“Output Hub”

A knowledge management tool optimised for the storage and retrieval of the outputs of large language models (LLMs) including GPTs - intended for both professional users of LLMs and individual users generating at scale and requiring robust output storage.

Output Hub aims to provide a cohesive framework for managing LLM outputs to facilitate their refinement and development by humans. It is designed to ensure that useful textual LLM generations are retained, well-indexed, and accessible - in much the same way that traditional knowledge management tools have sought to manage knowledge.

In addition to that, and in recognition of how LLM usage works, Output Hub *(working name)* also provides modules for recording prompts (ie, a prompt library) and for storing the configuration of custom LLM agents.

A core focus of the design of Output Hub is using the power of relational database design to create deep linkages within the system. This, in turn, can yield further value.

For example, by creating associations between custom LLM agents and their outputs, agent configurations can be improved iteratively. Other metadata elements have been added to reflect information that may be important in professional usage environments. An example of this is associating formal data retention policies with individual outputs.

The framework outlined in this system describes a system designed to enable LLM *outputs* to receive proper attention and management in internal information systems. The data model and architecture are both works in progress and the precise fields and tables outlined here should be understood as merely illustrative of the broader concept.

Daniel Rosehill (danielrosehill.com)
26-09-24

“Output Hub” - V1

Natural Language Data Model

Core Modules Overview:

The core modules in the system consist of four main tables: **Conversations**, **Prompts**, **Outputs**, and **Agents**. These tables are fundamental to the system's operation, capturing interactions, inputs, outputs, and the agents involved.

1. **Conversations Table:**

    - **Purpose:** This table holds conversations, which are instances where prompts and outputs are associated with a particular agent.
    - **Fields:**
        - `id`: A unique identifier for each conversation.
        - `title`: The title of the conversation.
        - `prompt_id`: References the associated prompt.
        - `output_id`: References the output generated during the conversation.
        - `agent_id`: References the agent (GPT/LLM) used.
        - `conversation_date`: Timestamp of when the conversation occurred, defaulting to the current time.
        - `user_id`: References the user involved in the conversation, with cascading deletes.

2. **Prompts Table:**

    - **Purpose:** Stores prompts used in the system, including whether they were manually engineered.
    - **Fields:**
        - `id`: A unique identifier for each prompt.
        - `title`: The title of the prompt (optional).
        - `prompt`: The actual text of the prompt, which is required.
        - `engineered`: Boolean field indicating if the prompt was manually engineered.
        - `created_at`: Timestamp of when the prompt was created, defaulting to the current time.

By: Daniel Rosehill (danielrosehill.com)
24-09-24

- `conversation_ids`: An array of conversation IDs related to this prompt.
- `output_ids`: An array of output IDs generated from this prompt.

3. **Outputs Table:**

   - **Purpose:** Stores outputs generated by agents in response to prompts.
   - **Fields:**
     - `id`: A unique identifier for each output.
     - `title`: The title of the output (optional).
     - `output`: The actual text of the output, which is required.
     - `output_edited`: Edited version of the output (optional).
     - `notes`: Additional notes about the output.
     - `created_at`: Timestamp of when the output was created, defaulting to the current time.
     - `conversation_ids`: An array of conversation IDs related to this output.
     - `prompt_id`: References the prompt that generated this output, with nullifying deletes.

4. **Agents Table:**

   - **Purpose:** Represents the agents (GPT/LLMs) used in the system.
   - **Fields:**
     - `id`: A unique identifier for each agent.
     - `name`: The name of the agent (required).
     - `description`: A description of the agent or its purpose.
     - `created_at`: Timestamp of when the agent was created, defaulting to the current time.
     - `conversation_ids`: An array of conversation IDs related to this agent.

By: Daniel Rosehill (danielrosehill.com)
24-09-24

“Output Hub” - V1

Natural Language Data Model

Lookup Tables Overview:

Several lookup tables are designed to provide additional metadata and classification to the core modules, allowing for more granular control and organization.

