Semantic data pipeline methodology and observability: giveadam project

Charles T. Gray

September 20, 2025

Abstract

This document presents a novel semantic layering methodology for research data pipeline design, demonstrated through analysis of UN Sustainable Development Goal (SDG) priorities from dam-affected communities in North India. Survey data was collected by Garima Gupta from residents of Tehri (where a dam was constructed 20 years ago) and Arunachal Pradesh (where a dam is under development).

The central methodological contribution is a semantically-structured approach to data transformation that extends dbt's engineering-focused patterns to prioritize research interpretability and stakeholder communication. Unlike standard dbt implementations that emphasize technical efficiency, this semantic layering methodology explicitly preserves research context, makes harmonization decisions transparent, and maintains conceptual clarity throughout the data pipeline.

The semantic layer architecture consists of four conceptually distinct stages: source base (raw data cleaning), source entities (region-specific preservation), semantic models (cross-source harmonization), and analytic models (research-ready datasets). This approach addresses the critical gap between technical data engineering practices and research methodology requirements, enabling automated observability while maintaining scholarly rigor.

This semantic layering methodology offers a replicable framework for research data management that bridges the divide between engineering efficiency and research transparency, with applications across all domains requiring multi-source data integration and stakeholder communication.

Contents

1 What are these data?

The data in this project aggregates two region-based surveys, with responses from participants in Tehri and Arunachal Pradesh, focusing on their ranked top three priorities of the United Nations Sustainable Development Goals (UN SDGs). The data was collected by Garima Gupta by surveying residents of the two regions in Hindi. The data is stored in the `data/` directory of the giveadam repository (see Section 2.2 for complete repository structure). This project was created by Charles T. Gray to aggregate survey responses across regions using a novel semantic layering methodology (detailed in Section 2) to investigate the following question.

> *What are the differences in SDG priorities between residents of Tehri, where a dam was constructed 20 years ago for hydroelectric power, and Arunachal Pradesh, where a dam is currently being developed?*

Figure 1 shows a treemap of the top 3 UN SDG priorities from survey respondents in Tehri (dam constructed 20 years ago) and Arunachal Pradesh (dam under development), North India.

