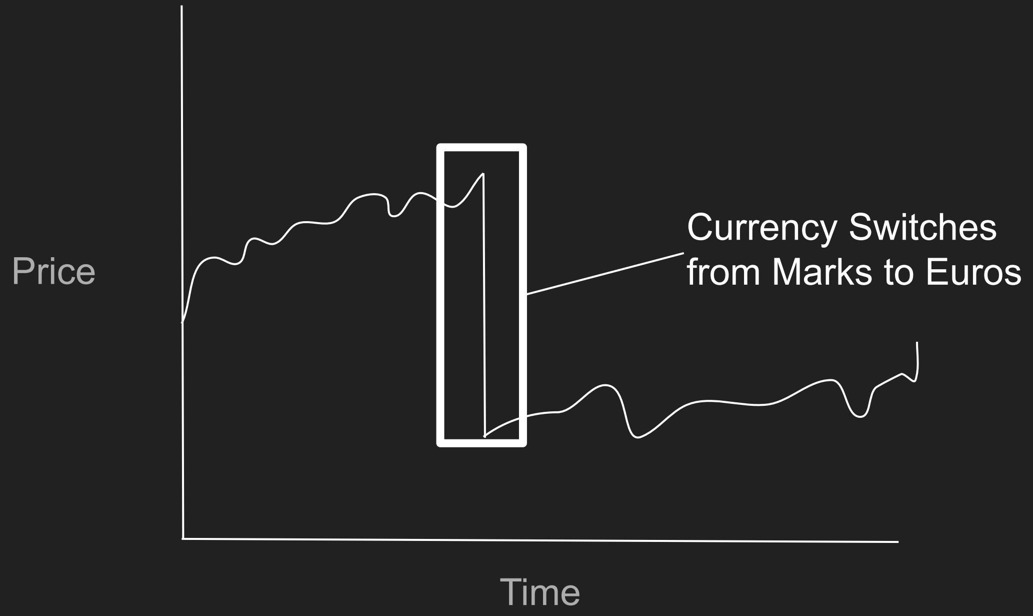
- Ex: Frankfurt Stock Exchange Quote



Datatype Issues: Time

- Time is common and important in most data

- Ex: Frankfurt Stock Exchange Quote



Datatype Problem Classes

1. Time
2. **Mereology**
3. Provenance

Datatype Issues: Mereology

- Mereology in data exists in many different forms

- Ex: COVID-19 Vaccination Data

Date	Fully Vaccinated	% of Eligible Population Fully Vaccinated
2021-06-22	1 200 000	3.5%
…	…	…
2021-07-29	1 335 000	2.6%
…	…	…
2022-05-28	1 086 100	2.2%

Datatype Issues: Mereology

- Mereology in data exists in many different forms

- Ex: COVID-19 Vaccination Data

Date	Fully Vaccinated	% of Eligible Population Fully Vaccinated
2021-06-22	1 200 000	3.5%
Children Become Part of Eligible Population		…
2021-07-29	1 335 000	2.6%
…	…	…
2022-05-28	1 086 100	2.2%

Datatype Issues: Mereology

- Mereology in data exists in many different forms

- Ex: COVID-19 Vaccination Data

Date	Fully Vaccinated	% of Eligible Population Fully Vaccinated
2021-06-22	1 200 000	3.5%
…	…	…
2021-07-29	1 335 000	2.6%
“Fully Vaccinated” definition changes to 3+ doses		…
2022-05-28	1 086 100	2.2%

Datatype Problem Classes

1. Time

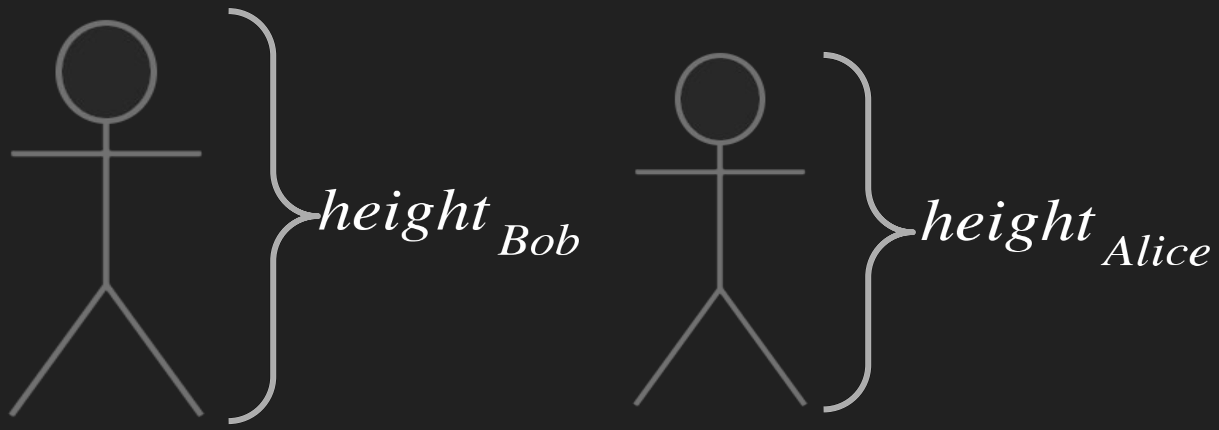
2. Mereology

3. **Provenance**

Datatype Issues: Provenance

- Data science operations transform real-world interpretations

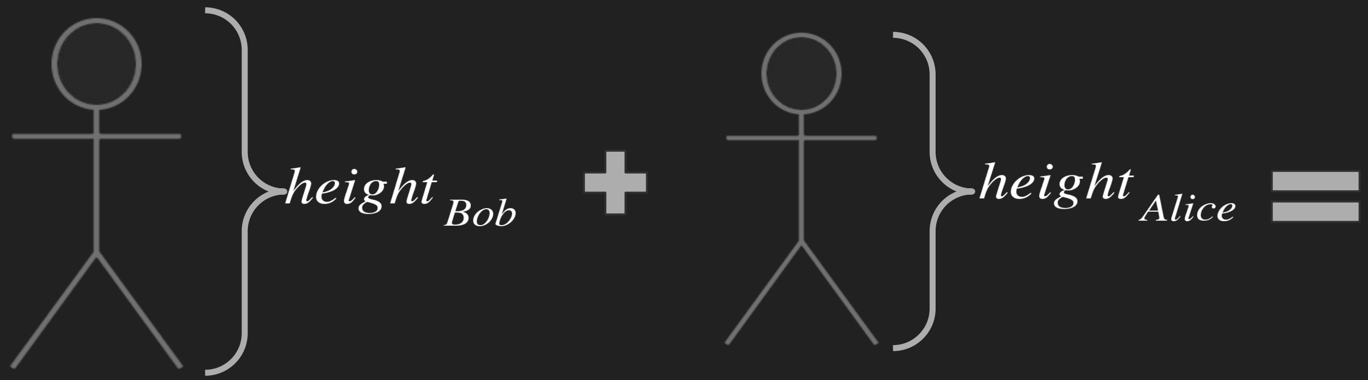
- Ex: Height Measurements



Datatype Issues: Provenance

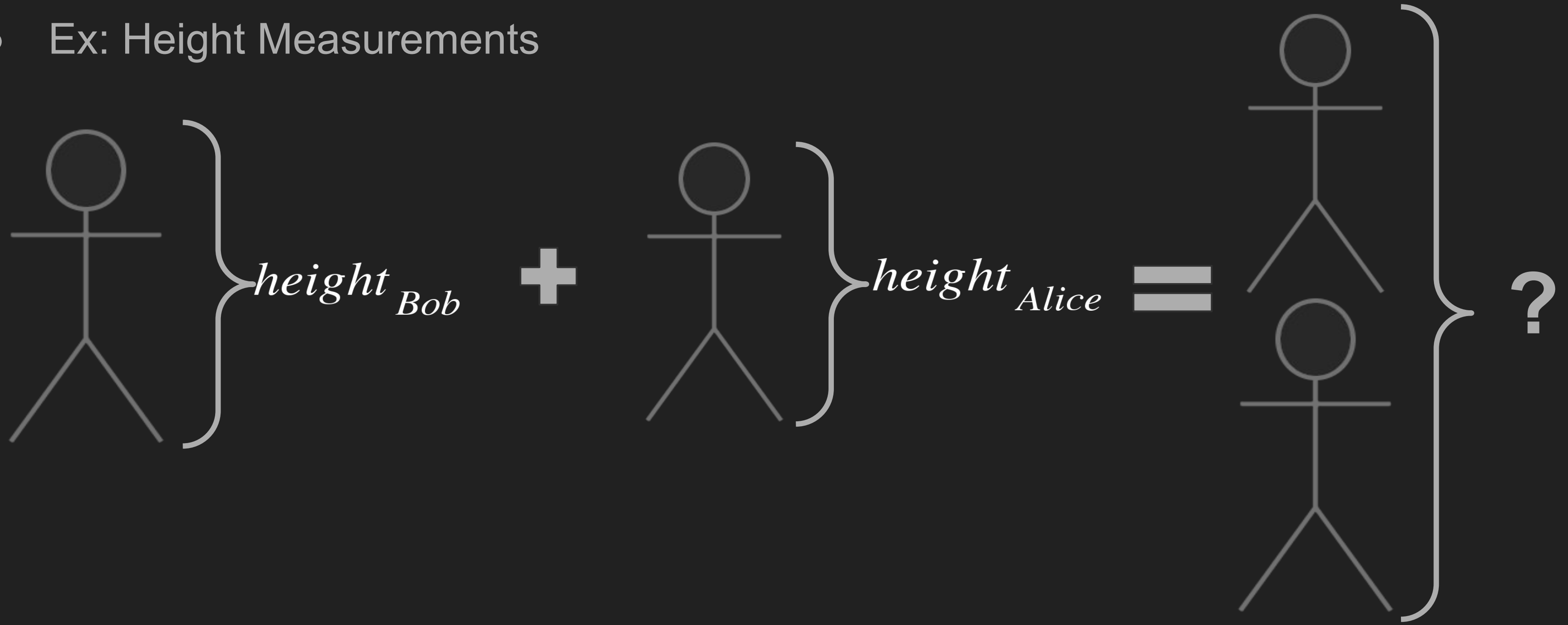
- Data science operations transform real-world interpretations

- Ex: Height Measurements



Datatype Issues: Provenance

- Data science operations transform real-world interpretations
- Ex: Height Measurements



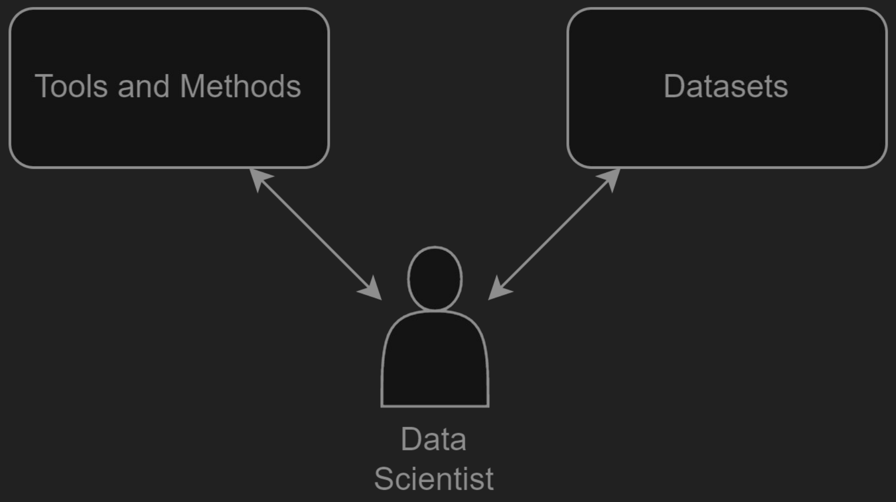
Outline

1. Why are datatypes problematic for data science?
    I. Datatype Problem Classes
    II. The gaps in current work
2. What are my contributions to solving this problem?
3. What is the significance of these contributions?
4. What are the next steps?

