**Technical Memo 1: Variability in Data Persistence Models**

**Issue:** Inability to support more than one data persistence model by the system.

**Problem**

The library management system needs to persist data in several formats, supporting both relational and document-based models for various use cases and scalability in the future.

**Summary of Solution**

It should abstract a flexible layer that will support multiple data models, ranging from Relational MySQL to SQL Server to document stores such as MongoDB and Redis. This layer shall provide runtime selection and configuration.

**Factors**

* Multi-Storage Requirements: Not all data models are created equal. Relational databases are best suited to handle structured data (data in predefined format) containing complex, interrelated relationships, whereas document stores are better suited for unstructured data.
* Scalability: Using multiple models of data would make the system more scalable easily because different services might require different types of storage.
* Adaptability: It should have a modular in nature data access layer to allow changes in the foreseeable future without refactoring code on a large scale.

**Solution**

Design Strategy based data persistence module that can switch at runtime depending on the configuration chosen. Utilize Factory pattern to create instances of specific database connections as required.

**Motivation**

This will provide room for flexibility and future-proofing, thus making the library system evolve with changing dynamics of storage.

**Alternatives**

* Use one data model throughout the system, which will limit the system to either relational or document-based storage.
* Separate codebases for each of the data models - will increase complexity in their maintainability.

**Pending Issues**

* Transaction consistency across varying types of databases.
* Rules of thumb regarding where to use each type of database.