1. **Tags Lookup:**

    - **Purpose:** A list of tags that can be associated with any of the core modules.
    - **Fields:**
        - `id`: A unique identifier for each tag.
        - `tagname`: The name of the tag.
        - `tagdesc`: A description of the tag.
        - `createdat`: Timestamp of when the tag was created.
    - **Data Relationships:** Many-to-many relationships with all core modules, allowing multiple tags per entity.

2. **Categories Lookup:**

    - **Purpose:** Categorizes the core modules, ensuring each entity belongs to at least one category.
    - **Fields:**
        - `id`: A unique identifier for each category.
        - `cat_name`: The name of the category.
        - `cat_desc`: A description of the category.
        - `created_at`: Timestamp of when the category was created.
    - **Data Relationships:** Many-to-many relationships with all core modules, with non-nullable constraints ensuring every entity has a category.

3. **Access UI Lookup:**

    - **Purpose:** Defines the user interface elements or access points for core modules.
    - **Fields:**
        - `id`: A unique identifier for each UI element.
        - `uiname`: The name of the UI element.

By: Daniel Rosehill (danielrosehill.com)
24-09-24

- **Data Relationships:** One-to-one relationships with each core module, allowing each entity to be associated with one access UI.

4. **Accuracy Level Lookup:**

   - **Purpose:** Indicates the accuracy level of outputs or agents, aiding in quality control.
   - **Fields:**
     - `id`: A unique identifier for each accuracy level.
     - `level_name`: The name of the accuracy level.
     - `desc`: A description of what the accuracy level entails.
   - **Data Relationships:** One-to-many relationships with outputs and agents, allowing multiple entities to share the same accuracy level.

5. **Actionability Level Lookup:**

   - **Purpose:** Categorizes outputs based on their actionability, determining which outputs require further action.
   - **Fields:**
     - `id`: A unique identifier for each actionability level.
     - `levelname`: The name of the actionability level.
     - `level_desc`: A description of the actionability level.
   - **Data Relationships:** One-to-many relationship with outputs, with nullable constraints allowing for optional actionability classification.

6. **Activity Status Lookup:**

   - **Purpose:** Tracks the status of agents and prompts, distinguishing between active and inactive entities.
   - **Fields:**
     - `id`: A unique identifier for each activity status.
     - `statusname`: The name of the activity status.
     - `status_desc`: A description of what the status represents.
   - **Data Relationships:** One-to-many relationships with agents and prompts, with non-nullable constraints ensuring every entity has a status.

By: Daniel Rosehill (danielrosehill.com)
24-09-24

---

7. **Agent Groups Lookup:**

    - **Purpose:** Groups agents into categories for easier management and filtering.
    - **Fields:**
        - `id`: A unique identifier for each group.
        - `groupname`: The name of the group.
        - `groupdesc`: A description of the group’s purpose.
    - **Data Relationships:** Many-to-many relationships with agents, allowing agents to belong to multiple groups.

8. **Automation Level Lookup:**

    - **Purpose:** Specifies the level of automation an agent possesses.
    - **Fields:**
        - `id`: A unique identifier for each automation level.
        - `levelname`: The name of the automation level.
        - `desc`: A description of the level of automation.
    - **Data Relationships:** Many-to-many relationships with agents, allowing multiple automation levels to be assigned.

9. **Creation Account Lookup:**

    - **Purpose:** Tracks which account was used to create or manage core module entities, useful for systems with multiple accounts.
    - **Fields:**
        - `id`: A unique identifier for each account.
        - `accountname`: The name of the account.
        - `accountemail`: The email associated with the account.
        - `notes`: Additional notes regarding the account.
    - **Data Relationships:** One-to-many relationships with all core modules, with non-nullable constraints requiring account assignment.

10. **Output Libraries Lookup:**

By: Daniel Rosehill (danielrosehill.com)
24-09-24

- **Purpose:** Organizes outputs into libraries, facilitating controlled access and management.
- **Fields:**
  - `id`: A unique identifier for each library.
  - `libraryname`: The name of the library.
  - `desc`: A description of the library.
- **Data Relationships:** Many-to-many relationships with outputs, allowing outputs to belong to multiple libraries.

