CellCards: Development of a dynamic ontology-derived ETL pipeline for automatic cell information extraction and analysis

Mary Czelusniak[1], Emily Tran[1], Fatima Oudeif[1], Jie Zheng[1], William D. Duncan[2], Alexander D. Diehl[3], Yongqun He[1].

[1] *University of Michigan, Ann Arbor, MI, USA.*
[2] *University of Florida, Gainesville, FL, USA.*
[3] *University at Buffalo, Buffalo, NY, USA.*

Abstract

The CellCards knowledgebase aims to systematically gather, and represent individual cell types. This study presents our development of a dynamic extraction, transformation, and loading (ETL) pipeline designed to automatically populate the CellCards database with a vast array of cells from ontologies, including the Cell Ontology (CL) and Cell Line Ontology (CLO). The CellCards database schema includes five tables, with a key feature being the use of one table to encompass all necessary terms from the ontologies and another table to outline the relationships among these terms. The ETL process is powered by a Python script that embeds SPARQL queries directed at the Ontobee SPARQL endpoint. The final ETL program successfully extracted and loaded over 3,500 cell types from CL and 40,000 cell line entries from CLO into the new CellCards database, including the cell type name, parent cell type, synonyms, anatomical locations, etc. The gene biomarkers of cells were automatically extracted from the Common Coordinate Framework Ontology (CCFO). This enhanced database will be used to update the website and query program, with these updates scheduled for summer 2024.

Keywords

Cell, Ontology, Cell Ontology, Uber-anatomy Ontology, Gene Ontology, Cell Line Ontology, Common Coordinate Framework Ontology, Knowledgebase, ETL (Extract, Transform, Load), Python, SPARQL, Multi-Threading.

1. Introduction

CellCards (https://cellcards.org) is an innovative, ontology-based knowledge system designed to systematically collect and integrate cell type information from a variety of sources about the diverse array of cells of the human body [1]. By doing so, CellCards aims to utilize the hierarchical organization of cellular data, paving the way for a structured approach to understanding cellular functions and relationships.

Ontologies have been widely used to consistently and semantically represent the knowledge about various cell types. The Cell Ontology (CL) [2] and Cell Line Ontology (CLO) [3] are two community-based ontologies for representing various cell types and cell line cells, respectively, which were used as our primary source of data extraction and reference for associating each cell type's annotations and relationships correctly. Other cell-related ontologies utilized include Uberon Anatomic Entity Ontology [4], used primarily for fetching anatomical entities that have relationships with cells from CL or CLO, Common Coordinate Framework Ontology (CCFO) [5] as the primary source of biomarker data, and the HUGO Gene Nomenclature Committee (HGNC) [6] as the source for related biomarkers IDs.

This paper reports our development of an enhanced ETL (Extract, Transform, Load) data pipeline that extracts, transforms, and loads over 3,000 cell types from CL and 35,000 cell lines from

[1] 15th International Conference on Biological and Biomedical Ontology, July 17-19 2024 Enschede, the Netherlands, (*note:* https://sites.google.com/view/cells2024).

EMAIL: czmary@umich.edu (A. 1); etran@umich.edu (A. 2) faoudeif@wccnet.edu (A. 3); jiezhen@med.umich.edu (A. 4) wdduncan@gmail.com (A. 5); addiehl@buffalo.edu (A. 6); yongqunh@med.umich.edu (A. 7);
ORCID: 0009-0007-4123-542X (A.1); 0000-0001-9189-9661 (A.7)

CLO, adding to around 45,000 rows of data across the tables in the CellCards database, utilizing methods such as multi-threading, and REST API requests, programs such as MySQL Workbench to manage the database, different query languages such as SPARQL and SQL to extract and load data, and Python that executes the ETL pipeline.

2. Methods

Data sources: Cell information is extracted from the CL [2] and CLO [3], gene names/labels and its association with each cell type contained within the aforementioned ontologies come from CCFO [5] and its annotation comes from HGNC [6].

