Documentation for Data Generation

Research Intern Take-Home Assignment

Abhay Singh

Personal Email: abhaysingh65534@gmail.com
Institute Email: 2022ume0193@iitjammu.ac.in
LinkedIn: linkedin.com/in/abhay-singh-4a8286254
GitHub: github.com/your-github

January 6, 2026

Contents

1 Overview

The objective of this work is to generate a realistic, enterprise-scale seed dataset simulating an Asana workspace for a B2B SaaS organization with thousands of employees. This dataset is designed to support reinforcement learning (RL) environments for computer-use AI agents by ensuring high data realism, logical consistency, and reproducibility.

Rather than relying on live scraping or external API calls, the data generation strategy uses a combination of **LLM-inspired heuristic templates**, **synthetic distributions**, and **derived relationships**. This approach mirrors real-world Asana usage patterns while keeping the pipeline deterministic and easy to evaluate.

Each column in the schema is generated using one of the following strategies:

- **LLM-inspired Heuristics**: Template-based natural language generation inspired by public GitHub issues, Asana templates, and project management best practices.
- **Synthetic + Heuristics**: Programmatic generation guided by realistic probability distributions and constraints.
- **Derived**: Values deterministically derived from other entities or relationships.

2 Organizations

This table stores high-level workspace information representing an organization in the system.

Column	Type	Generation Method	Description
organization_id	TEXT	Synthetic	Unique identifier for the organization.
name	TEXT	LLM-inspired Heuristic	Organization name following SaaS naming conventions.
domain	TEXT	Derived	Email domain derived from organization name.

3 Users

This table stores user accounts associated with an organization.

Column	Type	Generation Method	Description
user_id	TEXT	Synthetic	Unique identifier for the user.
organization_id	TEXT (FK)	Derived	References the parent organization.

name	TEXT	Synthetic Heuristic	+	Generated using realistic name distributions.
email	TEXT	Derived		Derived from user name with enforced uniqueness.
role	TEXT	Heuristic		User role within the organization.
is_admin	BOOLEAN	Heuristic		Indicates administrative privileges.
is_active	BOOLEAN	Heuristic		Indicates whether the user is active.
created_at	TIMESTAMP	Synthetic Heuristic	+	Account creation timestamp.
last_login	TIMESTAMP	Derived		Last login time for active users.

4 Teams

This table stores team groupings used to organize users and projects.

Column	**Type**	**Generation Method**	**Description**
team_id	TEXT	Synthetic	Unique identifier for the team.
organization_id	TEXT (FK)	Derived	References the organization.
name	TEXT	Heuristic	Team name following common enterprise patterns.

5 Projects

This table stores projects that group related tasks.

Column	**Type**	**Generation Method**		**Description**
project_id	TEXT	Synthetic		Unique identifier for the project.
organization_id	TEXT (FK)	Derived		References the organization.
team_id	TEXT (FK)	Derived		Owning team for the project.
name	TEXT	LLM-inspired Heuristic		Project name inspired by task management templates.
owner_id	TEXT (FK)	Derived		Project owner selected from users.
start_date	DATE	Synthetic Heuristic	+	Project start date.
due_date	DATE	Synthetic Heuristic	+	Project due date.

6 Sections

This table stores workflow stages within a project.

Column	Type	Generation Method	Description
section_id	TEXT	Synthetic	Unique identifier for the section.
project_id	TEXT (FK)	Derived	References the parent project.
name	TEXT	Heuristic	Workflow stage name.
order_index	INTEGER	Derived	Position of the section in workflow.

7 Tasks

This table stores individual units of work.

Column	Type	Generation Method	Description
task_id	TEXT	Synthetic	Unique identifier for the task.
organization_id	TEXT (FK)	Derived	References the organization.
title	TEXT	LLM-inspired Heuristic	Natural language task title.
assignee_id	TEXT (FK)	Derived + Heuristic	Assigned user or NULL.
creator_id	TEXT (FK)	Derived	User who created the task.
created_at	TIMESTAMP	Synthetic + Heuristic	Task creation time.
due_date	TIMESTAMP	Synthetic + Heuristic	Task due date.
is_completed	BOOLEAN	Heuristic	Completion status.
completed_at	TIMESTAMP	Derived	Completion timestamp.

8 Comments

This table stores comments added to tasks.

Column	Type	Generation Method	Description
comment_id	TEXT	Synthetic	Unique identifier for the comment.
task_id	TEXT (FK)	Derived	References the related task.
user_id	TEXT (FK)	Derived	Author of the comment.

content	TEXT	LLM-inspired Heuristic	Comment text.
is_system_generated	BOOLEAN	Heuristic	Indicates system-generated comments.

9 Custom Fields

This table defines reusable custom metadata fields.

Column	Type	Generation Method	Description
field_id	TEXT	Synthetic	Unique field identifier.
organization_id	TEXT (FK)	Derived	Owning organization.
name	TEXT	Heuristic	Custom field name.
type	TEXT	Heuristic	Field data type.
is_global	BOOLEAN	Heuristic	Indicates global availability.
created_by	TEXT (FK)	Derived	User who created the field.

10 Utility Modules

The `utils` directory contains shared helper functions used across the data generation pipeline to ensure modularity, readability, and reuse.

- **db.py**: Handles SQLite database initialization, schema execution, and connection configuration, including foreign key enforcement and performance-related settings.
- **faker_utils.py**: Provides helper functions for generating realistic synthetic names, emails, and short text snippets used in user and content generation.
- **time_utils.py**: Contains utility functions for generating bounded past and future timestamps to support realistic temporal distributions.
- **logger.py**: Configures a lightweight logging setup to track pipeline execution steps and aid debugging.
- **validation.py**: Implements post-generation sanity checks to verify temporal and relational consistency in the generated dataset.

—

11 Temporal Consistency

Temporal fields are generated with explicit constraints to prevent invalid states. In particular:

- A task cannot be completed before it is created.
- Completion timestamps are never generated in the future.
- Due dates are always generated after task creation.

These constraints are enforced programmatically during data generation and validated post-generation.

—

12 Relational Consistency

Referential integrity is enforced at both the schema and generation levels:

- Foreign key constraints ensure valid references between entities.
- Entities are generated in dependency order (organizations → users → teams → projects → tasks).
- Tasks are always linked to at least one project.

This guarantees that the resulting dataset is free from orphaned records and reflects valid business logic.

—

13 Conclusion

This methodology balances realism, scalability, and reproducibility. By combining LLM-inspired heuristics with deterministic logic, the dataset avoids artificial shortcuts while remaining suitable for controlled experimentation in RL environments.