11. **Data Retention Plan Lookup:**

- **Purpose:** Associates core module entities with data retention plans, useful for compliance with regulations like GDPR.
- **Fields:**
  - `id`: A unique identifier for each retention plan.
  - `planname`: The name of the retention plan.
  - `desc`: A description of the retention plan.
- **Data Relationships:** Many-to-many relationships with all core modules, with nullable constraints allowing optional retention plan assignment.

12. **Data Sensitivity Level Lookup:**

- **Purpose:** Labels core module entities based on their sensitivity, aiding in compliance and data management.
- **Fields:**
  - `id`: A unique identifier for each sensitivity level.
  - `sensitivitylevel`: The name of the sensitivity level.
  - `description`: A description of what the sensitivity level entails.
- **Data Relationships:** One-to-many relationships with all core modules, with nullable constraints allowing optional sensitivity classification.

13. **Programming Language Lookup:**

- **Purpose:** Lists programming languages associated with core module entities, useful for outputs containing code.

By: Daniel Rosehill (danielrosehill.com)
24-09-24

- **Fields:**
  - `id`: A unique identifier for each programming language.
  - `language`: The name of the programming language.
  - `desc`: A description of the programming language.
- **Data Relationships:** Many-to-many relationships with all core modules, allowing entities to be associated with multiple programming languages.

14. **Data Size Classifiers Lookup:**

- **Purpose:** Classifies data sizes across the system, grouping files or outputs by size.
- **Fields:**
  - `id`: A unique identifier for each size classification.
  - `size`: The name of the size classification.
  - `minsize`: The minimum size in bytes.
  - `maxsize`: The maximum size in bytes.
  - `desc`: A description of the size classification.

15. **Output Types Lookup:**

- **Purpose:** Specifies the format in which outputs are stored (e.g., markdown, text

, JSON). - **Fields:** - `id`: A unique identifier for each output type. - `type`: The name of the output type. - `desc`: A description of the output type. - **Data Relationships:** Many-to-many relationships with outputs, requiring non-nullable assignment of output types.

16. **Markdown Conversion Lookup:**

- **Purpose:** Tracks the conversion status of markdown files to other formats.
- **Fields:**
  - `id`: A unique identifier for each conversion status.
  - `status`: The name of the conversion status.
- **Data Relationships:** One-to-many relationship with outputs, requiring non-nullable status assignment.

By: Daniel Rosehill (danielrosehill.com)
24-09-24

17. **File Formats Lookup:**

    - **Purpose:** Lists file formats associated with generated files, useful for managing file outputs.
    - **Fields:**
        - `id`: A unique identifier for each file format.
        - `format`: The name of the file format.
        - `extension`: The file extension.
        - `desc`: A description of the file format.
    - **Data Relationships:** Many-to-many relationships with outputs, with nullable constraints allowing optional format assignment.

18. **Follow Up Activities Lookup:**

    - **Purpose:** Assigns follow-up activities to outputs and prompts (e.g., fact-check, review sources).
    - **Fields:**
        - `id`: A unique identifier for each follow-up activity.
        - `activitiy`: The name of the follow-up activity.
        - `desc`: A description of the follow-up activity.
    - **Data Relationships:** Many-to-many relationships with prompts and outputs, with nullable constraints allowing optional activity assignment.

19. **Github Plans Lookup:**

    - **Purpose:** Marks agents, prompts, or outputs for release on GitHub, facilitating open sourcing.
    - **Fields:**
        - `id`: A unique identifier for each GitHub plan.
        - `plan`: The name of the GitHub plan.
        - `desc`: A description of the GitHub plan.
    - **Data Relationships:** One-to-many relationships with prompts, agents, and outputs, with nullable constraints allowing optional plan assignment.