Updated design of CellCards MySQL database schema: The new database schema is based on the Genomics Unified Schema (GUS) Schema that was originally developed for the VEuPathDB project (https://github.com/VEuPathDB/GusSchema). Instead of having each type of entity (such as cell, cell line, anatomical entity, gene, etc.) as a table, the GUS schema uses a small number of general tables to represent different types of entities.

ETL development: The ETL data extraction method automates the data insertion process into the various tables within the CellCards MySQL database. A Python script was developed to streamline this workflow. The ETL process can be broken down into three main steps: (1) **Extract:** The ETL program embeds specific SPARQL query scripts to extract data from the ontologies. The most efficient scripts were integrated into the Python script through use of the SPARQLWrapper library (https://github.com/RDFLib/sparqlwrapper). This step involves taking an input cell ID (e.g., CL_0000653) and its relevant ontology (such as CL or CLO) and appending them to an Internationalized Resource Identifier (IRI) prefix to perform the SPARQL query appropriately. (2) **Transform:** Once the data is extracted, the next step is to transform it into the desired format. This might involve operations such as trimming whitespace, adding semicolons, or compiling multiple values into a single list if storage in a single column is desired. (3) **Load:** This final step involves all the operations required to load transformed data into its appropriate table in the CellCards database.

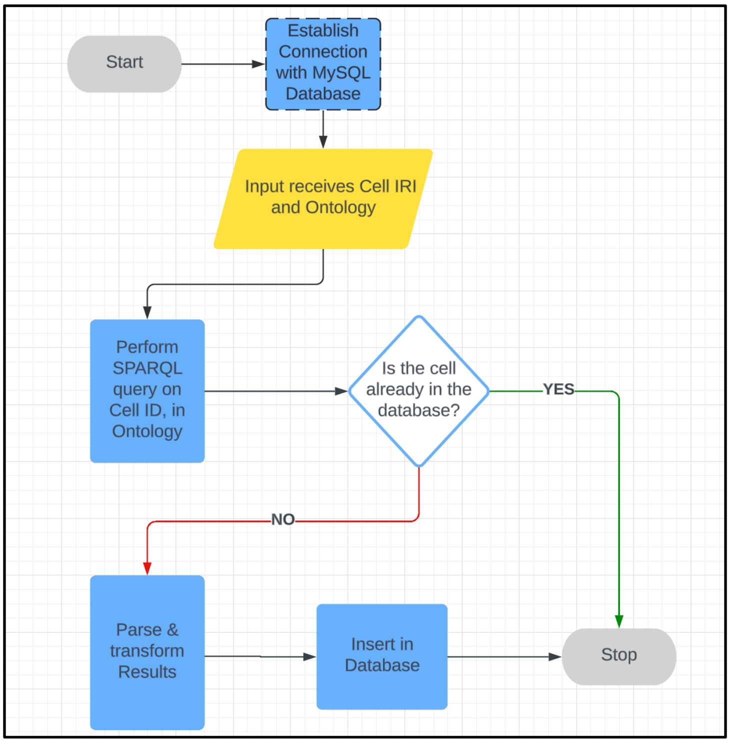
Start
Establish Connection with MySQL Database
Input receives Cell IRI and Ontology
Perform SPARQL query on Cell ID, in Ontology
Is the cell already in the database?
YES
NO
Parse & transform Results
Insert in Database
Stop

Figure 1: An ETL flowchart for querying a specific cell type in an ontology, such as CL.

A slightly different approach was taken to obtain the biomarker information during the extraction and transformation steps. Firstly, a SPARQL query on the CCFO ontology was executed to get each cell's biomarker IDs (HGNC IDs) and names. Then, three data structures were utilized to efficiently associate each biomarker with its corresponding cells: a set that contained all distinct HGNC IDs returned from the SPARQL query and two hashmaps. The first hashmap mapped a CL IRI to its key in the t_cells table, allowing a quick lookup of cell information. The second hashmap mapped an HGNC ID to a list of strings, with the gene's label as its first entry. This list of strings was then populated by the t_cells IDs of the cells associated with this biomarker as the results from the SPARQL query were parsed, where each cell's t_cells ID was obtained through its association to its CL IRI in the first hashmap mentioned, ensuring that each cell type's biomarker data was accurately linked to its database entry.

