UNIVERSITÄT AUGSBURG
Universitätsbibliothek

OPEN ACCESS
OPUS AUGSBURG

Hamilton- interactive ontology learning

Lukas Laskowski, Felix Draxler, Michael Hladik, Fabian Panse, Jan Portisch, Padhraic Smyth, Felix Naumann

Angaben zur Veröffentlichung / Publication details:

Laskowski, Lukas, Felix Draxler, Michael Hladik, Fabian Panse, Jan Portisch, Padhraic Smyth, and Felix Naumann. 2026. "Hamilton- interactive ontology learning." *Proceedings of the VLDB Endowment* 19 (12): 4534–37. https://doi.org/10.14778/3827998.3828059.

Nutzungsbedingungen / Terms of use: **CC BY-NC-ND 4.0**

Dieses Dokument wird unter folgenden Bedingungen zur Verfügung gestellt: / This document is made available under these conditions:
CC-BY-NC-ND 4.0: Creative Commons: Namensnennung - Nicht kommerziell - Keine Bearbeitung
Weitere Informationen finden Sie unter: / For more information see:
https://creativecommons.org/licenses/by-nc-nd/4.0/deed.de



Hamilton– Interactive Ontology Learning

Lukas Laskowski
lukas.laskowski@hpi.de
Hasso Plattner Institute
University of Potsdam, Germany

Felix Draxler
fdraxler@uci.edu
University of California, Irvine
Irvine, US

Michael Hladik
michael.hladik@sap.com
SAP SE
Walldorf, Germany

Fabian Panse
fabian.panse@uni-a.de
University of Augsburg
Augsburg, Germany

Jan Portisch
jan.portisch@sap.com
SAP SE
Walldorf, Germany

Padhraic Smyth
smyth@ics.uci.edu
University of California, Irvine
Irvine, US

Felix Naumann
felix.naumann@hpi.de
Hasso Plattner Institute
University of Potsdam, Germany

ABSTRACT

Relational databases store large amounts of structured data, but their schemas often do not directly correspond to semantic data models such as ontologies. Transforming database schemas into ontologies is therefore an important step for enabling ontology-based data access and data integration. However, this task is challenging because relational schemas are designed for data organization and integrity, not for representing domain semantics, and modeling may be context-dependent. In this demonstration, we present Hamilton, an interactive application for ontology learning from relational databases that combines automated modeling suggestions with expert-driven refinement. The demonstration showcases the full ontology learning workflow and illustrates how automatic modeling and human expertise can be effectively combined to construct high-quality ontologies from relational databases.

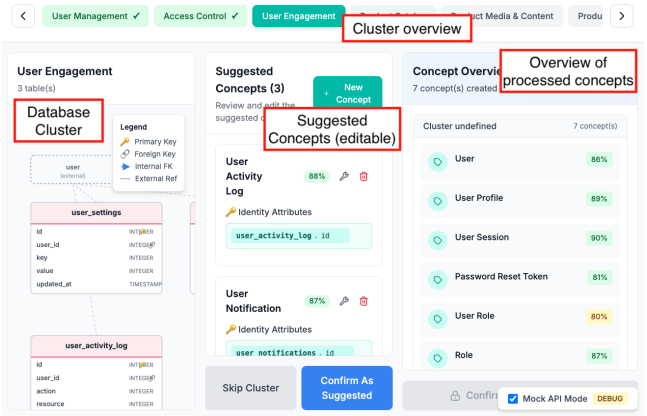


Figure 1: Screenshot of the Hamilton-application.

PVLDB Reference Format:
Lukas Laskowski, Felix Draxler, Michael Hladik, Fabian Panse, Jan Portisch, Padhraic Smyth, and Felix Naumann. Hamilton– Interactive Ontology Learning. PVLDB, 19(12): 4534 - 4537, 2026.
doi:10.14778/3827998.3828059

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/HPI-Information-Systems/hamilton.

1 ONTOLOGY LEARNING

The schemata of relational databases often do not directly correspond to semantic data models such as ontologies. Transforming those schemata into ontologies is therefore a key step for enabling ontology-based data access (OBDA) and data integration. However, this transformation is non-trivial due to an impedance mismatch between relational structures and semantic representations.

Existing approaches to ontology learning from relational databases can be broadly categorized into rule-based and LLM-based systems. Rule-based systems apply structural heuristics and generalize well, but often fail to capture the underlying semantics. LLM-based approaches provide stronger semantic modeling capabilities, but struggle to generalize to large schemata and typically lack mechanisms for incorporating expert input.