Datatype Issues: Current Solutions

- These approaches supplement datatypes with external knowledge and tools

- Applied in informal, ad-hoc, opaque, laborious ways



Datatype Issues: Current Solutions

1. Documentation

2. Provenance Tracking

3. Knowledge Representation

Datatype Issues: Current Solutions

1. **Documentation**

2. Provenance Tracking

3. Knowledge Representation

Documentation Standards

- Understand data through documentation standards
  - Provide list of important questions to be answered about the dataset

Motivation	Composition	Collection Process	Maintenance
...?	…?	…?	…?

Documentation Standards

- Understand data through documentation standards
  - Provide list of important questions to be answered about the dataset

Motivation	Composition	Collection Process	Maintenance
...?	…?	…?	…?

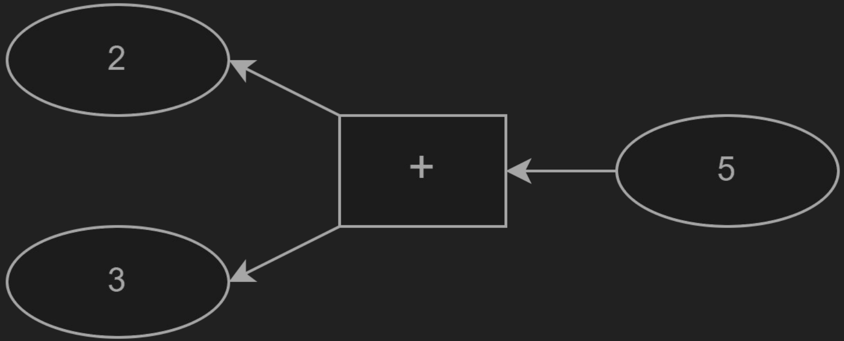
- We have a more complete picture of the dataset, however:
  - Description is still in natural language
  - Description is not provenance-integrated
  - Description is not machine readable

Datatype Issues: Current Solutions

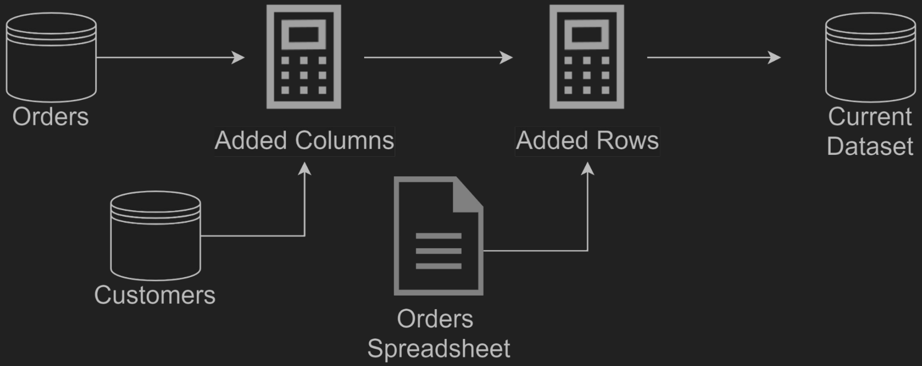
1. Documentation
2. **Provenance Tracking**
3. Knowledge Representation

Provenance Tracking

- Lineage-Provenance (What, How?)

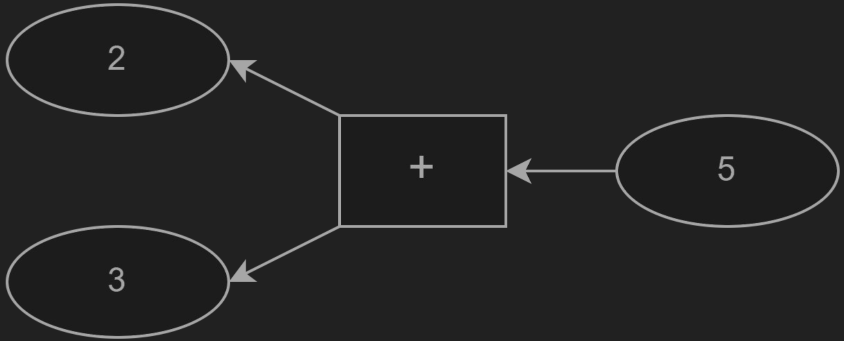


- Where-Provenance (Where From?)

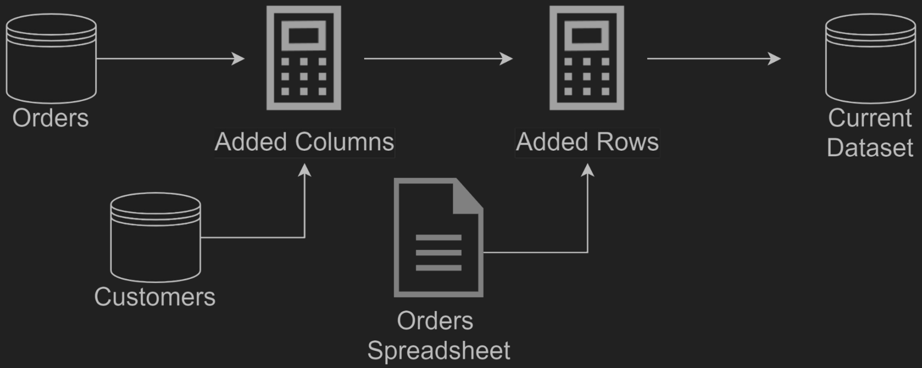


Provenance Tracking

- Lineage-Provenance (What, How?)



- Where-Provenance (Where From?)



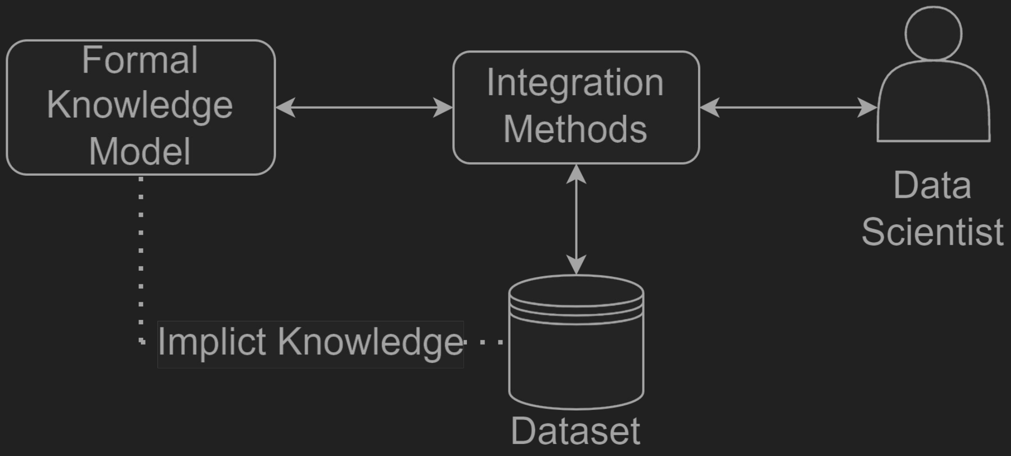
- No automatic error detection
- Does not encode real-world semantics
- Human verification still necessary

Datatype Issues: Current Solutions

1. Documentation
2. Provenance Tracking
3. **Knowledge Representation**

Knowledge Representation

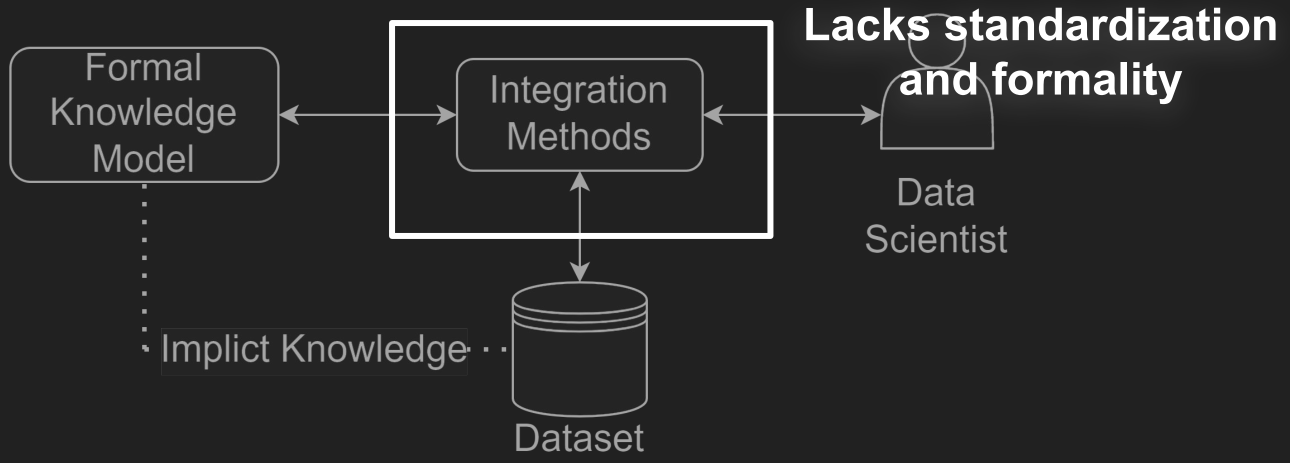
- Diverse knowledge can be represented at many levels of detail
- Integration method and role of data science operations is problematic