By: Daniel Rosehill (danielrosehill.com)
24-09-24

---

20. **LLMs Lookup:**

    - **Purpose:** Contains a list of large language models (LLMs), associating them with core modules.
    - **Fields:**
        - `id`: A unique identifier for each LLM.
        - `name`: The name of the LLM.
        - `desc`: A description of the LLM.
        - `by`: The creator or provider of the LLM.
    - **Data Relationships:** One-to-many relationships with prompts, outputs, and agents, with nullable constraints allowing optional LLM assignment.

21. **LLM Platforms Lookup:**

    - **Purpose:** Lists platforms that host LLMs, associating them with core modules.
    - **Fields:**
        - `id`: A unique identifier for each LLM platform.
        - `name`: The name of the LLM platform.
        - `desc`: A description of the platform.
    - **Data Relationships:** One-to-many relationships with prompts, outputs, and agents, with nullable constraints allowing optional platform assignment.

22. **Common Statuses Lookup:**

    - **Purpose:** Holds simple statuses (e.g., to do, pending, in progress, done) for fallback use in tables without dedicated status tracking.
    - **Fields:**
        - `id`: A unique identifier for each status.
        - `status`: The name of the status.
        - `desc`: A description of the status.
    - **Data Relationships:** One-to-many relationships with prompts, outputs, and agents.

23. **Agent Capabilities Lookup:**

By: Daniel Rosehill (danielrosehill.com)
24-09-24

- **Purpose:** Denotes the capabilities of agents (e.g., text summation), aiding in filtering and searching through agents.
- **Fields:**
  - `id`: A unique identifier for each capability.
  - `capability`: The name of the capability.
  - `desc`: A description of the capability.
- **Data Relationships:** Many-to-many relationships with agents, with nullable constraints allowing optional capability assignment.

24. **Knowledge Types Lookup:**

- **Purpose:** Groups outputs by the type of knowledge exhibited (e.g., guide, checklist).
- **Fields:**
  - `id`: A unique identifier for each knowledge type.
  - `type`: The name of the knowledge type.
  - `desc`: A description of the knowledge type.
- **Data Relationships:** Nullable relationships with outputs, allowing optional knowledge type assignment.

25. **Prompt Stage Lookup:**

- **Purpose:** Manages the prompt library by assigning stages to prompts (e.g., draft, active, deactivated).
- **Fields:**
  - `id`: A unique identifier for each prompt stage.
  - `stage`: The name of the prompt stage.
  - `desc`: A description of the prompt stage.
- **Data Relationships:** One-to-many relationship with prompts, with non-nullable constraints requiring stage assignment.

26. **Countries Lookup:**

By: Daniel Rosehill (danielrosehill.com)
24-09-24

“Output Hub” - V1

Natural Language Data Model

- **Purpose:** Lists countries to assign core module entities to specific geographies when applicable.
- **Fields:**
  - `id`: A unique identifier for each country.
  - `country`: The name of the country.
- **Data Relationships:** One-to-many relationships with prompts, outputs, and agents, with nullable constraints allowing optional country assignment.

By: Daniel Rosehill (danielrosehill.com)
24-09-24

“Output Hub” - V1

Natural Language Data Model

Data Relationships Overview:

Many-to-Many (M2M) Relationships:

1. **Tags Lookup** (`lookup_tags`)

    - **Core Modules:** All core modules (Conversations, Prompts, Outputs, Agents) can be associated with multiple tags, and each tag can be associated with multiple entities across these modules.

2. **Categories Lookup** (`lookup_cats`)

    - **Core Modules:** All core modules must be associated with at least one category, and each category can be linked to multiple entities across these modules.

3. **Agent Groups Lookup** (`lookup_agentgroups`)

    - **Agents:** Each agent can belong to multiple groups, and each group can contain multiple agents.

4. **Automation Level Lookup** (`lookup_automationlevel`)

    - **Agents:** Agents can have multiple automation levels, and each automation level can be associated with multiple agents.

5. **Output Libraries Lookup** (`lookup_outputlibraries`)

    - **Outputs:** Outputs can be stored in multiple libraries, and each library can contain multiple outputs.