Once the initial associations were made, the set of HGNC IDs was used to perform a request on the HGNC REST API to obtain the gene's NCBI Entrez Gene [7] and PR [8] IDs. Since a set was used as its container, we were guaranteed non-repeated HGNC IDs, meaning no unnecessary requests would be made. Four threads were then utilized to perform this task, where each executed the request to the REST API with a quarter of the HGNC set while respecting the API's limits of 10 requests per second. Implementing multi-threading was crucial for run-time efficiency, as without multiple threads, this process would take about 30 minutes to complete compared to the current 2-minute total run time, significantly speeding up the data extraction process and ensuring that the large volume of biomarker data was processed promptly without overloading the API. The Ontology of Genes and Genomes (OGG) identifiers were calculated based on its associated NCBI Entrez Gene IDs and the OGG ID generation rule [9].

After this process was done, the results from the threads were combined into another hashmap, where the HGNC ID mapped to a list of size 3, containing, in order, its PR ID, NCBI Gene ID, and description, which was then used to perform the insertion into the t_gene_proteins table in a similar fashion as the preceding tables.

Testing and evaluation: Our team utilized a manually generated podocyte CellCard to determine the specific information we needed to retrieve for each cell type. By closely examining the relationships and attributes listed on the podocyte CellCard, we could refine our data extraction and transformation processes to ensure we captured all relevant details. Several different ways of inputting data into the script were also developed, such as command line options for reading in CSV files (file input redirection) or creating a CSV file with the results from the queries.

3. Results

3.1 CellCards database schema design

Our updated CellCards MySQL database includes 5 populated tables (Figure 2). The table t_cells contains general information about the cell type and an ID that links a cell type to its biomarkers in the t_gene_proteins table, t_synonyms lists alternative names and identifiers, t_ontology_term catalogs its ontology data, t_ontology_term_relation outlines the relationships within its original ontology, and t_gene_proteins records its associated biomarkers and their identifiers across NCBI, HGNC, OGG and PR.

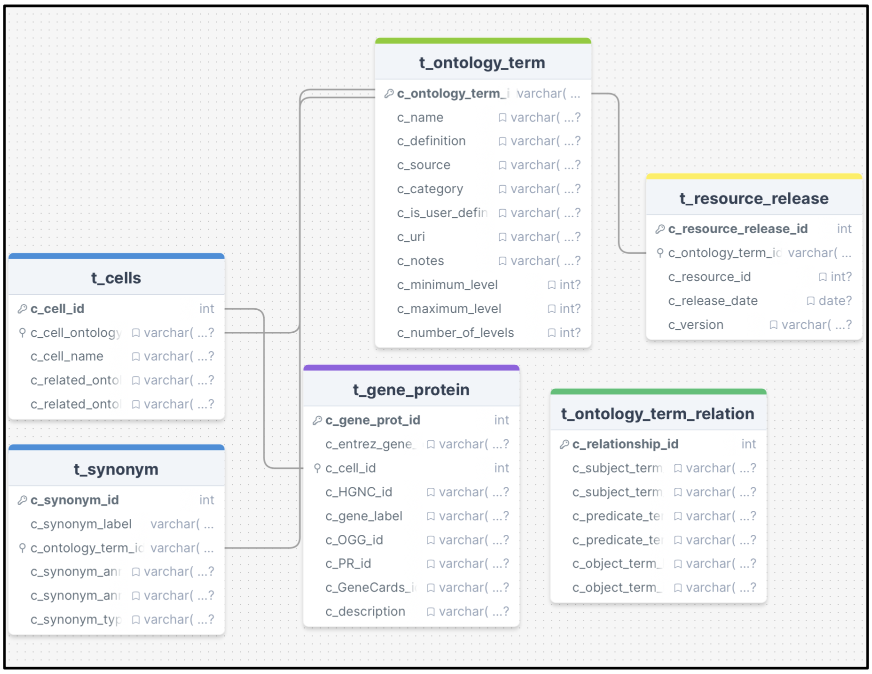
t_ontology_term
c_ontology_term_i varchar(...
c_name varchar(...?
c_definition varchar(...?
c_source varchar(...?
c_category varchar(...?
c_is_user_defin varchar(...?
c_uri varchar(...?
c_notes varchar(...?
c_minimum_level int?
c_maximum_level int?
c_number_of_levels int?
t_resource_release
c_resource_release_id int
c_ontology_term_ic varchar(...
c_resource_id int?
c_release_date date?
c_version varchar(...?
t_cells
c_cell_id int
c_cell_ontology varchar(...?
c_cell_name varchar(...?
c_related_onto varchar(...?
c_related_onto varchar(...?
t_gene_protein
c_gene_prot_id int
c_entrez_gene varchar(...?
c_cell_id int
c_HGNC_id varchar(...?
c_gene_label varchar(...?
c_OGG_id varchar(...?
c_PR_id varchar(...?
c_GeneCards_i varchar(...?
c_description varchar(...?
t_ontology_term_relation
c_relationship_id int
c_subject_term varchar(...?
c_subject_term varchar(...?
c_predicate_te varchar(...?
c_predicate_te varchar(...?
c_object_term_ varchar(...?
c_object_term_ varchar(...?
t_synonym
c_synonym_id int
c_synonym_label varchar(...
c_ontology_term_ic varchar(...
c_synonym_an varchar(...?
c_synonym_an varchar(...?
c_synonym_typ varchar(...?