Knowledge Representation

- Diverse knowledge can be represented at many levels of detail

- Integration method and role of data science operations is problematic



Outline

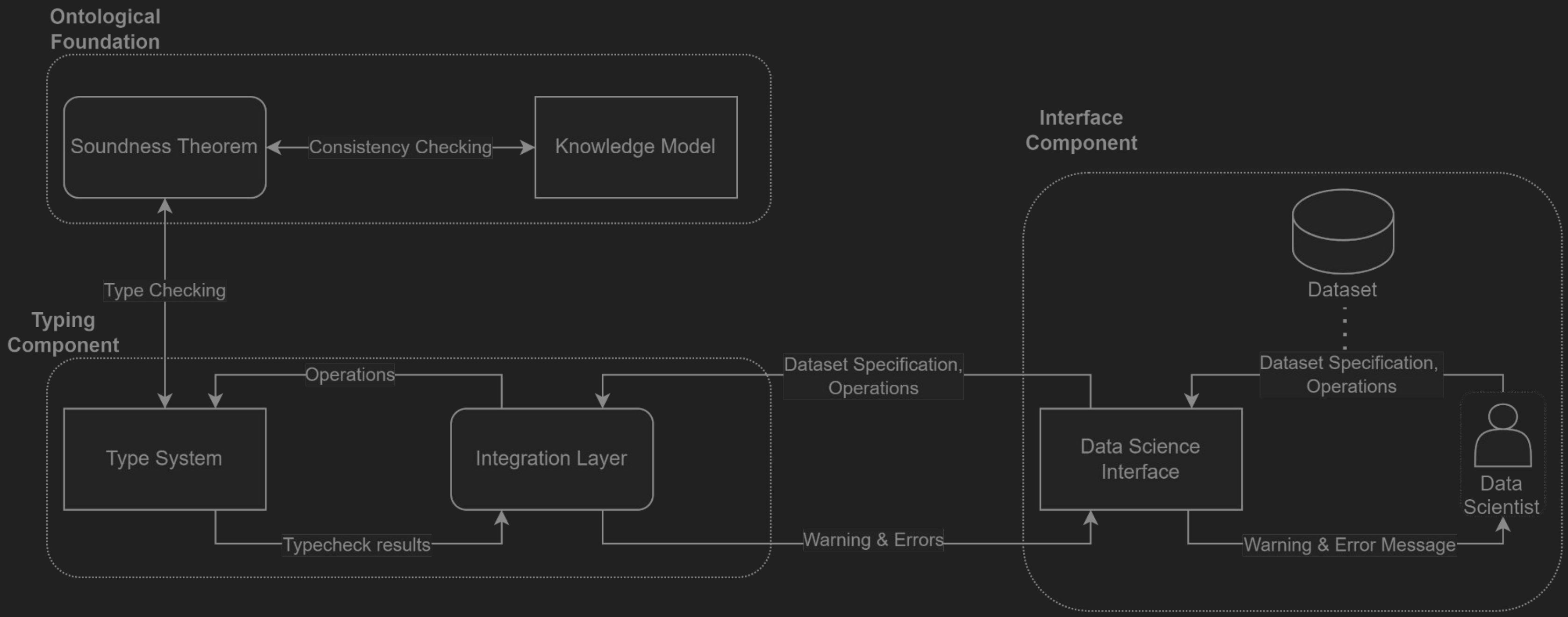
1. Why are datatypes problematic for data science?
2. **What are my contributions to solving this problem?**
3. What is the significance of these contributions?
4. What are the directions for future work?

The Meaningful Type Safety Framework (MeTS)

Ontologically-Sound Dependent Type Systems for Data Science

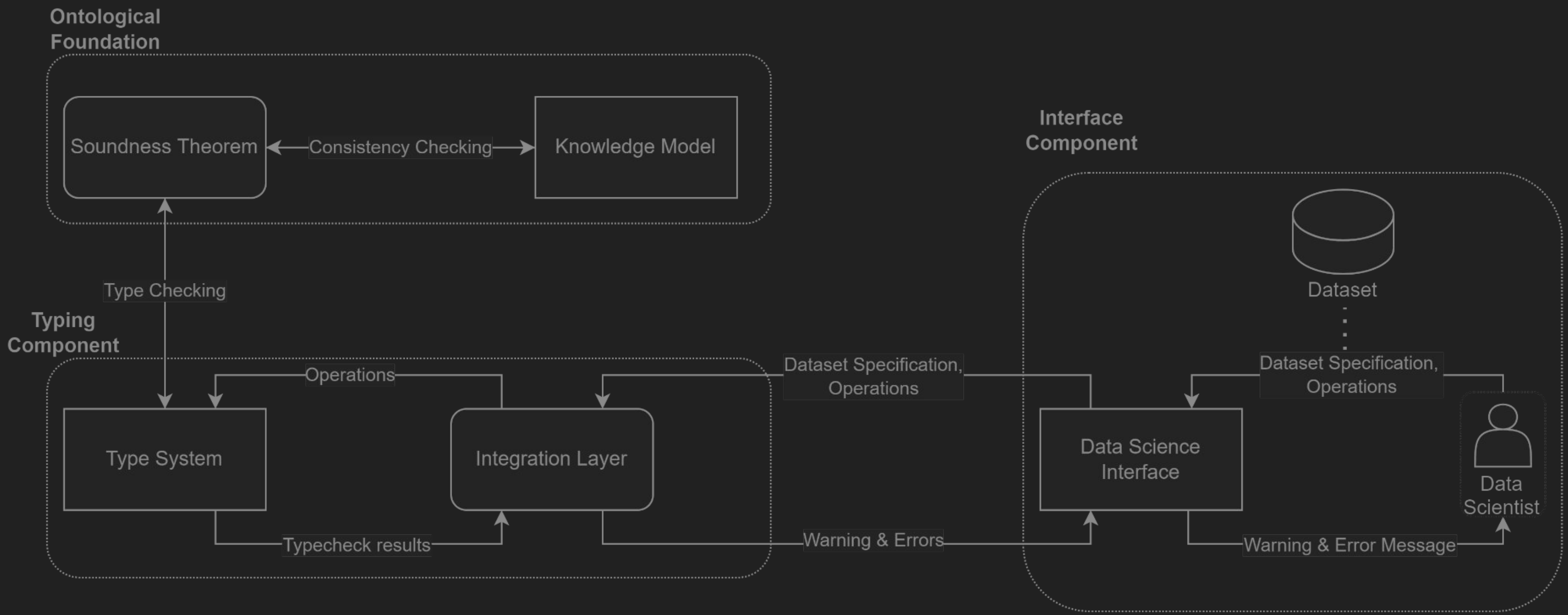
The Meaningful Type Safety Framework (MeTS)

Ontologically-Sound Dependent Type Systems for Data Science



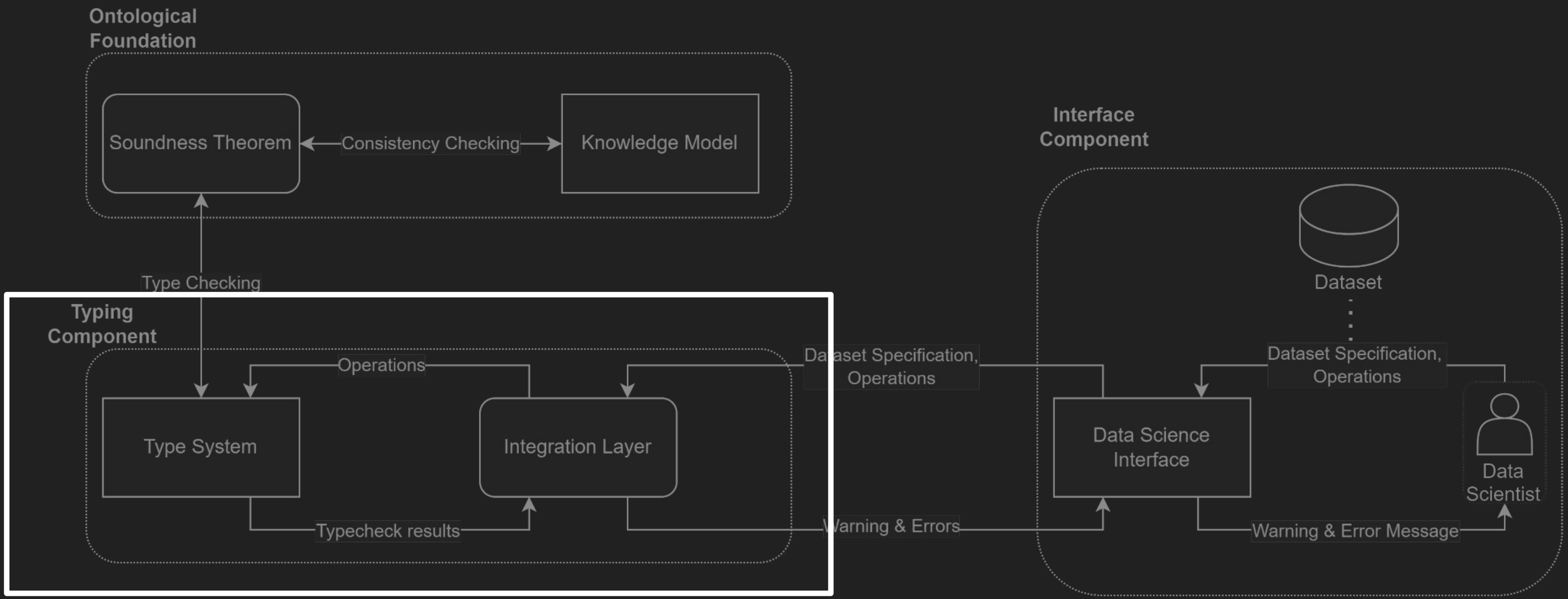
The Meaningful Type Safety Framework (MeTS)

Ontologically-Sound **Dependent Type Systems** for Data Science



The Meaningful Type Safety Framework (MeTS)