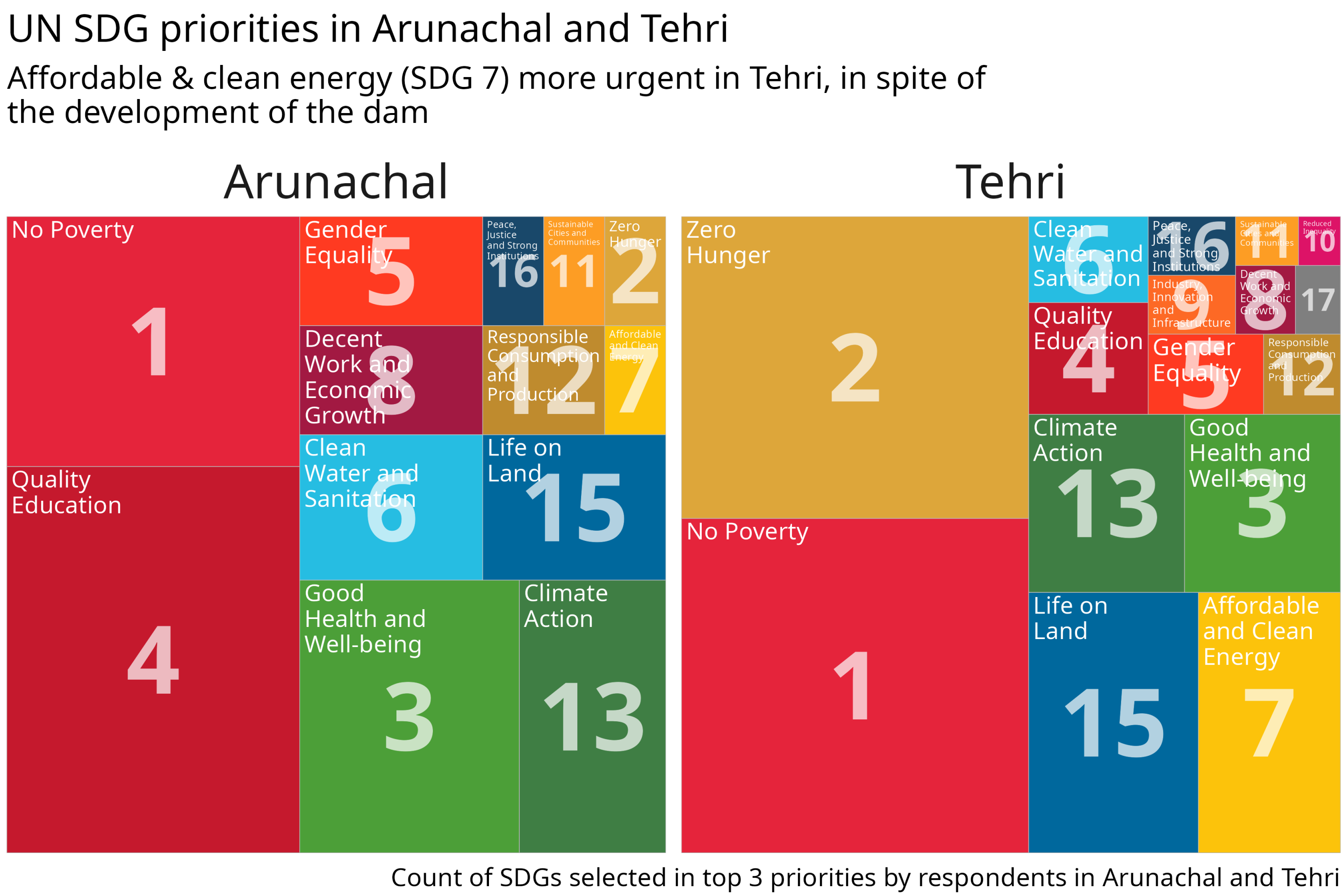

Figure 1: Treemap of top 3 UN SDG priorities from survey respondents in Tehri (dam constructed 20 years ago) and Arunachal Pradesh (dam under development), North India. Each rectangle represents a specific UN SDG, with its size proportional to the number of respondents who ranked it among their top three priorities. The treemap visually highlights the most and least prioritized SDGs in each region, providing insights into regional differences in development priorities.

By structuring the data in tidy format (one row per ranking by one respondent), the dataset enables flexible analysis across multiple dimensions of respondent characteristics and preferences.

2 Semantic data pipeline methodology

The core methodological innovation of the giveadam project is a semantic layering approach that reconceptualizes data transformation pipelines for research contexts. This methodology addresses a fundamental limitation in current data engineering practices: the tension between technical efficiency and research interpretability within the modern data stack paradigm [3].

2.1 The semantic layering paradigm

Traditional data engineering prioritizes performance optimization and code maintainability, often at the expense of conceptual clarity for non-technical stakeholders. The semantic layering methodology inverts this priority structure, organizing transformations around research concepts rather than technical convenience. This approach ensures that:

1. **Research context is preserved** throughout the transformation pipeline
2. **Harmonization decisions are explicit** and auditable by domain experts
3. **Stakeholder communication** uses conceptual rather than technical terminology
4. **Methodological transparency** is maintained without sacrificing automation

2.2 Repository architecture supporting semantic design

The giveadam repository is organized into distinct functional areas:

- **raw_data/** — Original survey data files as provided by Garima Gupta
- **data/** — Published, analysis-ready datasets (pipeline outputs)
- **dbt_project/** — Data transformation pipeline with semantic layering
- **observability/** — Automated methodology documentation and quality reports
- **scripts/** — Data preparation and analysis scripts (R, Python)
- **vis-scripts/** — Visualization generation scripts

2.3 Core innovation: Semantic transformation architecture

The central methodological contribution lies in reimagining data transformation layers as conceptual research stages rather than technical processing steps. Standard dbt implementations use staging → intermediate → marts layers optimized for engineering workflows [2]. The semantic layering methodology introduces a fundamentally different paradigm:

- **source_base/** — Raw data integrity and initial cleaning
- **source_entities/** — Context-preserving regional data entities
- **semantic/** — Explicit cross-source harmonization with documented assumptions
- **analytic/** — Research-question-specific datasets ready for analysis

2.3.1 Why semantic layering transforms research data management

Each semantic layer addresses specific research methodology requirements:

1. **Context preservation** — Source entities maintain regional survey characteristics, preventing premature harmonization that obscures methodological differences
2. **Transparent harmonization** — Semantic models explicitly document how different data sources are reconciled (e.g., SDG labeling differences between regions)
3. **Assumption visibility** — Every transformation decision is captured and testable, enabling methodological scrutiny
4. **Stakeholder accessibility** — Layer terminology reflects research concepts (semantic, analytic) rather than engineering jargon (staging, marts)
5. **Iterative extensibility** — New research questions can be addressed by extending the analytic layer without disrupting upstream logic

This approach resolves the fundamental tension between automated data processing and research transparency, enabling both computational efficiency and methodological rigor.

2.4 Automated observability for research transparency

The observability framework generates this methodology documentation automatically from pipeline artifacts, ensuring that:

- Documentation remains synchronized with actual transformations
- All data quality assumptions are explicitly validated and reported
- Research methodology is fully reproducible from code
- Stakeholders can verify data processing decisions without technical expertise

The complete automated documentation process includes both methodology generation and data publication. The validation tables in this document are generated through the automated process detailed in Section 5.3. Additionally, `scripts/publish_data.R` extracts the final analytic models from the dbt pipeline and exports them as CSV files to the `data/` directory for publication and external use.

The validation tables presented in this document are high-level summaries. Engineers or researchers requiring deeper pipeline investigation can use `dbt docs generate` and `dbt docs serve` for an interactive exploration of the complete data lineage, including detailed model specifications, column-level lineage, and test results.

3 FAIR principles implementation

The giveadam project implements all four FAIR principles [1] through its semantic layering methodology and automated observability framework detailed in Section 2. This section demonstrates how the semantic approach enhances traditional FAIR compliance by making research processes themselves findable, accessible, interoperable, and reusable.

3.1 Findable: Discovery and identification

Rich metadata and identifiers: The datasets are published with comprehensive metadata in `data/README.md`, including detailed column descriptions, data provenance, and research context. Each dataset has unique identifiers (`respondents.csv`, `SDG_rankings.csv`) and is version-controlled in the GitHub repository `giveadam` at `https://github.com/softloud/giveadam`.

Semantic documentation: Unlike standard data repositories, the semantic layering approach provides findable documentation at multiple conceptual levels. Researchers can locate relevant transformations by research concept (semantic models in Table 1) rather than technical implementation details. The four-layer architecture shown in Figure 3 enables discovery through conceptual navigation.

Automated cataloging: The observability framework generates searchable metadata from pipeline artifacts (Table 1), ensuring that all data transformations and quality checks are discoverable through the automated documentation system detailed in Section 5.3.

3.2 Accessible: Retrieval and usability

Open access and standard protocols: Data are published in universally accessible CSV format [24] without authentication barriers. The complete repository is openly available via HTTPS and Git protocols [13], supporting both web browser access and programmatic retrieval through GitHub [14].

Human and machine readable: Column names follow interpretable conventions prioritizing domain understanding over technical convenience. The semantic layer architecture (Figure 3) ensures that data structure reflects research logic rather than processing efficiency.

Multiple access modalities: Researchers can access data through multiple pathways: direct CSV download, Git repository cloning [13], or programmatic URL access in R [9] or Python [8]. The interactive dbt documentation (`dbt docs serve`) [5] provides a web-based interface for exploring complete data lineage.