In practice, ontology modeling is inherently context-dependent: multiple modeling choices may be valid, and selecting an appropriate representation often requires domain knowledge and human judgment. Consequently, ontology learning systems should support expert-guided refinement of automatically generated suggestions.

To address these challenges, we present Hamilton [6], an interactive ontology learning system that combines structural analysis with semantic modeling and enables human-in-the-loop refinement. In this demonstration, we showcase the full workflow of Hamilton, including schema decomposition, ontology suggestion, uncertainty estimation, and interactive editing.

This work is licensed under the Creative Commons BY-NC-ND 4.0 International License. Visit https://creativecommons.org/licenses/by-nc-nd/4.0/ to view a copy of this license. For any use beyond those covered by this license, obtain permission by emailing info@vldb.org. Copyright is held by the owner/author(s). Publication rights licensed to the VLDB Endowment.
Proceedings of the VLDB Endowment, Vol. 19, No. 12 ISSN 2150-8097.
doi:10.14778/3827998.3828059

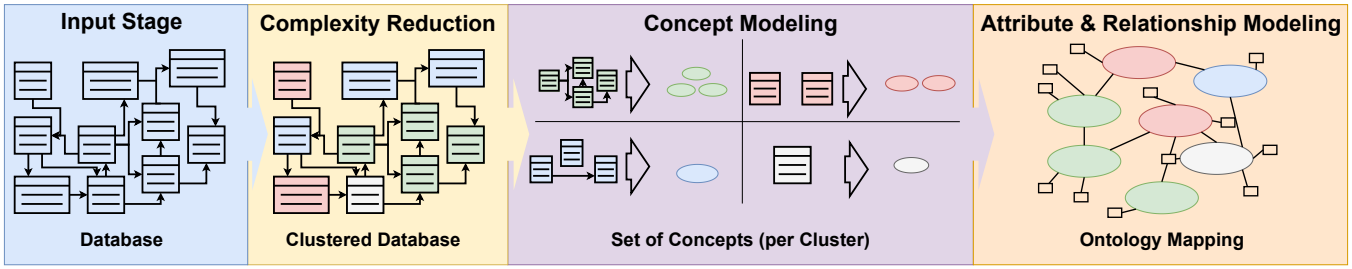


Figure 2: Workflow of Hamilton.

2 THE HAMILTON APPLICATION

Hamilton is a multi-stage application that guides users through the ontology learning process. At each stage, users can refer to Hamilton's automatic modeling suggestions for learning specific aspects of the ontology and refine them using intuitive editing tools. Figure 2 illustrates the overall workflow. The system's final output is a database-ontology mapping using the mapping language D2RQ [1]. In the following, we introduce each stage in more detail.

2.1 Complexity reduction

After uploading the database to Hamilton, the first step is to reduce structural and cognitive complexity. Database schemata are usually very large, consisting of many tables and constraints. Even databases of medium-sized web applications, such as that of the e-commerce platform Magento, comprise more than 200 tables [5]. Modeling such schemata as a whole exceeds the manageable scope. Ontology learning requires considering tables and constraints alongside other tables and deferring modeling decisions based on them. When presented globally, these modeling decisions become cognitively demanding and can cause modeling errors.

To overcome this challenge, Hamilton decomposes the database into topically coherent clusters prior to the actual ontology learning process. To perform this task, we developed Schuyler [8] – a self-supervised database table clustering system. It takes a database as input and returns clusters of topically related tables, without requiring any labels or training data. Schuyler extracts triplets of database tables by applying simple heuristics. These triplets consist of an anchor table, a table that supposedly belongs to the same cluster as the anchor table, and a table that supposedly belongs to another cluster. Then, the system finetunes an embedding model using triplet loss and the extracted triplets. Hamilton visually shows the clustered database schema to the expert. In this "clustering"-view, the user can explore the clustering and obtain a comprehensive overview of the topics covered by the database. Then, the expert can further improve this clustering by moving tables across clusters using drag&drop. By reducing complexity upfront, Hamilton enables the expert to work on manageable sub-schemata that consist of topically related tables.