Ontologically-Sound **Dependent Type Systems** for Data Science



MeTS Type System

- Dependent pair types enforce operation preconditions

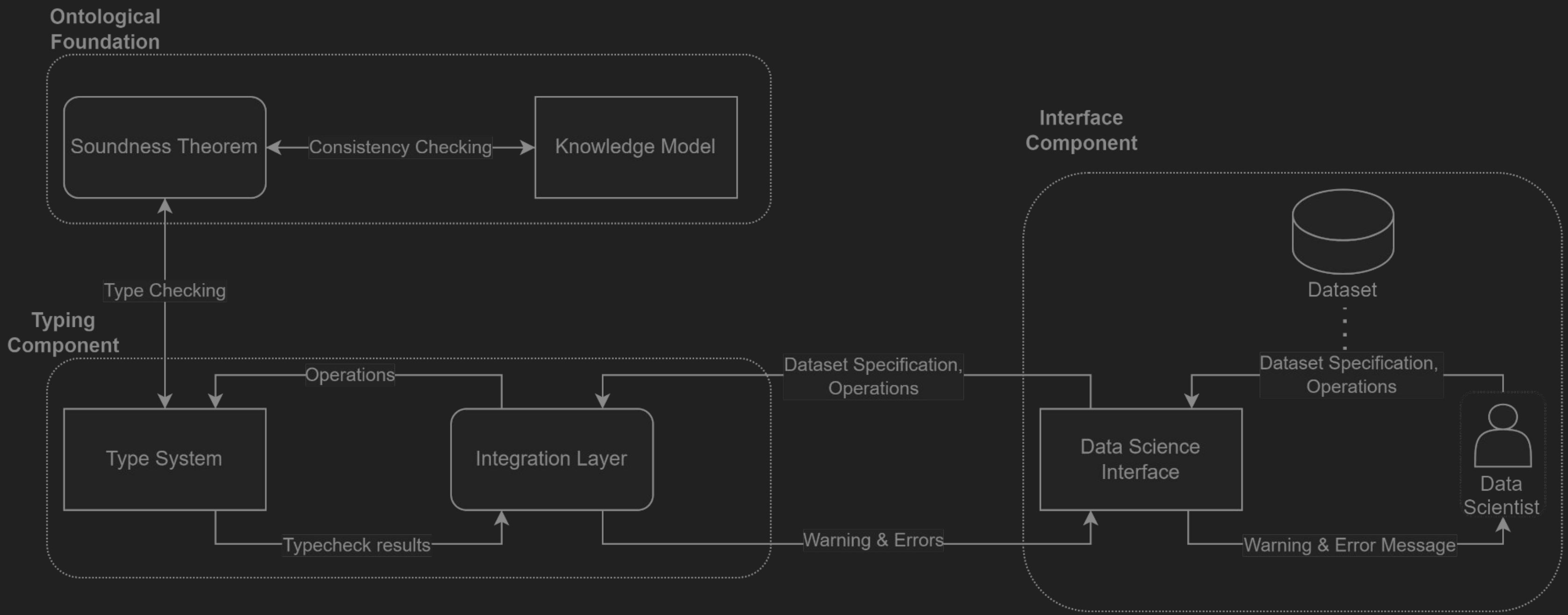
```
RegionSum : List Disjoint Region -> Region
```

- Values change after each operation : Provenance-Integrated

```
PopAvg(operands) = Avg Over operands
```

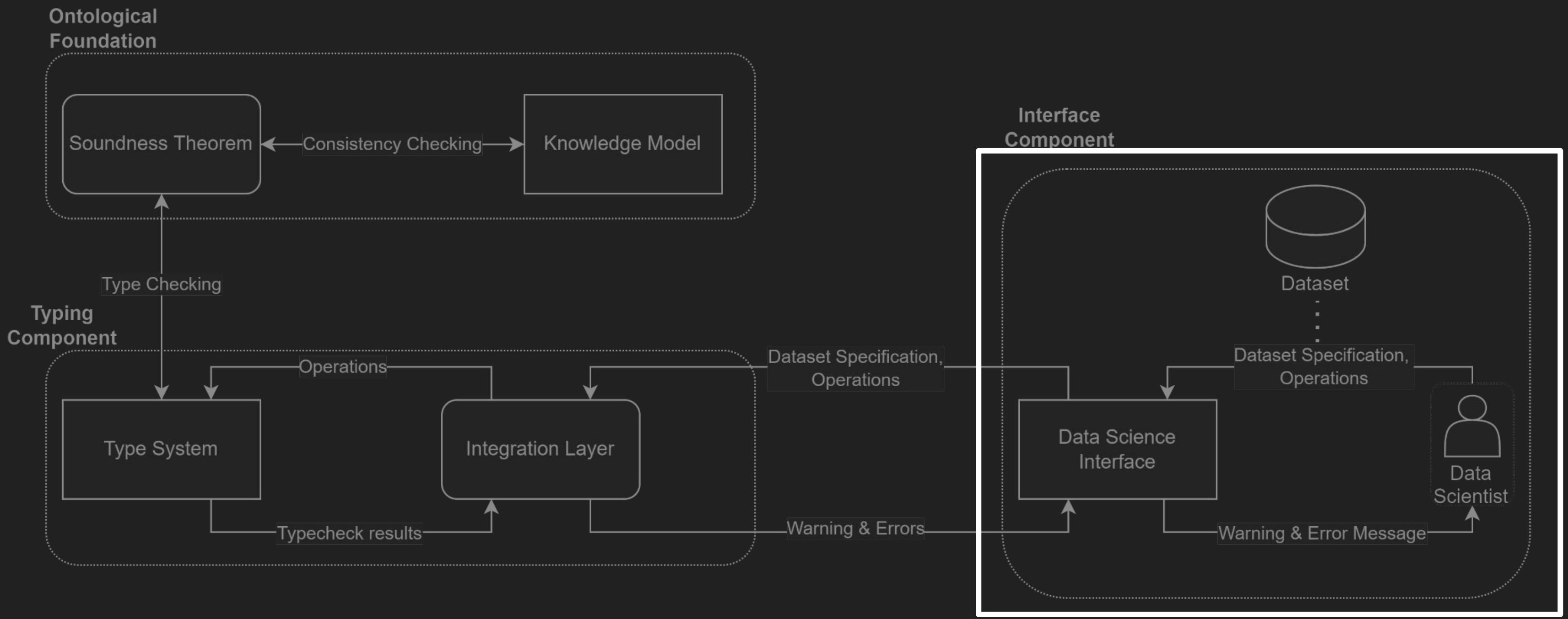
The Meaningful Type Safety Framework (MeTS)

Ontologically-Sound Dependent Type Systems **for Data Science**



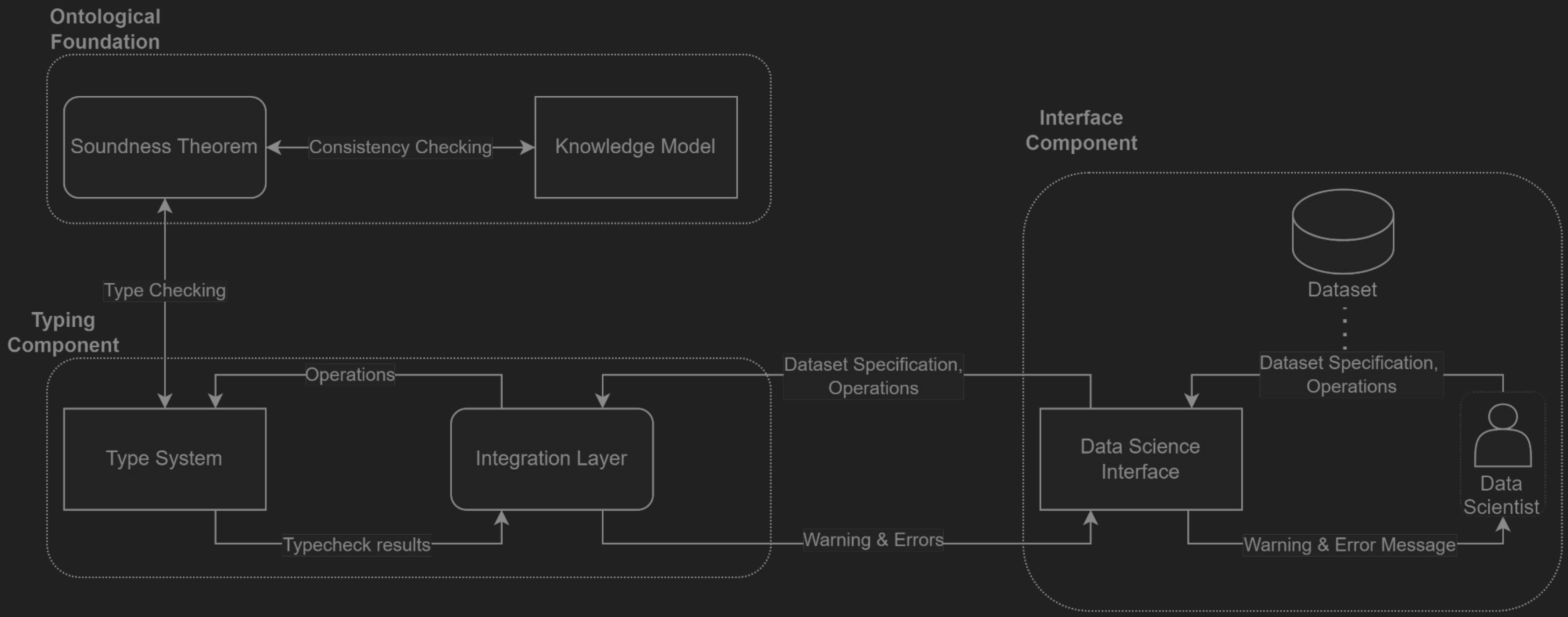
The Meaningful Type Safety Framework (MeTS)