6. **Data Retention Plan Lookup** (`lookup_dataretentionplan`)

By: Daniel Rosehill (danielrosehill.com)
24-09-24

- **Core Modules:** Entities across all core modules can be associated with multiple data retention plans, and each retention plan can apply to multiple entities.

7. **Programming Language Lookup** (`lookup_programminglanguage`)
    - **Core Modules:** Entities across all core modules can be linked to multiple programming languages, and each programming language can be associated with multiple entities.
8. **File Formats Lookup** (`lookup_fileformats`)
    - **Outputs:** Outputs can result in multiple files with different formats, and each format can be associated with multiple outputs.
9. **Follow Up Activities Lookup** (`lookup_followup_activities`)
    - **Prompts and Outputs:** Both prompts and outputs can have multiple follow-up activities assigned, and each activity can apply to multiple prompts or outputs.
10. **Agent Capabilities Lookup** (`lookup_agent_capabilities`)
    - **Agents:** Each agent can have multiple capabilities, and each capability can be associated with multiple agents.

By: Daniel Rosehill (danielrosehill.com)
24-09-24

“Output Hub” - V1

Natural Language Data Model

One-to-Many (O2M) Relationships:

1. **Conversations Table** (`conversations`)
    - **Core Modules:** Conversations can be associated with a single prompt, output, and agent, but each prompt, output, and agent can be part of multiple conversations.
2. **Prompts Table** (`prompts`)
    - **Outputs:** Each prompt can generate multiple outputs, but each output is linked to only one prompt.
3. **Outputs Table** (`outputs`)
    - **Markdown Conversion Lookup** (`lookup_md_conversion`): Each output can have one conversion status, but each conversion status can be associated with multiple outputs.
4. **Accuracy Level Lookup** (`lookup_accuracy_level`)
    - **Outputs and Agents:** Each output and agent can be associated with one accuracy level, but each accuracy level can apply to multiple outputs or agents.
5. **Actionability Level Lookup** (`lookup_actionability`)
    - **Outputs:** Each output can have one actionability level, but each level can be associated with multiple outputs.
6. **Activity Status Lookup** (`lookup_activitystatus`)
    - **Agents and Prompts:** Each agent or prompt can have one activity status, but each status can apply to multiple agents or prompts.

By: Daniel Rosehill (danielrosehill.com)
24-09-24

7. **Creation Account Lookup** (`lookup_creationaccount`)
    - **Core Modules:** Each entity in the core modules can be associated with one account, but each account can have multiple associated entities.
8. **Output Types Lookup** (`lookup_output_types`)
    - **Outputs:** Each output must be associated with one output type, but each type can be linked to multiple outputs.
9. **Github Plans Lookup** (`lookup_github_plans`)
    - **Prompts, Agents, Outputs:** Each entity in these core modules can have one associated GitHub plan, but each plan can be linked to multiple entities.
10. **LLMs Lookup** (`lookup_llms`)
    - **Prompts, Outputs, Agents:** Each entity can be associated with one LLM, but each LLM can apply to multiple entities.
11. **LLM Platforms Lookup** (`lookup_llm_platforms`)
    - **Prompts, Outputs, Agents:** Each entity can be associated with one LLM platform, but each platform can apply to multiple entities.
12. **Common Statuses Lookup** (`lookup_statuses`)
    - **Prompts, Outputs, Agents:** Each entity can have one status, but each status can apply to multiple entities.
13. **Knowledge Types Lookup** (`lookup_knowledge_types`)
    - **Outputs:** Each output can be associated with one knowledge type, but each type can apply to multiple outputs.

By: Daniel Rosehill (danielrosehill.com)
24-09-24

14. **Prompt Stage Lookup** (`lookup_promptstage`)

    - **Prompts:** Each prompt must be associated with one stage, but each stage can apply to multiple prompts.

15. **Countries Lookup** (`lookup_countries`)

    - **Prompts, Outputs, Agents:** Each entity can be associated with one country, but each country can apply to multiple entities.

By: Daniel Rosehill (danielrosehill.com)
24-09-24