Figure 2: The CellCards database schema.

3.2 Ontology-based ETL extraction of cell information

Using the ETL data extraction method, we successfully loaded all of the data present in CL and CLO into the CellCards database tables t_cells, t_synonyms, t_ontology_term, t_gene_proteins, and t_ontology_term_relation, while optimizing run times by employing multi-threading, significantly reducing processing time when requesting data through the HGNC REST API.

Currently, 2 SPARQL scripts are implemented into the Python ETL and it collectively fetches information such as: cell ID, name, definition, "part of" relationships, synonyms, database references, and HGNC IDs. This integration formed a flexible and robust scaffold capable of retrieving all of the required attributes and characteristics associated with an arbitrary cell ID from either CL or CLO.

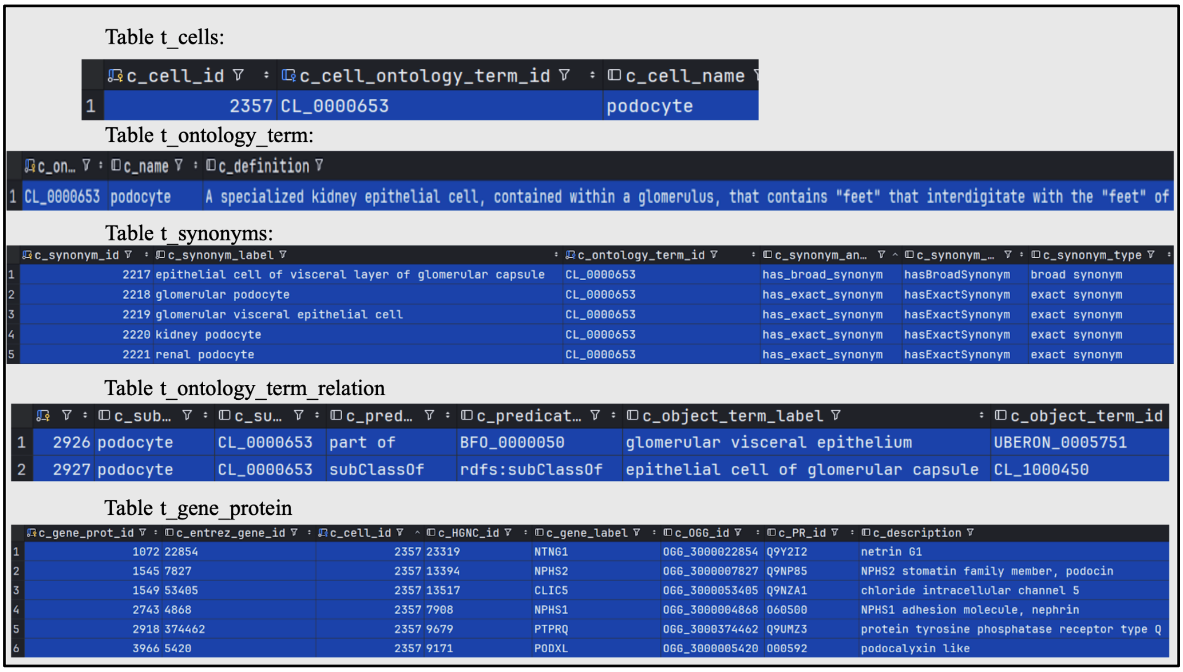

Table t_cells:

	c_cell_id	c_cell_ontology_term_id	c_cell_name
1	2357	CL_0000653	podocyte

Table t_ontology_term:

	c_on…	c_name	c_definition
1	CL_0000653	podocyte	A specialized kidney epithelial cell, contained within a glomerulus, that contains "feet" that interdigitate with the "feet" of

Table t_synonyms:

	c_synonym_id	c_synonym_label	c_ontology_term_id	c_synonym_an…	c_synonym_…	c_synonym_type
1	2217	epithelial cell of visceral layer of glomerular capsule	CL_0000653	has_broad_synonym	hasBroadSynonym	broad synonym
2	2218	glomerular podocyte	CL_0000653	has_exact_synonym	hasExactSynonym	exact synonym
3	2219	glomerular visceral epithelial cell	CL_0000653	has_exact_synonym	hasExactSynonym	exact synonym
4	2220	kidney podocyte	CL_0000653	has_exact_synonym	hasExactSynonym	exact synonym
5	2221	renal podocyte	CL_0000653	has_exact_synonym	hasExactSynonym	exact synonym

Table t_ontology_term_relation

		c_sub…	c_su…	c_pred…	c_predicat…	c_object_term_label	c_object_term_id
1	2926	podocyte	CL_0000653	part of	BFO_0000050	glomerular visceral epithelium	UBERON_0005751
2	2927	podocyte	CL_0000653	subClassOf	rdfs:subClassOf	epithelial cell of glomerular capsule	CL_1000450