Ontologically-Sound Dependent Type Systems **for Data Science**



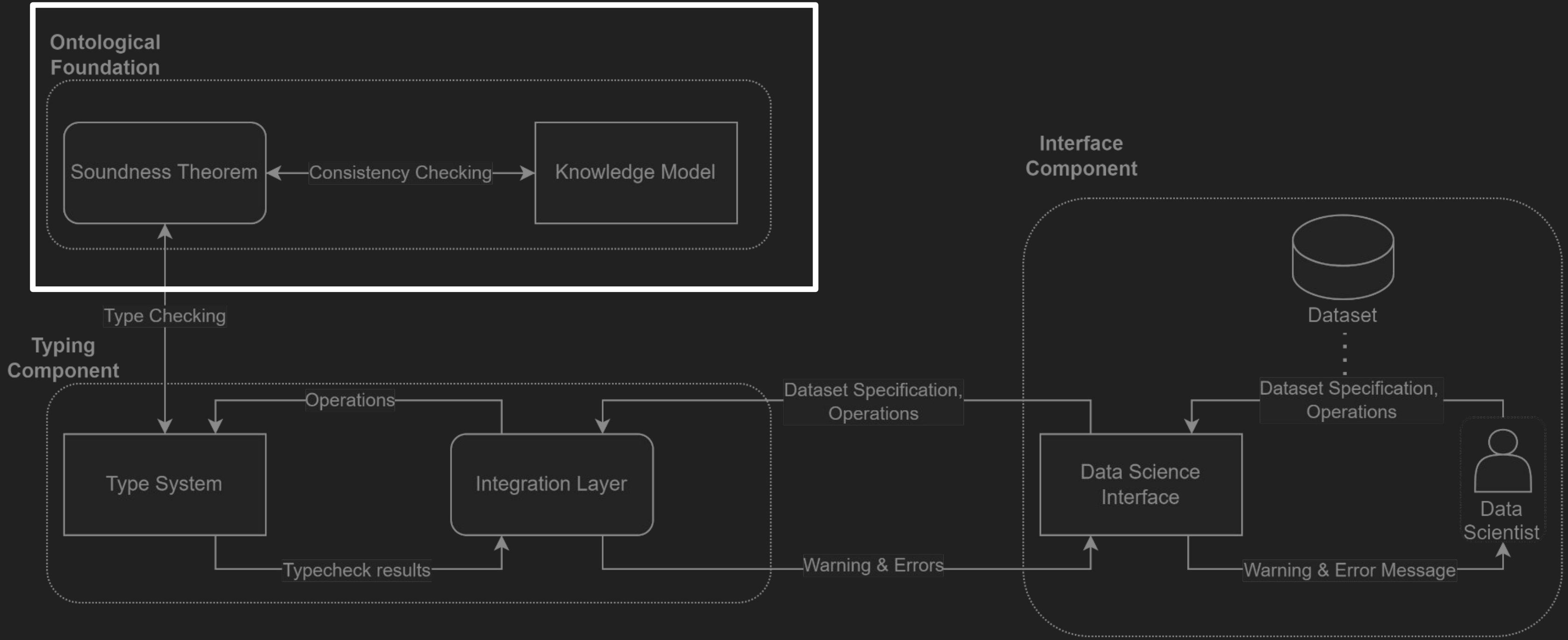
The Meaningful Type Safety Framework (MeTS)

Ontologically-Sound Dependent Type Systems for Data Science



The Meaningful Type Safety Framework (MeTS)

Ontologically-Sound Dependent Type Systems for Data Science

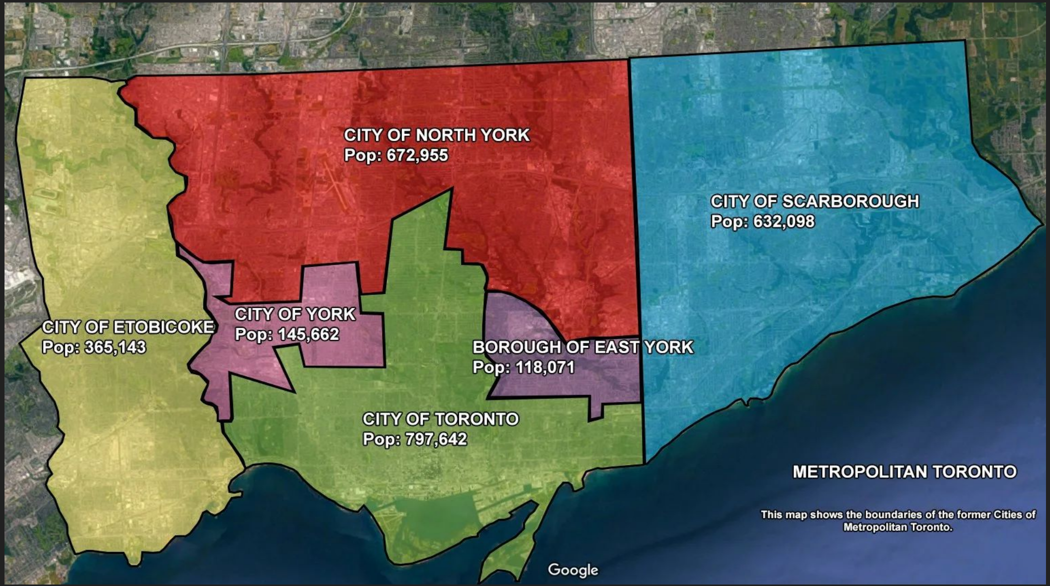


Census Data Ontology

- Represents the fundamental factors of census data
    - Movement of People
    - Crowd mereology
    - Geopolitical occupation
    - Geospatial mereology

Census Data Ontology

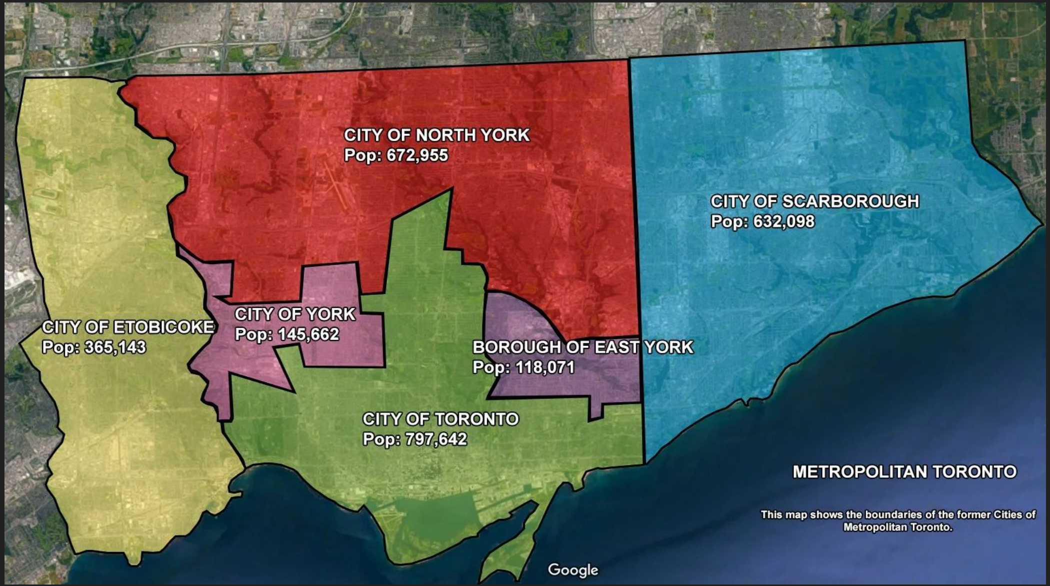
- Represents the fundamental factors of census data
  - Movement of People
  - Crowd mereology
  - Geopolitical occupation
  - Geospatial mereology