3.2.1 SDG rankings dataset

This dataset contains respondent rankings enriched with demographic metadata:

- **id_respondent:** Unique identifier for each respondent
- **rank:** Priority ranking (1=highest, 2=medium, 3=lowest priority)
- **sdg_number:** UN SDG number (1-17)
- **sdg_label:** Full name of the Sustainable Development Goal
- **age:** Age of respondent in years
- **gender:** Self-reported gender

- **displacement_status:** Dam impact classification
- **region:** Survey location (tehri, arunachal)

By enriching the responses with respondent metadata, we can analyse responses in the context of respondent demographics and characteristics. For example, in Figure 2, we can see the distribution of top 3 UN SDG priorities by gender.

3.2.2 Respondents dataset

- **id_respondent:** Unique identifier for each respondent
- **age:** Age of respondent in years
- **gender:** Self-reported gender (Male, Female, Prefer not to say)
- **displacement_status:** Impact classification related to dam construction
- **region:** Survey location (tehri, arunachal)

3.3 Interoperable: Integration and exchange

Standard data formats and vocabularies: Data are published in CSV format [24] using UTF-8 encoding with standardized missing value representation (NA). Column naming follows consistent conventions across datasets, enabling seamless joining and integration across the semantic layers shown in Figure 3.

Semantic harmonization documentation: The semantic layer explicitly addresses interoperability challenges by documenting how disparate data sources are harmonized. For example, SDG labeling differences between regional surveys are reconciled with full documentation of mapping decisions in the semantic models (Table 1), enabling other researchers to understand and adapt the harmonization logic.

Modular pipeline architecture: The dbt project structure (Section 2.2) separates concerns across semantic layers, enabling selective reuse of transformation logic. Researchers can adopt the source entity patterns for context preservation while modifying semantic harmonization for different research domains. Each semantic layer (Table 1) implements specific interoperability functions that can be independently understood and modified.

Cross-tool compatibility: Tidy data principles ensure compatibility across analytical software. The semantic layer structure provides conceptual interoperability—researchers can understand and adapt the methodology regardless of their technical implementation preferences.

3.4 Reusable: Extension and adaptation

Comprehensive provenance and documentation: Complete methodology documentation enables confident reuse across research contexts. The automated observability system (Section 5.3) ensures that documentation evolves with implementation, preventing methodology drift that undermines reusability.

Extensible semantic framework: The semantic layering methodology provides a reusable framework beyond this specific dataset. The four-layer architecture (source base → source entities → semantic → analytic) demonstrated in Figure 3 can be adapted for any multi-source research data integration challenge.

Quality assurance infrastructure: The validation framework (Table 2) provides reusable patterns for data quality verification across research contexts. These automated tests ensure that reused components maintain data integrity standards.

Licensing and attribution: Data are licensed under Creative Commons Attribution 4.0 International (CC BY 4.0) [18], enabling broad reuse with appropriate attribution. The license covers both datasets and methodology, encouraging adaptation of the semantic layering approach.

Technical infrastructure reusability: The repository can be forked and the dbt pipeline extended for new data sources or research questions. The modular structure supports iterative development—researchers can extend the analytic layer for new questions without modifying upstream transformations.

Methodological transferability: The semantic layering approach addresses fundamental challenges in research data management that extend beyond this specific domain. The methodology's emphasis on context preservation, transparent harmonization, and stakeholder communication applies to any research requiring multi-source data integration and methodological transparency.

4 Data lineage

In `raw_data/` the original two survey datasets are found. To aggregate across the `.csv` [24] and Kobo tool [19] data export in `.xlsx` format, a data transformation pipeline was implemented using the Data Build Tool (dbt) [5, 6] in the `dbt_project/` directory (detailed in Section 2.2). The dbt project consists of SQL [22] scripts, tests, and documentation that transform the raw data into a tidy format suitable for analysis. Identifying which columns to extract from the Kobo export was nontrivial; `scripts/tehri-cols.R` contains the R script [9, 10] used to identify the relevant columns, and a markdown table [23] identifying the columns associated with survey questions can be found in `figure_and_tables/tehri_columns.md`.

