Problem Statements for the Platform

Problem Identification

1. **Complexity in Locating Relevant Datasets:**
   QR need access to diverse datasets across domains, but identifying relevant datasets quickly and accurately is challenging due to the fragmented nature of data sources.
2. **Lack of Metadata or Contextual Information for Data Interpretation:**
   The absence of detailed metadata or contextual descriptions hinders researchers from understanding the data's origin, structure, or potential biases.
3. **Difficulty in Transforming and Preparing Data Efficiently:**
   Raw datasets require extensive cleaning, transformation, and integration with other datasets before they can be used in modeling or analysis.
4. **Limited Options for Accessing Both Raw and Processed Data:**
   QR needs both raw data for custom transformations and processed data for immediate use, but platforms often lack flexibility in offering both types of access seamlessly.

Platform Solutions

The platform addresses these challenges through its core components: **`DataCatalog`**, **`DataLake`**, **`DataWorkbench`**, and **`QuandlDataManager`**, offering an integrated solution tailored to QRs' needs.

1. **Complexity in Locating Relevant Datasets:**

`DataCatalog` provides centralized access to datasets with the following functionalities:

- `add_category()`: Interacts with **`DataCategory`** to assign category attribute to dataset.
- `search_datasets()`: Performs keyword-based searches across multiple categories for datasets.

2. **Lack of Metadata or Contextual Information for Data Interpretation:**

`QuantlDataManager` addresses this challenge with `fetch_quandl_table()` which enables tailored queries, integrates with Quandl's documentation, and provides clear feedback. This ensures that researchers receive relevant and interpretable data.

3. **Difficulty in Transforming and Preparing Data Efficiently:**

`DataWorkbench` provides robust tools for data preparation:

- `clean_data()`: Handles the null value, missing columns, duplicates, and anomalies to ensure the data is clean and structured enough for the subsequent analysis.
- `prepare_data()`: Integrates with **`DataLake`** and combines data retrieval, cleaning, and preprocessing into a single step to streamline the entire data preparation workflow into high-level, reusable methods.

4. **Limited Options for Accessing Both Raw and Processed Data:**

`DataLake` offers dual access to data, catering to diverse research needs:

- `self.raw_data, self.processed_data`: Initialized to provide storage for next access.
- `store_data()`: Based on the processed flag, saves the dataset into either raw_data or processed_data dictionaries
- `retrieve_data()`: Interacts with `store_data()`, raising signals for the data categorization.