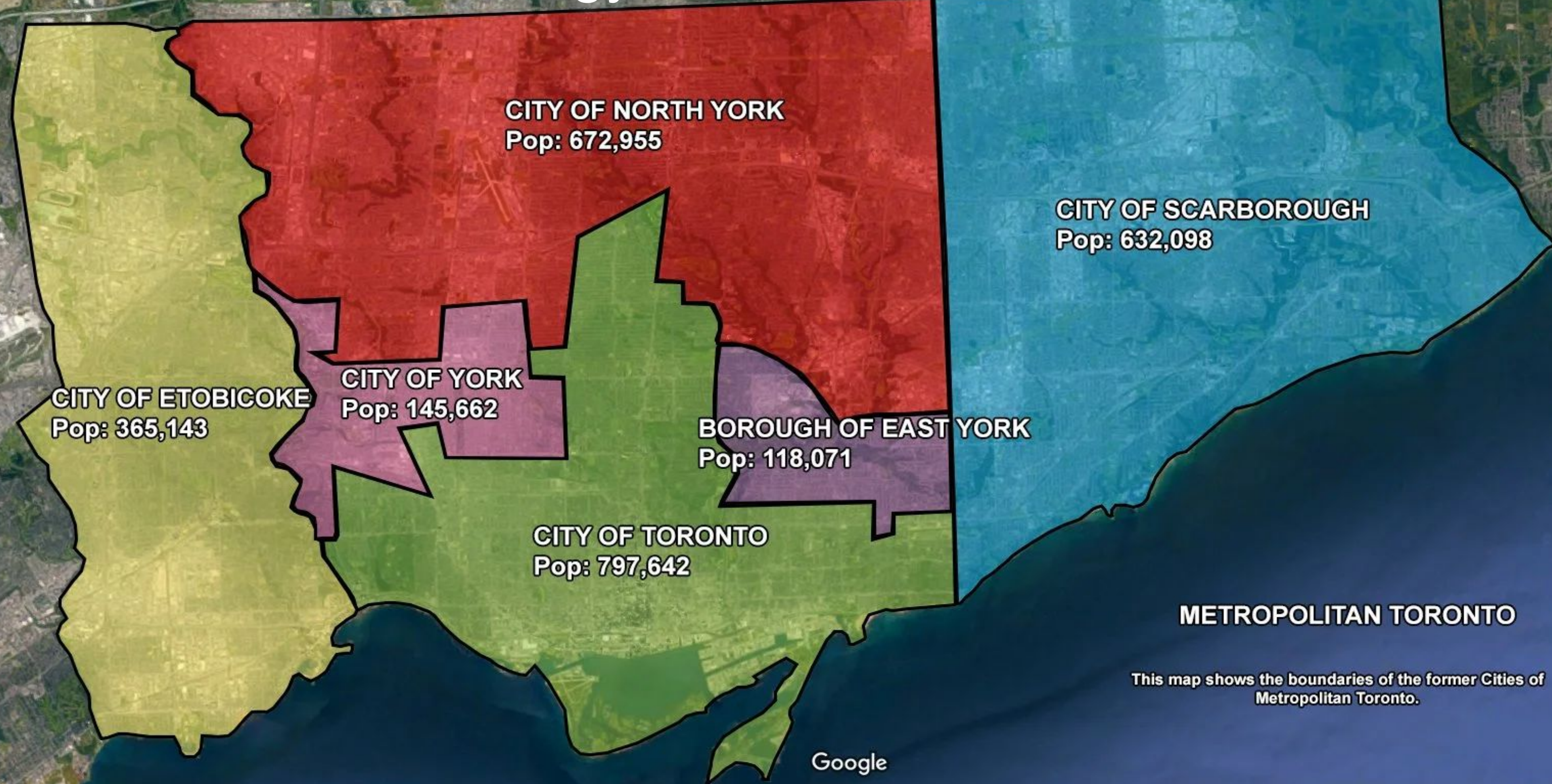
Census Data Ontology

- “I live in Toronto” - different interpretations over time

- Toronto as a region of land vs the geopolitical entity

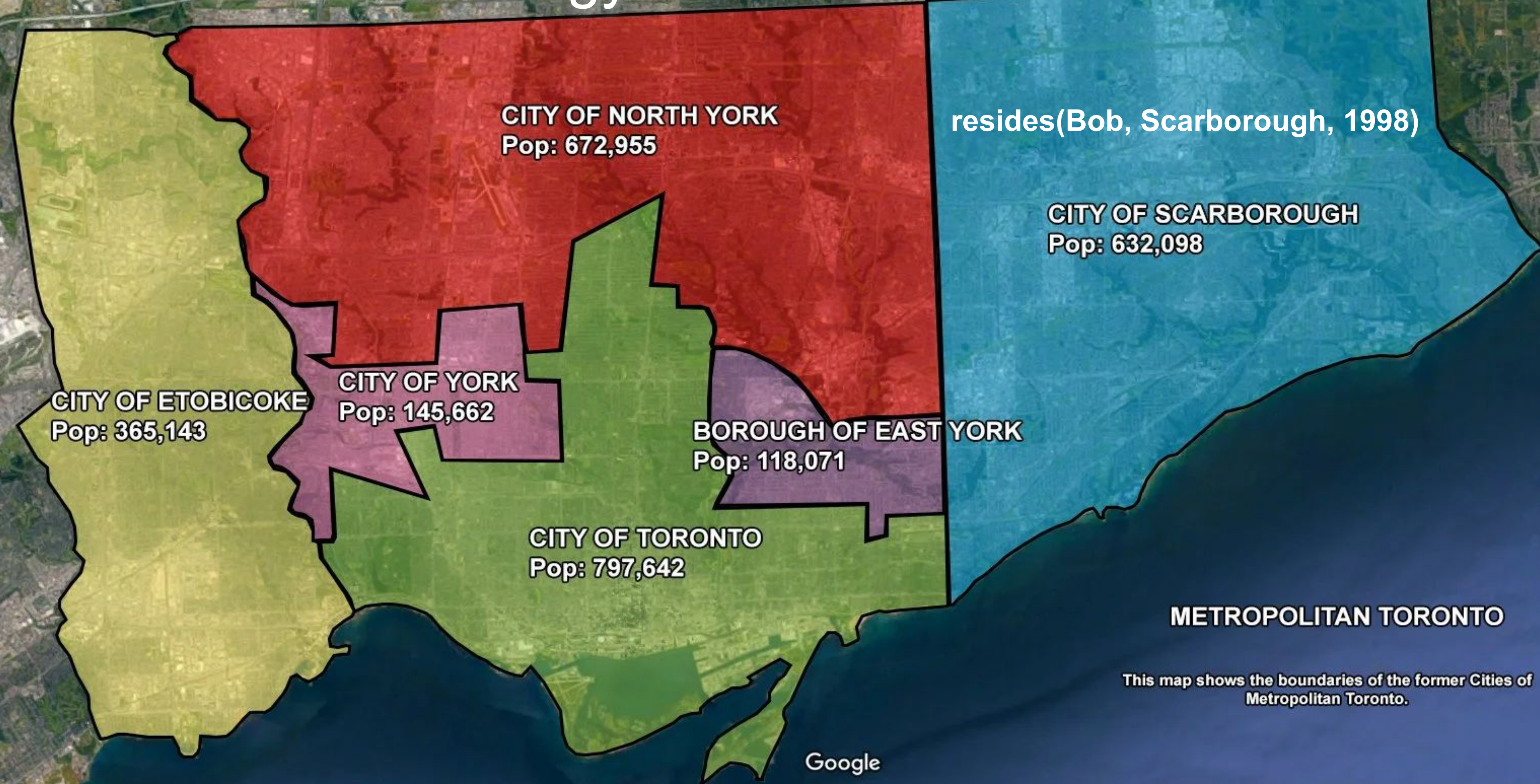


Census Data Ontology

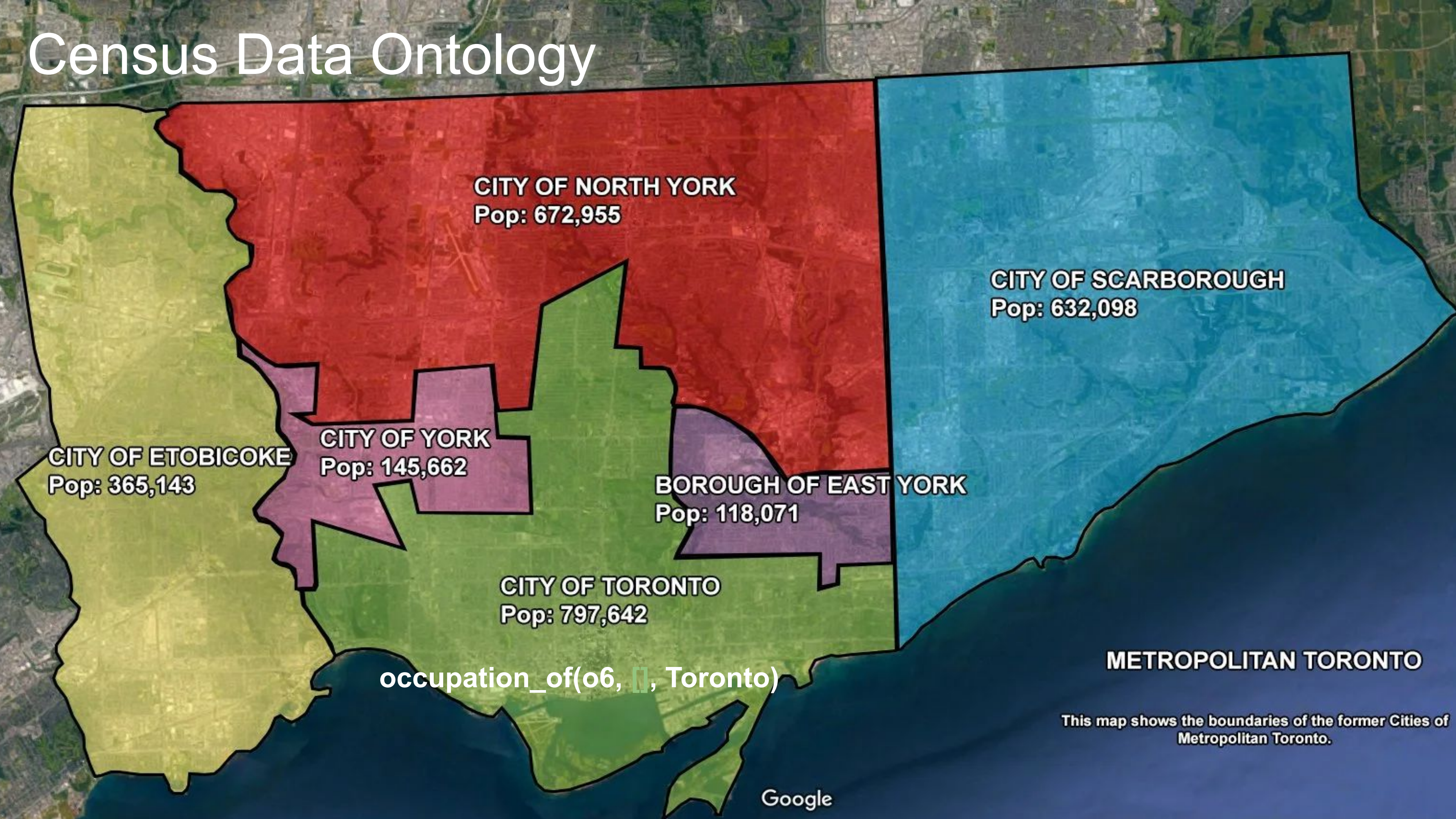


Census Data Ontology

resides(Bob, Scarborough, 1998)

