ForgeDB: Database Design and ETL Pipeline for Studying Emergent Cooperation in LLM Agents

Emily D. Carpenter
Marketing and Data Sciences
Anderson College of Business and Computing
Regis University, Denver, CO, USA
ecarpenter004@regis.edu

Abstract

This practicum project adds a data engineering contribution to the GENESIS (General Emergent Norms, Ethics, and Societies in Silico) research project at Regis University. The GENESIS project is investigating whether intelligent computer agents are able to develop cooperative behaviors and moral reasoning via simulated interactions, specifically through use of the Iterated Prisoner's Dilemma (IPD) testing environment [1].

The core simulation framework and Large Language Model (LLM) agent code has been provided by the project researchers, but there is no systematic method to store, organize, and analyze experimental results from multiple game runs. As such, this project addresses that gap by designing and implementing a database solution to capture experimental data generated by intelligent agents playing the IPD game in addition to studying the possible emergence of cooperation between agents.

The infrastructure created in this practicum project will enable researchers to conduct analysis across experiments to identify the conditions that lead to cooperative emergence, examine patterns in agent reasoning, and contribute towards planned research publications, the first step in a multi-year plan for the GENESIS research project. This practicum demonstrates practical data engineering skills including database design and interface with the research project, version control, and documentation practices while contributing to ongoing Artificial Intelligence (AI) ethics research that explores the boundaries between optimization and moral behavior.

I. Introduction/Background

The purpose of this project is to contribute to the research on GENESIS proposed by Dr. Douglas Hart and Dr. Kellen Sorauf, Regis University, which investigates the behavioral patterns exhibited by intelligent agents and whether they follow Jonathan Haidt's six moral foundations, starting with cooperation [1]. The research will be conducted on the Framework for Observing and Researching Group Emergence (FORGE) platform via the department's compute cluster supporting LLM and multi-agent reinforcement learning (MARL).

This practicum project will contribute to the project by testing agent behavior patterns and providing infrastructure to record and facilitate analysis of agent behavioral patterns. Several solutions to meet the data engineering challenges are available, including a PostgreSQL database or SQLite database file, both of which interface with the Python programming language and are supported on the Linux operating system currently installed on the cluster.

The results of student research, including this practicum, are expected to provide data for the first phase of a multi-year research plan to investigate agentic behavior. Phase I deliverables include planned research publications expected to be published towards the end of the current calendar year.

II. Problem Statement

This practicum project joins ongoing research by Regis University professors that "investigates moral foundation emergence through institutional evolution," starting with "simple cooperation in fixed games" and culminating in "complex institutional ecosystems addressing cooperation, authority, loyalty, and potentially care and sanctity," behaviors analogous to Jonathan Haidt's moral foundations [1].

The current testing environment produces output in the JSON file format. This creates a challenge for researchers to analyze behavior across multiple rounds of testing, including changes in testing variables to try to guide agents towards cooperation. This practicum will provide the infrastructure to record testing results conducted by multiple researchers, which should be flexible enough to adjust to changes as research evolves from fixed game behaviors to MARL via Ray RLlib that leverages all machines in the compute cluster.

III. Methodology/Approach

The database application PostgreSQL [2] was selected to serve as a persistent storage system for the GENESIS project due to its open-source license and easy integration with the Python programming language used throughout the project. To facilitate the Extract, Transform, and Load (ETL) process of project JSON files, a Python module was created that utilized the Psycopg 3 [3] library to interface with the PostgreSQL database application. This module provided methods for processing and querying results produced by experiments conducted for the research project.

The JSON files produced by the research experiments consisted of several top-level keys metadata, configuration, prompts with several nested objects and arrays agents, episodes:rounds. The structure was consistent but could benefit from normalization when stored within a relational database application such as that utilized by PostgreSQL.

A total of four tables were created in the `FORGE` database, one each to store experiment results and configuration data [`RESULTS`], agent data [`LLM_AGENTS`], episode data [`EPISODES`], and rounds detail data [`ROUNDS`]. A 5th table [`RESEARCH_LOG`] was created to serve as a general research notebook for the project to document milestones and other information as required by the primary researchers.

A serialized primary key was created for the `RESULTS` table to be used as a foreign key in the `LLM_AGENTS` and `EPISODES` tables. Additionally, a serialized primary key [`EPISODE_ID`] was created in the `EPISODES` table to relate the `EPISODES` and `ROUNDS` tables. Each child table (i.e. `LLM_AGENTS`, `EPISODES`, `ROUNDS`) was configured such that deletion of a row from the primary experiment table [`RESULTS`] would also delete related data in the child tables (known as a deletion cascade). Lastly, several SQL views were created to provide researchers with ”prepared” SQL statements to facilitate easy querying of the data without being required to construct complex SQL script files.

The full database schema is presented in the Entity Relationship Diagram (ERD) in Figure 1. All database objects created for the IPD game involving 2 LLM agents were created in a common database schema named `ipd2` to allow creation of additional schemas for other types of LLM games.

The SQL views listed below were created for the database and are detailed in the GitHub repository SQL script setup file:

- `raw_data_vw`:
  - Returns experiment metadata with original raw JSON.
- `results_vw`:
  - Returns all data columns within the database with each row representing a single round per agent for the IPD game.
- `experiment_summary_vw`:
  - Returns one row per experiment with agent data pivoted to columns. Recommended for comparing experiments at a high level of analysis.
- `episode_summary_vw`:
  - Returns one row per episode with agent data pivoted to columns. Useful for tracking cooperation trajectories across episodes. Also used in creation of the ”scatter plot connected points” chart that allows viewing cooperation rates by episodes.