Data quality assurance includes automated identification and exclusion of test data. Two test rows were identified and excluded from the Tehri dataset due to substantial missingness and clear test data indicators (documented in `dbt_project/analyses/tehri_rows_ex` This exclusion process reflects the methodological principles outlined in Section 2.

Figure 3 shows the directed acyclic graph (DAG) for the Data Build Tool (dbt) project `dbt_project/` that produced the .csv data found in `data/`. The DAG illustrates the relationships and dependencies between various data models, tests, and sources within the project. Each node represents a table produced during transformation, while the edges indicate the flow of data and dependencies among them. This visualization demonstrates the four-layer semantic architecture described in Section 5.1 and shows how data is transformed and validated throughout the pipeline, ensuring data integrity and reliability.

Table 1 lists the dbt models created by the `dbt_project/`, ordered by semantic layer. These models implement the semantic extension of dbt's standard layering approach detailed in Section 2.3. The validation framework supporting these models is documented in Table 2.

Table 1: Data Build Tool (DBT) models created by dbt_project, ordered by observability layer.

Model	Description
Analytic Models	
ana_respondents	NA
ana_top3	NA
Semantic Models	
sem_respondents	NA
sem_sdg_labels	NA
sem_top3	NA
Source Entity Models	
se_respondents_arunachal	Each row represents a respondent from Arunachal Pradesh.
se_respondents_tehri	Each row represents a respondent from Tehri.
se_top3_arunachal	Each row represents a respondent's ranking of their top 3 UN Sustainable Development Goals (SDGs) for Arunachal Pradesh, India.
se_top3_tehri	Each row represents a respondent's ranking of their top 3 UN Sustainable Development Goals (SDGs) for Tehri, India.
Base Models	
base_arunachal	Raw data from survey in Arunachal Pradesh, collected by Dr Garima Gupta. Each row represents the survey responses of a single respondent.
base_tehri	Raw data from survey in Tehri, collected by Dr Garima Gupta. Each row represents the survey responses of a single respondent.

5 Semantic layer implementation and observability

The observability framework operationalizes the semantic layering methodology through automated monitoring and validation at each conceptual stage. This implementation demonstrates how semantic design principles translate into technical infrastructure while maintaining research transparency.

5.1 Semantic layers as methodological framework

The four-layer semantic architecture serves as both a conceptual framework and technical implementation. Each layer embodies specific research methodology principles:

- **Source Base Models:** Embody the principle of data integrity preservation. These foundational models maintain fidelity to original data sources while implementing only essential cleaning operations. By materializing as seeds, they create an immutable record of processed raw data, enabling complete methodological auditing.
- **Source Entities:** Operationalize the context preservation principle. Each regional dataset maintains its original structure and labeling conventions, preventing premature harmonization. This layer enables validation of regional data characteristics and supports comparative analysis of data collection methodologies.
- **Semantic Models:** Implement the transparent harmonization principle. This layer explicitly documents how cross-source differences are resolved, such as reconciling disparate SDG labeling systems. All harmonization logic is encoded in SQL with accompanying documentation, making methodological decisions auditable and modifiable.

- **Analytic Models:** Realize the research-readiness principle. These models combine harmonized data with enriched metadata to directly support specific research questions. By materializing as tables, they optimize performance for iterative analysis while maintaining full lineage to upstream decisions.

5.1.1 Methodological advantages of semantic layering

This semantic architecture delivers specific methodological benefits that standard data engineering approaches cannot provide:

1. **Assumption explicitness:** Every harmonization decision is documented and testable
2. **Context preservation:** Regional and methodological differences are maintained until explicitly resolved
3. **Stakeholder communication:** Layer names and logic reflect research concepts rather than technical implementation
4. **Methodological auditability:** Complete transformation lineage enables scholarly review and replication
5. **Iterative extensibility:** New research directions can be accommodated without fundamental restructuring

5.2 Interactive data exploration

The validation tables presented in this document provide high-level summaries of data quality and pipeline structure. For comprehensive pipeline investigation, stakeholders can access the full interactive documentation using:

```
cd dbt_project/
dbt docs generate
dbt docs serve
```

This provides an interactive web interface showing detailed model specifications, column-level lineage, test results, and complete data flow visualization.