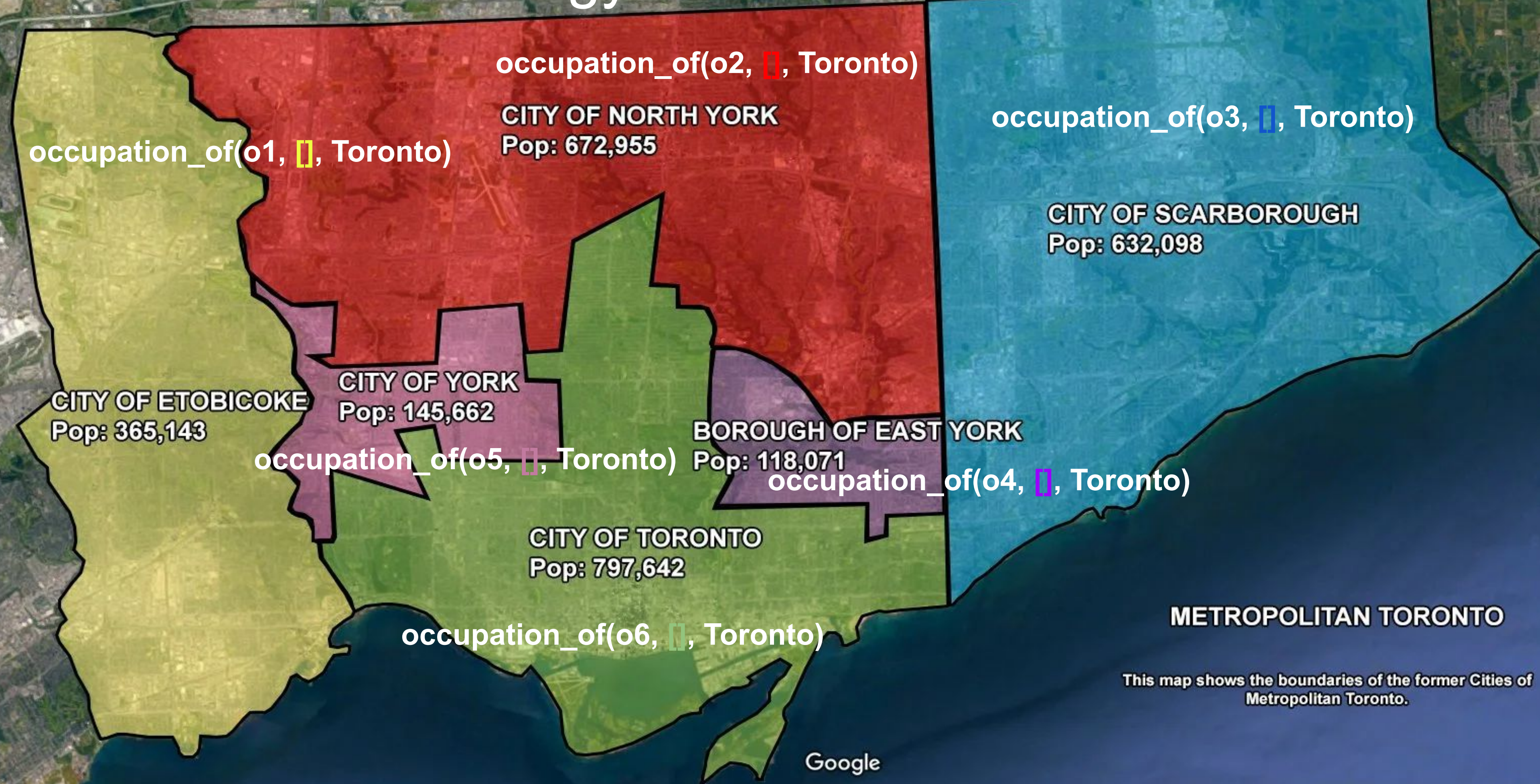


Census Data Ontology



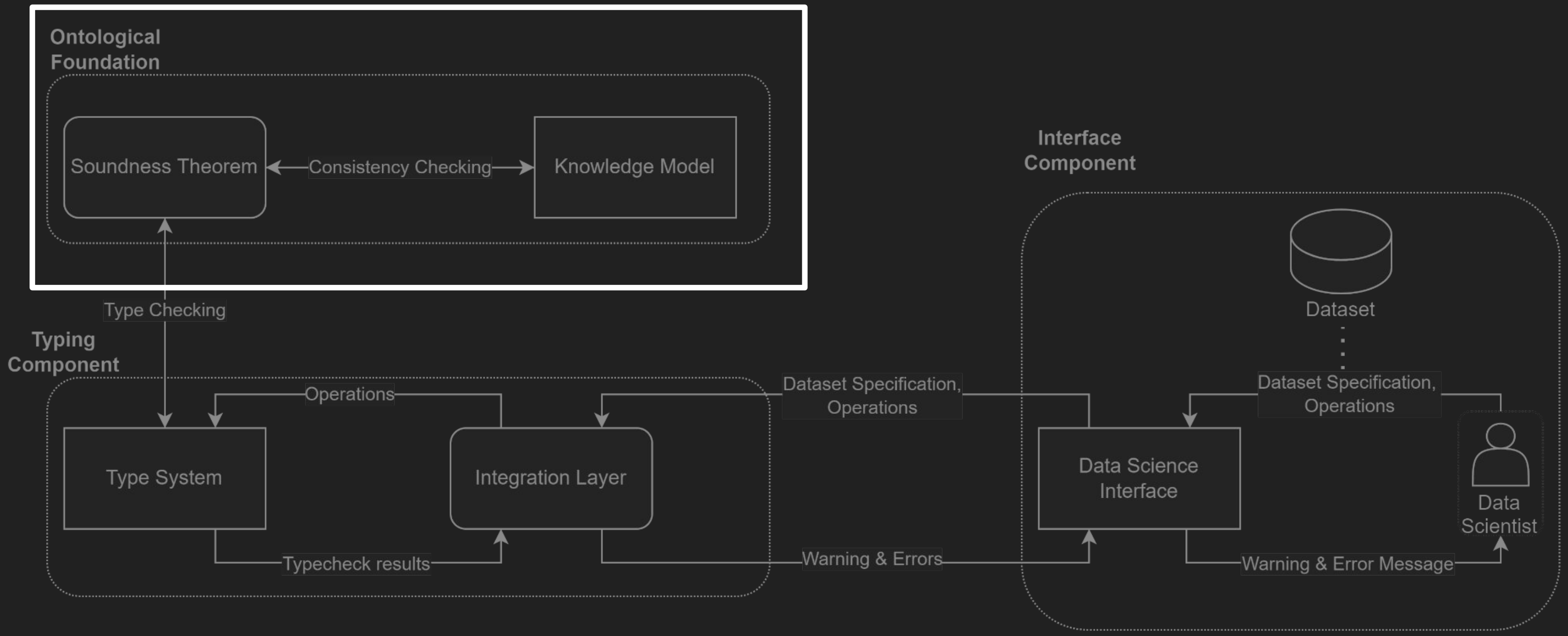
occupation_of(o6, [], Toronto)

Census Data Ontology



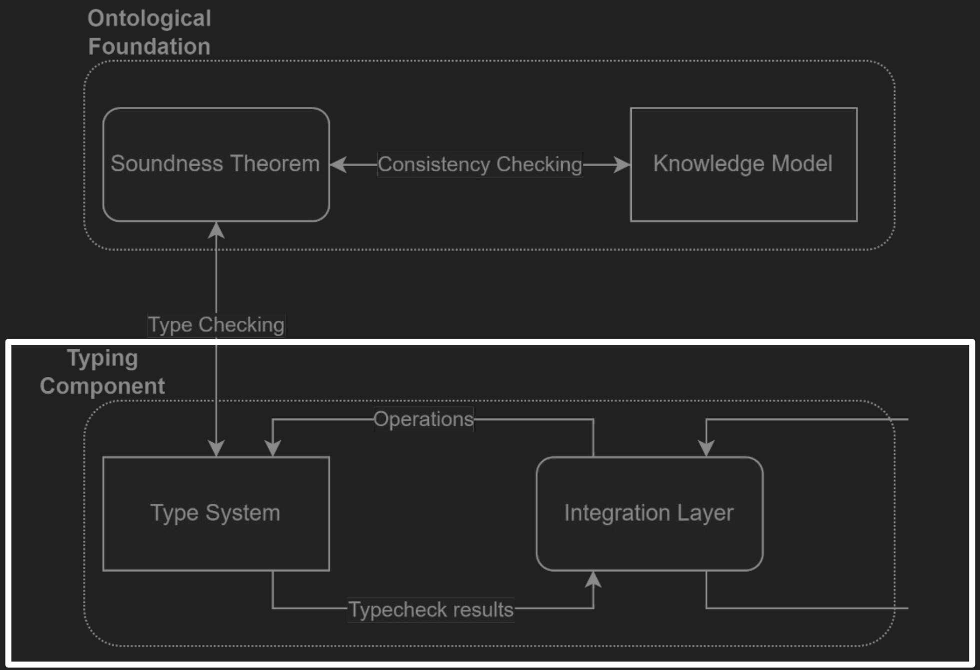
The Meaningful Type Safety Framework (MeTS)

Ontologically-Sound Dependent Type Systems for Data Science



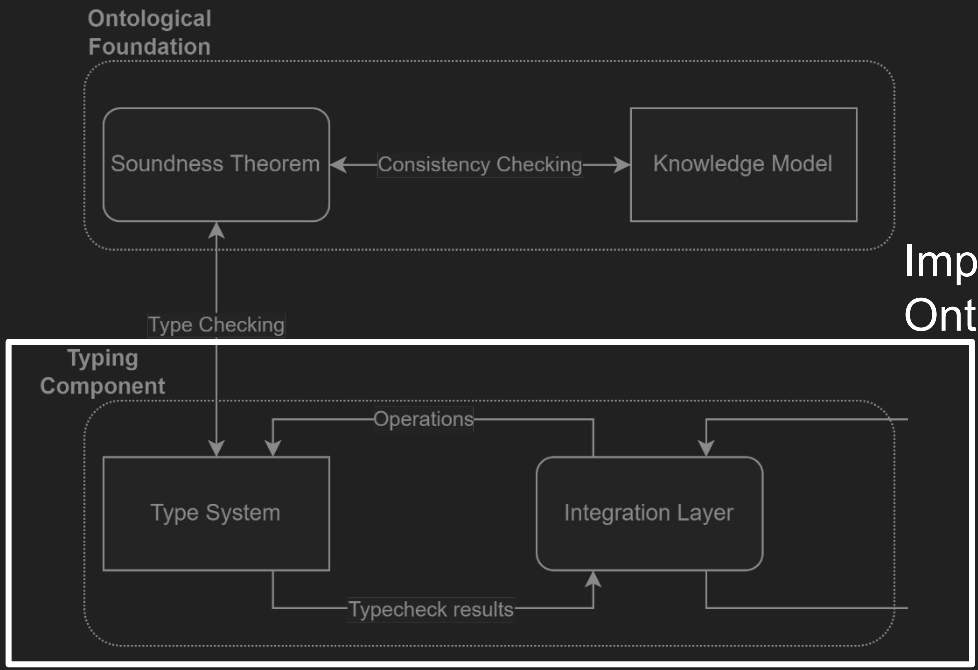
The Meaningful Type Safety Framework (MeTS)

Ontologically-Sound Dependent Type Systems for Data Science



The Meaningful Type Safety Framework (MeTS)

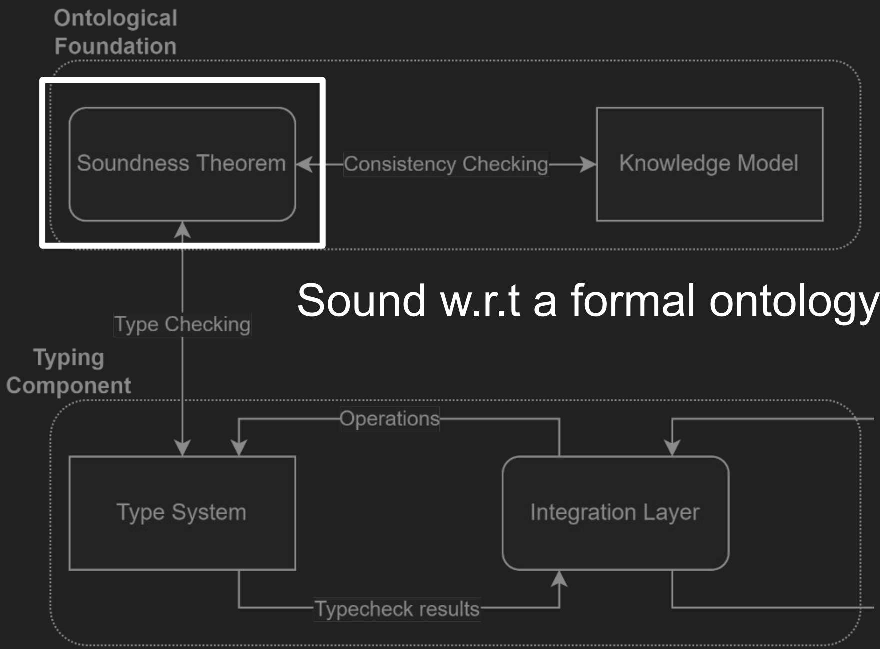
Ontologically-Sound Dependent Type Systems for Data Science



Implicit
Ontological Commitments

The Meaningful Type Safety Framework (MeTS)

Ontologically-Sound Dependent Type Systems for Data Science



Sound w.r.t a formal ontology

Outline

1. Why are datatypes problematic for data science?
2. What are my contributions to solving this problem?
3. **What is the significance of these contributions?**
4. What are the directions for future work?