2.2 Concept modeling

After reducing schema complexity, the Hamilton-application navigates the expert to the concept modeling stage. In this stage, the system iterates over the sub-schemata of the database clusters and

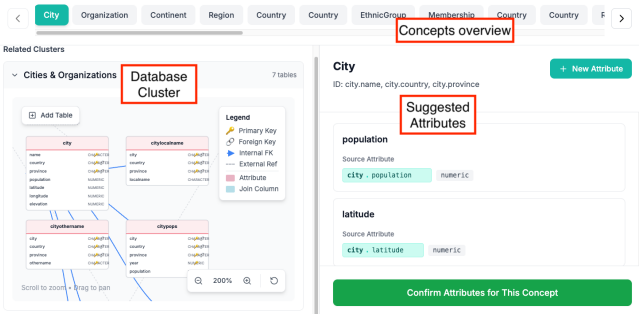


Figure 3: Attribute modeling.

suggests ontology concepts based on the provided tables and constraints. Figure 1 depicts the concept modeling view. The top of the screen shows which cluster is currently being processed, which clusters have already been processed, and which clusters are still pending ("Cluster overview" in Figure 1). On the left is an overview of the tables that belong to the cluster currently being edited. The center of the image shows the core of this view: this is where the suggested concepts are displayed, and the expert can edit them. For example, they can add new identifying attributes, remove or rename concepts, and add joins to combine data from different tables within the same concept. Whenever references to the database are needed, such as when adding a new identifying attribute, the user can simply click the respective attribute in the database cluster viewer. The database viewer on the left always highlights which tables are involved in which concept. On the right is a cross-cluster overview. To avoid creating the same concept multiple times, all concepts that have already been created are displayed here.

The automatic concept suggestions of Hamilton are generated using a reasoning-based fine-tuned language model [6]. For this purpose, we trained the model on a curated collection of database/ontology pairs covering diverse and challenging modeling scenarios. Additionally, we described each modeling process in a chain-of-thought manner. Then, we finetuned a reasoning-based LLM, Qwen 14B, on the ontology learning task. This model can now be used to generalize to new domains and databases.

2.3 Attribute modeling

After modeling and confirming all concepts, the expert is directed to the attribute modeling stage. In this stage, the expert iterates over the previously modeled concepts and can edit the suggested

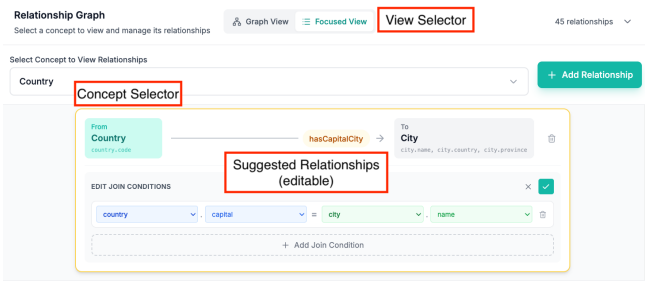


Figure 4: Relationship modeling.

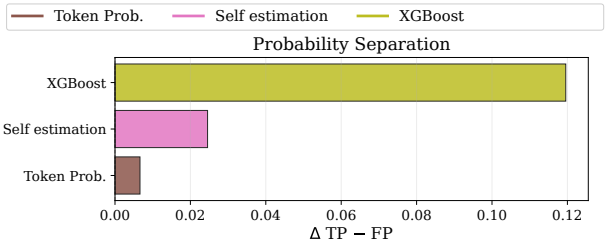


Figure 5: Uncertainty measurement.

attributes for each concept individually. Again, Hamilton suggests attributes and provides a number of tools to edit them. Figure 3 depicts this view of the Hamilton-application. On the left, similar to the concept viewer, are the underlying database clusters, which include tables that the respective concept refers to. The right-hand side visualizes the proposed attributes of this concept. To suggest attributes, Hamilton relies on a hybrid approach of combining LLMs with structural heuristics: An LLM first classifies the tables that the respective concept refers to, based on their role in the database. E.g., entity tables represent entity types while relationship tables connect them. The actual attributes are then deferred for each concept based on the respective table types using rules.

2.4 Relationship modeling

In the final stage, Hamilton uses predefined rules to suggest relationships between the modeled concepts (across all sub-schemata) based on the types and foreign keys of the corresponding tables. Figure 4 shows the view of the Hamilton-application, where users can inspect and revise all relationships for a selected concept.

3 UNCERTAINTY QUANTIFICATION

Ontology modeling decisions are inherently uncertain. Multiple modeling representations may be equally valid for a given schema. The decision of which representation to choose often depends on the context, the human modeling expert, and implicit domain knowledge that is not provided by the database.

Exposing the uncertainty of the suggestions generated by Hamilton is therefore essential for enabling informed human oversight and effective refinement. Hamilton provides an uncertainty estimate for each suggested concept. This estimate allows users to filter and prioritize concepts based on the model's confidence. Concepts with high confidence typically require little or no review, whereas concepts with lower confidence can be inspected and corrected by the expert if necessary.