5.3 Observability table generation

The validation tables presented in this document (Tables 1 and 2) are automatically generated from dbt artifacts to ensure methodology documentation remains synchronized with the actual pipeline implementation.

5.3.1 Automated metadata extraction

The observability tables are generated through the following automated process:

1. **dbt artifacts generation** — Running `dbt build` produces `manifest.json` and `run_results.json` containing complete pipeline metadata
2. **Python extraction** — `create-obs-tables/get-obs-dat.py` extracts model descriptions and test results from JSON artifacts
3. **R formatting** — `create-obs-tables/obs-table.R` formats extracted data into LaTeX tables
4. **Document compilation** — LaTeX tables are included in this methodology document via `\input` commands

This automation ensures that any changes to model descriptions, test specifications, or pipeline structure are immediately reflected in the methodology documentation, maintaining complete transparency between implementation and documentation.

5.3.2 Model materializations

The semantic layers employ different dbt materializations optimized for their function:

- **Source base models** — Materialized as `seeds` (CSV files loaded directly into DuckDB)
- **Source entity models** — Materialized as `views` to preserve disk space while maintaining fast access for downstream models
- **Semantic models** — Materialized as `views` to enable flexible harmonization logic without storage overhead
- **Analytic models** — Materialized as `tables` to optimize query performance for research analysis and data export

The progression from views to tables reflects the increasing stability and query frequency of models as they approach the analytical layer. Source and semantic layers prioritize flexibility and maintainability through views, while analytic models prioritize performance through table materialization for repeated research queries.

5.4 Data validation

Table 2 lists the dbt tests applied to `dbt_project/` transformations, ordered by semantic layer. These tests document the data quality assumptions validated at each transformation step in the lineage (Figure 3). The tests ensure data integrity across the semantic pipeline and provide automated quality assurance for research reproducibility.

Table 2: Data Build Tool (DBT) tests applied to dbt_project models, ordered by observability layer.

Model	Test	Columns	Arguments	Result
Analytic Model Tests				
ana_respondents	unique_combination_of_columns	id_respondent	NA	success
Semantic Model Tests				
Source Entity Tests				
se_respondents_arunachal	accepted_values	gender	Male__Female__Prefer_not_to_say	success
se_respondents_arunachal	not_null	id_respondent	NA	success
se_respondents_arunachal	unique	id_respondent	NA	success
se_respondents_arunachal	unique_combination_of_columns	id_respondent	NA	success
se_respondents_tehri	accepted_values	gender	Male__Female__Prefer_not_to_say	success
se_respondents_tehri	not_null	id_respondent	NA	success
se_respondents_tehri	unique	id_respondent	NA	success
se_respondents_tehri	unique_combination_of_columns	id_respondent	NA	success
se_top3_arunachal	accepted_values	rank	1__2__3	success
se_top3_arunachal	not_null	id_respondent	NA	success
se_top3_arunachal	unique_combination_of_columns	id_respondent	rank	success
se_top3_tehri	accepted_values	rank	1__2__3	success
se_top3_tehri	not_null	id_respondent	NA	success
se_top3_tehri	unique_combination_of_columns	id_respondent	rank	success
Base Model Tests				
base_arunachal	unique_combination_of_columns	id_respondent	NA	success
base_arunachal	unique_combination_of_columns	respondents	NA	success
base_tehri	not_null	id_respondent	NA	success
base_tehri	unique	id_respondent	NA	success
base_tehri	unique_combination_of_columns	id	NA	success
base_tehri	unique_combination_of_columns	id_respondent	NA	success

6 Use of NLP tools

The giveadam project architecture, semantic data pipeline design, and all data transformations were conceived and implemented by Charles T. Gray. GitHub Copilot [17] was used for documentation editing and selected development operations tasks (supporting JSON manifest parsing scripts), but did not contribute to the fundamental design decisions, data modeling approach, or core analytical implementations. All code development, methodological innovations, and research insights represent the original work of the author.