Contributions & Significance

1. MeTS Framework

2. Census Data Ontology

3. MeTS Soundness Theorem

Contributions & Significance

1. **MeTS Framework**

2. Census Data Ontology

3. MeTS Soundness Theorem

Contributions & Significance

1. **MeTS Framework**
   I. Applies dependent types to model real-world knowledge for data science
   II. Elevates type safety to a meaningful result
   III. Ensures real-world interpretation is upheld throughout the data science pipeline

2. Census Data Ontology

3. MeTS Soundness Theorem

Contributions & Significance

1. MeTS Framework

2. **Census Data Ontology**

3. MeTS Soundness Theorem

Contributions & Significance

1. MeTS Framework

2. **Census Data Ontology**
   I. Models fundamental factors of census data
   II. Provides exceptional expressiveness
   III. Models census data operations

3. MeTS Soundness Theorem

Contributions & Significance

1. MeTS Framework

2. Census Data Ontology

3. **MeTS Soundness Theorem**

Contributions & Significance

1. MeTS Framework

2. Census Data Ontology

3. **MeTS Soundness Theorem**
   I. **Decouples ontological commitments from type system implementation**
   II. **Enables increased knowledge sharing and interoperability**

Outline

1. Why are datatypes problematic for data science?
2. What are my contributions to solving this problem?
3. What is the significance of these contributions?
4. **What are the directions for future work?**

Future Work

1. Correspondence Theorem

2. Ontology for Data Quantities

3. Tractability

Future Work

1. **Correspondence Theorem**

2. Ontology for Data Quantities

3. Tractability

Future Work

1. **Correspondence Theorem**
   I. **Soundness AND completeness**
   II. **Requires detailed specification for logic and type systems**
   III. **New methods of collaboration**

2. Ontology for Data Quantities

3. Tractability

Future Work

1. Correspondence Theorem

2. **Ontology for Data Quantities**

3. Tractability

Future Work

1. Correspondence Theorem

2. **Ontology for Data Quantities**
    I. **Meta-model of existing ontologies**
    II. **Provenance and operation-centric**
    III. **Guides development of future ontologies and MeTS**

3. Tractability

Future Work

1. Correspondence Theorem

2. Ontology for Data Quantities

3. Tractability

Future Work

1. Correspondence Theorem

2. Ontology for Data Quantities

3. **Tractability**
   I. **Minimal change to data scientists workflow**
   II. **Integrate MeTS into existing data science tools**
   III. **Enable increased adoption**

Thank you!

Questions?

References

1. Gebru, Timnit, et al. "Datasheets for datasets." Communications of the ACM 64.12 (2021): 86-92.
2. Herschel, Melanie, Ralf Diestelkämper, and Houssem Ben Lahmar. "A survey on provenance: What for? What form? What from?." The VLDB Journal 26.6 (2017): 881-906.
3. Acar, Umut A., et al. "A Graph Model of Data and Workflow Provenance." TaPP. 2010.
4. Grüninger, Michael, et al. "Foundational Ontologies for Units of Measure." FOIS. 2018.
5. Firat, Aykut. Information integration using contextual knowledge and ontology merging. Diss. Massachusetts Institute of Technology, 2003.
6. Fox, Mark S. "The semantics of populations: A city indicator perspective." Journal of Web Semantics 48 (2018): 48-65.
7. Fox, Mark S. "An ontology engineering approach to measuring city education system performance." Expert Systems with Applications 186 (2021): 115734.
8. Albuquerque, Antognoni, and Giancarlo Guizzardi. "An ontological foundation for conceptual modeling datatypes based on semantic reference spaces." IEEE 7th International Conference on Research Challenges in Information Science (RCIS). IEEE, 2013.
9. Löh, Andres, Conor McBride, and Wouter Swierstra. "A tutorial implementation of a dependently typed lambda calculus." Fundamenta informaticae 102.2 (2010): 177-207.
10. Brady, Edwin. "Idris, a general-purpose dependently typed programming language: Design and implementation." Journal of functional programming 23.5 (2013): 552-593.

References

11. Balasubramanian, Vidhya. "InGIST: A Queryable and Configurable IndoorGIS Toolkit." Geospatial Infrastructure, Applications and Technologies: India Case Studies. Springer, Singapore, 2018. 93-105.
12. Abdelmoty, Alia I., et al. "A critical evaluation of ontology languages for geographic information retrieval on the Internet." Journal of Visual Languages & Computing 16.4 (2005): 331-358.
13. A Shallow Embedding of Pure Type Systems into First-Order Logic
14. Dapoigny, Richard, and Patrick Barlatier. "Towards Ontological Correctness of Part-whole Relations with Dependent Types." FOIS. Vol. 2010. 2010.
15. Barlatier, Patrick, and Richard Dapoigny. "A type-theoretical approach for ontologies: The case of roles." Applied Ontology 7.3 (2012): 311-356.
16. Dapoigny, Richard, and Patrick Barlatier. "Formalizing context for domain ontologies in Coq." Context in computing. Springer, New York, NY, 2014. 437-454.

Appendix / FAQ

- Type Theoretic Questions
- Knowledge Modelling Questions
- Implementation Questions
- Related Work Questions

Appendix / FAQ - Type Theoretic

- Why dependent types?
- What is the importance of Curry-Howard isomorphism?
- Reasoning for notational choices

Back to Appendix

Appendix / FAQ

- Why Dependent Types?
  - Great expressiveness, still decidable
  - Datasets already have types associated with them, existing type-checks are trivial at best, idea is to simply make type-checking more robust through more expressive types
  - Type system is more directly integrated
- Curry-howard isomorphism and soundness thm also ensures ontological commitments of implementation can be made explicit and decoupled, key advantage

Back to Appendix

Appendix / FAQ

- Notational Choices
  - Close to martin-lof notation for dependent types, which is the foundation for dependent types, well-understood and compact
  - Functional programming syntax is very similar to haskell, ML, nothing revolutionary or unseen, should be simple to grasp for those familiar with functional programming

Back to Appendix

Appendix / FAQ - Knowledge Modelling

- **What is soundness vs completeness, how does it fit into correspondence theorem?**
- **Reasoning for soundness theorem to be constructed as it was**
- **Why not OWL?**
- **What about more sophisticated mereologies?**

Back to Appendix

Appendix / FAQ - Soudness/Completeness

- What is soundness?
  - Ensuring that a mets type environment is sound w.r.t a first order logic theory by proving that any inconsistent FOL sentence is translated from a type-unsafe ground term
- What is completeness?
  - Ensure that soundness is true in both directions, to also ensure that any FOL sentence which maps to a type-unsafe term is inconsistent
- Why prove soundness in the method I did?
  - For data science, the verification of rules is done by type-checking, seeing if a value satisfies the definition of a type. We are interested in why these checks would fail. On the FOL side, this is equivalent to determining when a sentence produces an inconsistency, so proof should show these co-occurring

Back to Appendix

Appendix / FAQ - Why not OWL

- Limitation of XSD datatypes for operations
  - We need to represent quantities associated with many concepts and place restrictions on the ways in which they may be combined
  - OWL alone does not support interpretations of quantities in conjunction with provenance of data science operations
- Why not supplement OWL?
  - If reliant on external tools to make up for lack of expressiveness, ontological commitments are being upheld implicitly, outside of the formal knowledge model, which defeats the purpose of a formal knowledge model

Back to Appendix

Appendix / FAQ - Mereological Expressiveness

- Functionality of the mereology comes from the dependently-typed evaluation
  - Type terms contain functions over data structures, these modelling decisions determine the level of expressiveness of the ontology
  - Functions must be total to keep type-checking decidable
  - This problem overlaps with additional work for correspondence theorem, to determine which FOL axioms in general could be represented in the type system. As it stands, a complete answer cannot be provided

Back to Appendix

Appendix / FAQ - Implementation

- **Why idris?**
- **What is dataset specification?**
- **How do expression trees work?**
- **Why StatCAN census data?**

Back to Appendix

Appendix / FAQ - Idris

- **Why idris?**
    - General-purpose programming - Only current work that deals with representing ontologies with dependent types utilizes less accessible presentation of dependent types in the context of Coq, a proof assistant
    - Idris is a general-purpose programming language with dependent types, it is far more accessible to programmers and data scientists in general, fits with the scope and intended audience of MeTS
    - Idris has an active community, lots of support available
    - Idris syntax is clean and familiar to functional programmers

Back to Appendix

Appendix / FAQ - Dataset Specification

- **What is dataset specification?**
  - Associating a dataset with its real-world concepts
  - Once the dataset and real-world concepts are linked, real-world rules pertaining to those concepts will be enforced through any subsequent data science operation

Back to Appendix

Appendix / FAQ - Expression Trees

- **How do expression trees work?**
    - Values encode provenance
    - Function preconditions check values for provenance, to enforce provenance-based rules

```
PopAvg(operands) = Avg Over operands

PopSum : List Not Aggregated Population
```

Back to Appendix

Appendix / FAQ - StatCAN Motivation

- **Why StatCAN Census Data?**
    - Time, Geospatial Regions, Counts, all generalizable concepts important to many datasets
    - Publicly available and widely-documented data

Back to Appendix

Appendix / FAQ - Related Work

- Barlartier and Dapoigny's work
- City indicators
- Fox and Huang's Knowledge Provenance
- Provenance Graphs
- Refinement types

Back to Appendix

Appendix / FAQ - Provenance Graphs

- Provenance graphs are formal but do not formally capture real-world information
  - They provide a methodology to formally represent data provenance
  - They do not provide a method to interpret this information with respect to the concepts represented within the dataset

Back to Appendix

Appendix / FAQ - City Indicators

- **How does census data ontology differ from GCI, city indicators, etc?**
    - Intent - intended specifically for modelling fundamental factors of census data to be interpreted in the data science application
    - Expressiveness - Specified in FOL rather than owl because of MeTS representational requirements

Back to Appendix

Appendix / FAQ - Barlartier/Dapiogny

- **Barlartier and Dapoigny's work**
  - K-DTT - expresses ontological classes through dependent types, uses DOLCE as a high-level general taxonomy
  - Relates description logic to dependent types using the Coq theorem prover

Back to Appendix

Appendix / FAQ - Refinement Types

- **Why not liquid haskell, Scala refined, etc?**
  - Refinement types rely on type inference rather than type-checking
  - Refinement types are a kind of sub-typing, rather than constructing new types
  - Dependent pair types are supplied with a proof of the condition holding, refinement types need to find the proof themselves, liquid haskell does so using an SMT solver
  - Validating data science operations is reflected better in type-checking than in performing type-inference, so relying on dependent pair types makes more sense

Type Inference	Type Checking
What rules does this computation uphold? What is the most restrictive rule this computation upholds?	Here are the rule(s) I want this computation to follow, does it uphold them?

Back to Appendix

Appendix / FAQ

- What about knowledge provenance work?
  - Knowledge provenance in general is different from data science provenance knowledge
  - MeTS focuses not on the where/how provenance and trustworthiness, but on how real-world interpretations are affected by data science operations
  - Not only explain where/how quantities are obtained, but what those derived quantities now represent and how they could be further manipulated

Back to Appendix