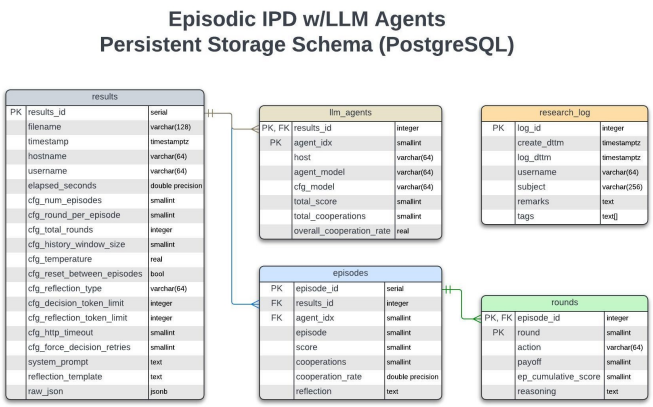

Fig. 1. Forge Database Entity Relationship Diagram

- `rounds_summary_vw`:
  - Returns one row per round with both agents' data side-by-side (i.e. pivoted from rows to columns). Good for providing round-by-round comparison of agent behavior.
- `rounds_detail_vw`:
  - Returns one row per round per agent (unpivoted data). This is similar to the `results_vw` but is focused on round-level data without the configuration column data.

To facilitate transfer of data from the experiment JSON files into the database, a Python module script file [`forgedb.py`] was created with one class object that could be instantiated in a separate Python script or Jupyter Notebook file. The module contained one class object [`ForgeDB`] with three groups of methods, each group to handle database queries, research log queries, and JSON file processing. JSON files themselves could be imported into the database through the Linux shell or as method calls from a `ForgeDB` variable object created within a Python Script file or Jupyter Notebook code block.

All changes completed to configure and install the persistent storage solution for the research project were documented in a Markdown text file [`ForgeDB - Data Access & ETL Reference.pdf`] saved to the project GitHub repository. This reference document includes instructions on how to utilize the Python module to process JSON experiment files and query results stored in the database.

The AI assistant Claude [4] was utilized during development of the database schema, ETL pipeline, and project documentation. All code and content proposed by the AI agent were reviewed, edited, and approved by the author, and all work included in this project is the original work of the author.

IV. Data Analysis Collection/Acquisition, Preparation, EDA, Visualization, Reporting

This practicum project focused on the Data Engineering aspects of the research project to provide a persistent storage medium to organize experimental data for long term retention and analysis. Access to the database was restricted to the department compute cluster, and access granted only to research participants.

Data generated for the research project includes results of IPD game participation by LLM agents for varying input variables in experiment conditions created by each research participant.

As such, no data were created or produced for this aspect of the project. Moving forward, future participants in the research project will benefit from a single authoritative source of data to conduct ongoing research and analysis of the GENESIS project.

V. Expected Outcomes

The outcomes of this project are to contribute to the body of research proposed by Dr. Hart and Dr. Sorauf. The goal for future planned publications is to achieve an emergence of cooperative behavior between intelligent agents.

Specific contributions from this practicum include creation of infrastructure to serve as persistent storage to allow researchers to analyze test result data across experiments, and contribute to future planned publications as directed by project principals.

VI. Timeline

The revised proposed timeline for the Practicum I project is as follows:

- Week 1:
  - Discuss research project with principals and other student(s).
  - Submit project proposal to WorldClass
- Weeks 2-5:
  - Design, implement, and test persistent data store solution.
  - Build ETL interface for loading and querying database.
- Week 6:
  - Present persistent storage solution to research group.
  - Complete final revisions to persistent storage solution.
- Week 7: Prepare findings and practicum deliverables.
- Week 8: Submit practicum deliverables and present project to class.

VII. Conclusion

This practicum project focused on providing a persistent storage solution for the GENESIS research project in order to meet the Data Engineering specialization requirement path followed by the author for the Data Science Degree Program at Regis University. One of the problems identified at the beginning of the research project was how to handle the many files produced as a result of experimentation with the LLM agents participating in the IPD game system as well as how to retain data as participants rotate out of the project.

The database storage solution provided by this practicum project will allow researchers to centralize all results files as well as document milestones achieved during research phases through use of the database research log. Additional benefits of the persistent storage solution include the ability to query results across *all* experiments regardless of which researcher generated results or whether a particular student participant has moved on to other projects.

The structured PostgreSQL RDBMS solution utilized for this research project allows separation of game types into individual database schemas. This particular practicum project was created in the `ipd2` schema to denote use of the Iterated Prisoner's Dilemma where the "2" denotes an initial participation of 2 agents.

Future work could include use of a NoSQL application to allow for a more rapid deployment and implementation phase of persistent storage. The inherent schema-less design of NoSQL solutions (e.g. MongoDB, CouchDB, DynamoDB, etc...) can readily meet changing research goals but come with an increased need to understand key:value pairs generated in experiment data output files.

References

[1] D. Hart and K. Sorauf, "GENESIS project: From cooperation to conscience," 2025, Unpublished project documentation, Regis University.
[2] The PostgreSQL Global Development Group, "PostgreSQL," https://www.postgresql.org/, 1996–2026.
[3] D. Varrazzo and The Psycopg Team, "Psycopg 3," https://www.psycopg.org/psycopg3/, n.d., [Computer software].
[4] Anthropic, "Claude," https://claude.ai/, 2025–2026, aI assistant used for database design, code development, and documentation.