References

[1] Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., ... & Mons, B. (2016). The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data*, 3(1), 1-9.

[2] dbt Labs. (2024). How we structure our dbt projects. *dbt Documentation*. Retrieved from `https://docs.getdbt.com/best-practices/how-we-structure/1-guide-overview`

[3] Airbyte. (2023). What is the modern data stack and why should you care? *Airbyte Blog*. Retrieved from `https://airbyte.com/blog/modern-data-stack`

[4] Kimball, R., & Ross, M. (2013). *The data warehouse toolkit: the definitive guide to dimensional modeling*. John Wiley & Sons.

[5] dbt Labs. (2024). dbt Core: Transform data in your warehouse. Version 1.9+. Retrieved from `https://github.com/dbt-labs/dbt-core`

[6] MotherDuck & dbt Labs. (2024). dbt-duckdb: dbt adapter for DuckDB. Version 1.9.6+. Retrieved from `https://github.com/duckdb/dbt-duckdb`

[7] Raasveldt, M., & Mühleisen, H. (2019). DuckDB: an embeddable analytical database. In *Proceedings of the 2019 International Conference on Management of Data* (pp. 1981-1984).

[8] Van Rossum, G., & Drake, F. L. (2009). *Python 3 Reference Manual.* CreateSpace.

[9] R Core Team. (2024). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. Retrieved from https://www.R-project.org/

[10] Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., ... & Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, 4(43), 1686.

[11] McKinney, W. (2010). Data structures for statistical computing in Python. In S. van der Walt & J. Millman (Eds.), *Proceedings of the 9th Python in Science Conference* (pp. 56-61).

[12] Astral. (2024). uv: An extremely fast Python package installer and resolver. Retrieved from https://github.com/astral-sh/uv

[13] Chacon, S., & Straub, B. (2014). *Pro Git.* Apress.

[14] GitHub, Inc. (2024). GitHub: Where the world builds software. Retrieved from https://github.com

[15] Lamport, L. (1994). *LaTeX: a document preparation system: user's guide and reference manual.* Addison-Wesley.

[16] Microsoft Corporation. (2024). Visual Studio Code. Retrieved from https://code.visualstudio.com/

[17] GitHub, Inc. (2024). GitHub Copilot. Retrieved from https://github.com/features/copilot

[18] Creative Commons. (2024). Creative Commons Attribution 4.0 International License. Retrieved from https://creativecommons.org/licenses/by/4.0/

[19] Harvard Humanitarian Initiative. (2024). KoBo Toolbox: Data collection tools for challenging environments. Retrieved from https://www.kobotoolbox.org/

[20] Bray, T. (2017). The JavaScript Object Notation (JSON) Data Interchange Format. RFC 8259. Retrieved from https://tools.ietf.org/html/rfc8259

[21] Ben-Kiki, O., Evans, C., & döt Net, I. (2009). YAML Ain't Markup Language (YAML™) Version 1.2. Retrieved from https://yaml.org/spec/1.2/spec.html

[22] Chamberlin, D., & Boyce, R. (1974). SEQUEL: A structured English query language. In *Proceedings of the 1974 ACM SIGFIDET workshop on Data description, access and control* (pp. 249-264).

[23] Gruber, J. (2004). Markdown. Retrieved from https://daringfireball.net/projects/markdown/

[24] Shafranovich, Y. (2005). Common Format and MIME Type for Comma-Separated Values (CSV) Files. RFC 4180. Retrieved from https://tools.ietf.org/html/rfc4180