Table t_gene_protein

	c_gene_prot_id	c_entrez_gene_id	c_cell_id	c_HGNC_id	c_gene_label	c_OGG_id	c_PR_id	c_description
1	1072	22854	2357	23319	NTNG1	OGG_3000022854	Q9Y2I2	netrin G1
2	1545	7827	2357	13394	NPHS2	OGG_3000007827	Q9NP85	NPHS2 stomatin family member, podocin
3	1549	53405	2357	13517	CLIC5	OGG_3000053405	Q9NZA1	chloride intracellular channel 5
4	2743	4868	2357	7908	NPHS1	OGG_3000004868	O60500	NPHS1 adhesion molecule, nephrin
5	2918	374462	2357	9679	PTPRQ	OGG_3000374462	Q9UMZ3	protein tyrosine phosphatase receptor type Q
6	3966	5420	2357	9171	PODXL	OGG_3000005420	O00592	podocalyxin like

Figure 3: Demonstration of ETL result using the podocyte as an example.

Figure 3 provides a comprehensive illustration of the data available for "podocyte" (CL_0000653) across the tables t_cells, t_synonyms, t_ontology_term, t_ontology_term_relation, and t_gene_proteins. The figure highlights the detailed and interconnected nature of the data.

The database now holds over 45,000 rows of data across these five tables. Each table serves a distinct purpose in encapsulating the vast array of information derived from the CL and CLO. This includes not just basic cell type data but also complex relational data that ties together synonyms, ontology terms, and biomarker information, thereby offering a robust and comprehensive resource for researchers. This extensive dataset ensures that researchers have access to a wealth of information for in-depth analysis and study, further advancing our understanding of cellular biology and ontology.

4. Discussion

The CellCards ETL (Extract, Transform, Load) method proved to be highly efficient in extracting, transforming, and loading thousands of cell records from both the Cell Ontology (CL) and the Cell Line Ontology (CLO) into the CellCards database. This process not only established a comprehensive foundation for unifying diverse cell type data but also enabled the detailed mapping of cell type connections, relationships, and interactions. By doing so, it provides a robust backend infrastructure for the CellCards website.