3.1 Uncertainty quantification methods

We observed that the model's raw token probabilities mostly indicate high confidence (i.e., probabilities close to 1.0) – they do not reliably reflect the correctness of the generated concepts. Therefore, we train a learned uncertainty estimator using XGBoost: we select a subset of database-ontology pairs and execute the finetuned model on this data. For each predicted concept, we extract a set of features that describe the respective modeling task. Based on these features, the model's prediction, and the ground truth labels, we train the uncertainty estimation model. The resulting calibration model predicts an adjusted uncertainty score that reflects the likelihood that a concept prediction is correct. We use the following feature groups:

Relational Complexity. The complexity of the input schema influences the uncertainty of the resulting ontology suggestions. More complex schemata typically lead to more difficult modeling decisions. Therefore, we include structural complexity features, such as the number of tables in the corresponding sub-schema.

Model Uncertainty. We leverage the probabilistic nature of the finetuned LLM to quantify uncertainty. Since the model generates output autoregressively by sampling tokens from a probability distribution, we can extract token-level probabilities $\hat{y}_t$. These probabilities are used as features for the uncertainty calibration model.

Concept Properties. In addition to input complexity and model uncertainty, properties of the generated concepts also influence the uncertainty. We therefore include features such as the number of generated concepts and the token length of each concept definition.

3.2 Evaluation setup

We compare the suggested uncertainty estimation method with two baselines to show the effectiveness of using a calibration model:

Token Probability. As LLMs generate output tokens according to probability distributions, token probabilities can be used as a proxy for model confidence. Yet, as noted above, a large portion of the generated output consists of highly predictable structural tokens, which often have probabilities close to 1.0. To mitigate this issue, we only consider the probabilities of tokens representing ontological content, such as database references.

Self-reported. Another approach to estimating uncertainty is to prompt the model to explicitly output a confidence score alongside its prediction [4]. Although LLMs are not explicitly trained to produce calibrated probabilities, such verbalized confidence scores provide a simple and model-agnostic way to estimate uncertainty.

3.3 Results

Figure 5 compares the three uncertainty quantification methods. The "Probability Separation" chart measures the difference between the uncertainty scores of true positives and false positives. Ideally, this difference should be large and positive, indicating that incorrect predictions are associated with lower confidence. The results show that, while the differences for the token probability approach and the self-estimation approach are only 0.01 and 0.02, respectively, the calibration model approach (XGBoost) shows a larger difference

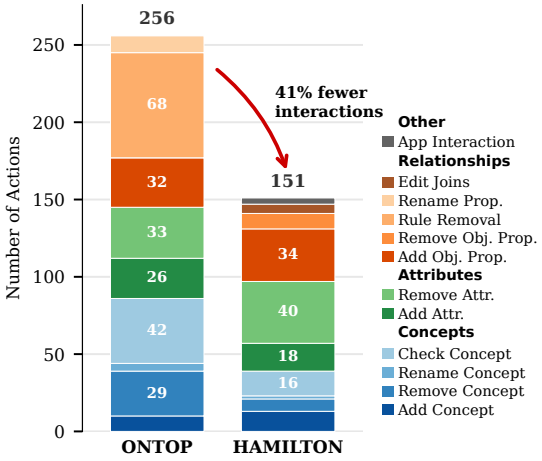


Figure 6: Interaction Comparison of Ontop vs. Hamilton

of 0.12. Although the XGBoost calibration approach achieves the best performance among the evaluated methods, its overall results remain limited. One reason is that ontology modeling decisions are often context-dependent. Multiple modeling choices may be valid for a given schema, and alternative but reasonable interpretations may therefore be counted as incorrect by the evaluation.

4 INTERACTION COMPARISON

To evaluate the practical benefits of Hamilton, we compare it with Ontop [3], the leading ontology learning system combining automatic modeling suggestions with human refinement. Ontop follows a rule-based approach and is integrated into the widely used ontology modeling application Protégé. The plugin enables users to review, adapt, and extend the automatically generated ontology.

We assess the quality of the automatic modeling suggestions produced by both systems using our Burr benchmark suite [7]. The results [6], measured in terms of F1-scores, show that Hamilton outperforms Ontop on average across all categories: concepts (0.54 vs. 0.48), attributes (0.56 vs. 0.47), and relationships (0.47 vs. 0.31). At the same time, these findings emphasize that human refinement remains a crucial step for achieving high-quality ontologies.