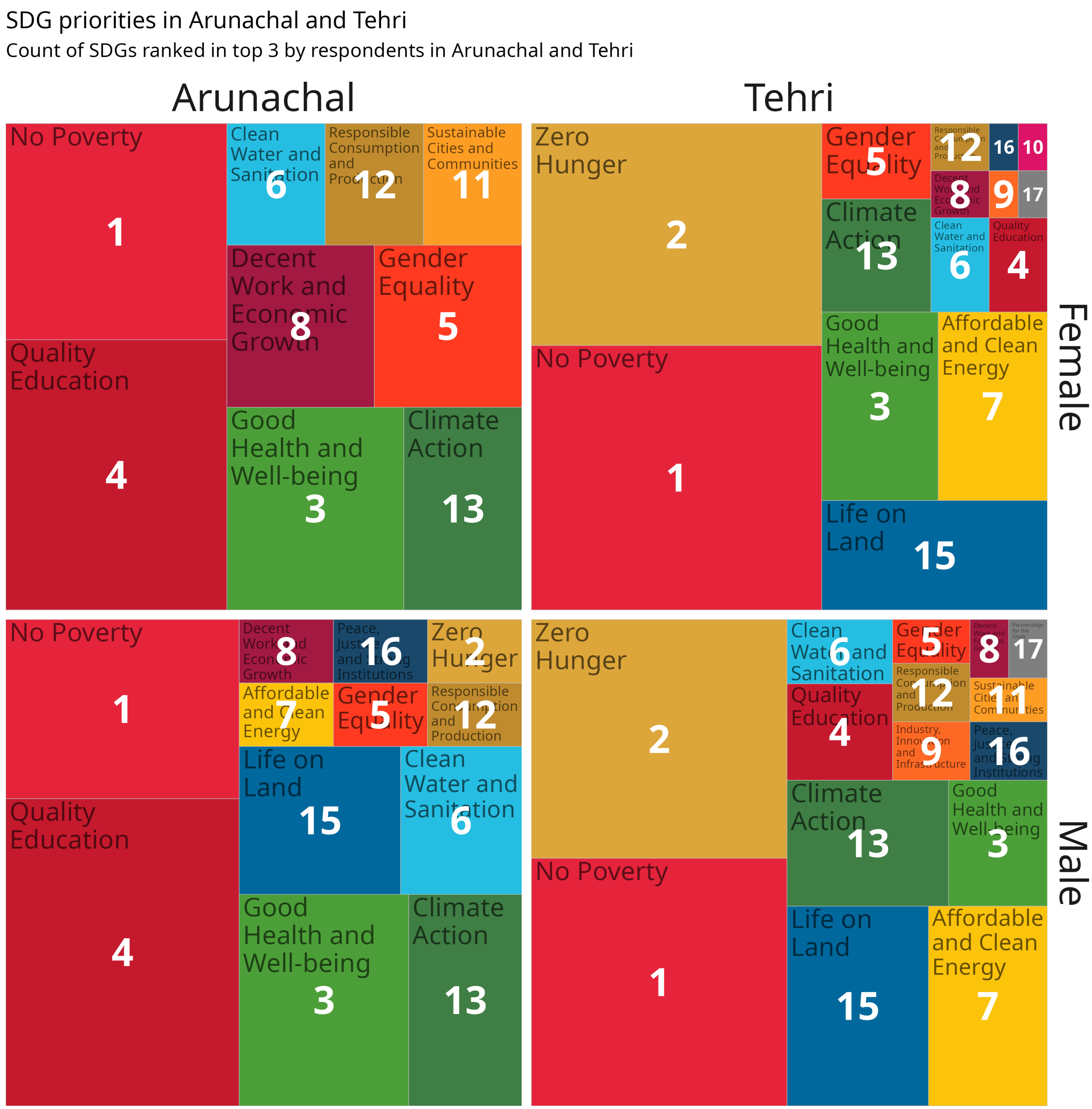

Figure 2: Treemap of top 3 UN SDG priorities from survey respondents in Tehri and Arunachal Pradesh, North India, by gender. Each rectangle represents a specific UN SDG, with its size proportional to the number of respondents of a particular gender who ranked it among their top three priorities. The treemap visually highlights the most and least prioritized SDGs in each region, providing insights into gender differences in development priorities.

Figure 3: Data Build Tool (dbt) directed acyclic graph (DAG) showing the semantic layer structure from source entities through analytic models.