Developed with semantically enabled interoperable ontologies, CellCards ensures that its data is both machine-readable and human-readable, facilitating better understanding and utilization of cell data. This design allows the database to seamlessly integrate with various programs, enhancing its utility across a wide range of applications, including bioinformatics, computational biology, clinical research, and drug development. The ability to integrate CellCards into different computational and analytical contexts significantly broadens its applicability and effectiveness in supporting scientific research and innovation.

Future direction: The plan for the web interface is to be simple yet powerful, allowing users to navigate and access information efficiently. Future enhancements will include additional computational analysis tools that leverage the extensive knowledge within the database.

Acknowledgements

We acknowledge the Undergraduate Research Opportunity Program (UROP) of the University of Michigan for supporting MC's research on the project.

References

[1] He Y, Duncan W, Oudeif F, Lloyd A, Tian Y, Chen Y, Cooke M, Diehl AD, Mungall C, Hur J, Clarkson M, Ferrez M, Bvrd JB, Schaub J, Sarder P, Jain S. CellCards: An ontology-based knowledgebase for categorization and analysis of individual cells and cell connectomes. *The 6th International Cells in Experimental Life Science Workshop (CELLS 2022), in association with the 2022 International Conference on Biomedical Ontologies (ICBO-2022)*, Ann Arbor, MI, USA.

[2] A. D. Diehl, T. F. Meehan, Y. M. Bradford, M. H. Brush, W. M. Dahdul, D. S. Dougall, *et al.*, "The Cell Ontology 2016: enhanced content, modularization, and ontology interoperability," *J Biomed Semantics,* vol. 7, p. 44, July 4 2016.

[3] S. Shantivijai, Y. Lin, Z. Xiang, T. F. Meehan, A. D. Diehl, U. D. Vempati*, et al.*, "CLO: The Cell Line Ontology," *J Biomed Semantics,* vol. 5, p. 37, 2014.

[4] C. J. Mungall, C. Torniai, G. V. Gkoutos, S. E. Lewis, and M. A. Haendel, "Uberon, an integrative multi-species anatomy ontology," *Genome Biol,* vol. 13, p. R5, 2012.

[5] Herr, B.W., Hardi, J., Quardokus, E.M. *et al.* Specimen, biological structure, and spatial ontologies in support of a Human Reference Atlas. *Sci Data* 10, 171 (2023). https://doi.org/10.1038/s41597-023-01993-8

[6] Seal RL, Braschi B, Gray K, Jones TEM, Tweedie S, Haim-Vilmovsky L, Bruford EA. Genenames.org: the HGNC resources in 2023. Nucleic Acids Res. PMID: 36243972 DOI: 10.1093/nar/gkac888

[7] Maglott D, Ostell J, Pruitt KD, Tatusova T. Entrez Gene: gene-centered information at NCBI *Nucleic Acids Res.* 2007 Jan; 35(Database issue): D26–D31.

[8] Natale DA, Arighi CN, Blake JA, Bona J, Chen C, Chen SC, Christie KR, Cowart J, D'Eustachio P, Diehl AD, Drabkin HJ, Duncan WD, Huang H, Ren J, Ross K, Ruttenberg A, Shamovsky V, Smith B, Wang Q, Zhang J, El-Sayed A, Wu CH. Protein Ontology (PRO): enhancing and scaling up the representation of protein entities. *Nucleic Acids Res.* 2017 Jan 4;45(D1):D339-D346.

[9] He Y, Liu Y, Zhao B. OGG: a biological ontology for representing genes and genomes in specific organisms. *Proceedings of the 5th International Conference on Biomedical Ontologies (ICBO)*, Houston, Texas, USA. October 8-9, 2014. Pages 13-20.

[10] HuBmap Consortium, "The human body at cellular resolution: the NIH Human Biomolecular Atlas Program," *Nature,* vol. 574, pp. 187-192, Oct 2019.