To estimate the human effort required to adapt the automatically generated modeling suggestions to the ground-truth ontology, we employ the standardized GOMS framework using the Keystroke-Level Model (KLM) [2]. This approach enables an objective estimation of effort by quantifying the number of user interactions required to complete the task. As shown in Figure 6, Hamilton reduces the total number of required actions from 256 to 151 for the Mondial database [9], corresponding to a reduction of 41%. As a result, users perform fewer manual actions such as adding concepts, attributes, or relationships from scratch. Additionally, the individual actions require fewer steps in Hamilton. For example, while Ontop requires 150 steps to add a concept, Hamilton enables the user to perform this task with only 16 steps [6]. In Ontop, the expert has to write many mapping queries, while Hamilton substantially eases this process through interactive suggestions and integrated database navigation. On average, Hamilton enables a 5.9x speed-up for individual tasks, such as concept modeling.

5 DEMONSTRATION SCENARIOS

In this demonstration, attendees can experience the entire ontology learning workflow of Hamilton using a focused version of the real relational database Mondial [9]. The goal of the demo is to present how Hamilton supports experts in the ontology modeling process through interactive guidance and generated suggestions. To this end, the attendees execute each of Hamilton's stages (complexity reduction, concept modeling, etc.), review the results of each stage, and correct potential modeling errors via simple drag&drop actions.

In this process, attendees will be presented with two special scenarios, which illustrate the difficulty of the ontology learning task and highlight the advantages of Hamilton. In these scenarios, a straightforward transformation of the relational schema suggests a different ontology than the actual ground-truth:

"Missing Attributes": The attendees explore how Hamilton assists in extending ontological concepts with additional attributes that are stored in tables not yet connected to the respective concept. This task is usually work-intensive, as identifying the correct joins and understanding the semantic connections between tables requires detailed schema knowledge and time to write the join queries. Hamilton automatically detects that joins are needed and lets the expert easily create them by interacting with the integrated database table viewer.

"Misclassified Concept": The attendees are presented with a scenario in which Hamilton incorrectly suggests modeling a relationship table as a concept. Using Hamilton's interactive mode, they can correct this error by converting the concept into a relationship and assigning the associated attributes to the two concepts connected by it.

REFERENCES

[1] Christian Bizer and Andy Seaborne. 2004. D2RQ-treating non-RDF databases as virtual RDF graphs. In *Proceedings of the International Semantic Web Conference (ISWC)*, Vol. 2004.

[2] Stuart Card, Thomas Moran, and Allen Newell. 1983. *The Psychology of Human-Computer Interaction* (1 ed.). CRC Press. https://doi.org/10.1201/9780203736166

[3] Óscar Corcho, Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro, and Guohui Xiao. 2017. Ontop: Answering SPARQL queries over relational databases. *Semantic Web* 8, 3 (2017), 471–487. https://doi.org/10.3233/SW-160217

[4] Jiahui Geng, Fengyu Cai, Yuxia Wang, Heinz Koeppl, Preslav Nakov, and Iryna Gurevych. 2024. A Survey of Confidence Estimation and Calibration in Large Language Models. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 6577–6595. https://doi.org/10.18653/v1/2024.naacl-long.366

[5] Inchoo. 2014. Magento MySQL Database Structure. https://inchoo.net/magento-1/magento-mysql-database-structure/ Accessed: 2025-02-12.

[6] Lukas Laskowski, Felix Draxler, Michael Hladik, Fabian Panse, Jan Portisch, Padhraic Smyth, and Felix Naumann. 2026. Hamilton: Learning Ontologies from Relational Databases. *Proceedings of the VLDB Endowment (PVLDB)* (2026). https://github.com/HPI-Information-Systems/hamilton

[7] Lukas Laskowski, Michael Hladik, Jan Portisch, Fabian Panse, and Felix Naumann. 2025. Burr: A Benchmark for Ontology Learning from Relational Databases [Experiments & Analysis]. *Proc. ACM Manag. Data* 3, 6, Article 305 (Dec. 2025), 27 pages. https://doi.org/10.1145/3769770

[8] Lukas Laskowski, Fabian Panse, Michael Hladik, Jan Portisch, and Felix Naumann. 2026. Schuyler: Self-Supervised Clustering of Tables in Relational Databases. *Proceedings of the VLDB Endowment (PVLDB)* 19, 4 (2026), 657–669.

[9] Wolfgang May. 1999. *Information extraction and integration with Florid: The Mondial case study.* Technical Report.