# **Database Choice: SQL vs NoSQL for ResearchNest**

## **1. Introduction**

### The ResearchNest prototype requires a robust backend database to manage the **hierarchical progress tracker** (milestones → stages → tasks → subtasks) and **student profiles**. The team evaluated both **SQL (relational databases)** and **NoSQL (MongoDB)** to determine the most suitable option for this project.

### Although NoSQL offers flexibility, the decision was made to proceed with **SQL** for the following reasons:

### Familiarity and ease of use for the team.

### Strong data integrity and ACID compliance.

### Simplified reporting and querying for academic evaluation.

### 

## **2. SQL Databases**

### **2.1 Advantages of SQL**

### **Structured Data Modeling**: Hierarchical progress can be normalized into tables with foreign key relationships, ensuring clear and enforceable constraints.

### **Data Integrity (ACID Compliance)**: SQL ensures that updates such as task completion cascade reliably, reducing risk of inconsistent data.

### **Powerful Querying**: SQL provides advanced querying (JOIN, GROUP BY, ORDER BY) which simplifies analytics such as “List all students who completed the Thesis Submission milestone.”

### **Familiarity**: The team already has prior knowledge of SQL, reducing the learning curve and development time.

### **Better for Reporting**: Faculty/Admin dashboards often require tabular data and aggregate reports, which SQL supports effectively.

### **2.2 Limitations of SQL**

### **Rigid Schema**: Schema modifications (e.g., adding a new level of hierarchy) require ALTER statements and migrations.

### **Complex Joins**: Retrieving an entire milestone–stage–task–subtask tree requires multiple joins, which can increase query complexity.

### **Scalability**: Traditional SQL systems rely on vertical scaling, which may be less flexible for very large-scale systems.

### 

## **3. NoSQL Databases (MongoDB)**

### **3.1 Advantages of NoSQL**

### **Flexible Schema**: Supports unstructured and semi-structured data. Hierarchical data like milestones and subtasks can be stored as nested JSON within a single document.

### **Faster Prototyping**: Minimal setup is required, making it easier to quickly build prototypes.

### **Efficient for Nested Data**: Fetching a student’s full progress requires only one query since milestones, stages, tasks, and subtasks can be embedded.

### **Scalability**: Designed for horizontal scaling, suitable for large-scale distributed systems.

### **3.2 Limitations of NoSQL**

### **Weaker Transaction Guarantees**: Although MongoDB supports transactions, its consistency model is not as strict as SQL’s ACID compliance.

### **Complex Reporting**: Aggregate pipelines are less intuitive than SQL’s relational queries, making reporting harder for academic use cases.

### **Learning Curve**: Requires a shift from relational thinking, which may slow down development for teams more comfortable with SQL.

### 

## **4. Why SQL Was Chosen for ResearchNest**

### Although MongoDB is naturally suited for hierarchical structures, SQL was selected as the preferred choice for this prototype because:

### **Academic Fit**: SQL databases are more widely taught, understood, and evaluated in academic contexts. Demonstrating ER diagrams, normalized schemas, and SQL queries aligns with project assessment requirements.

### **Team Expertise**: The team has greater familiarity with relational databases, which reduces risk and speeds up implementation.

### **Data Integrity & Consistency**: Strong ACID guarantees ensure that status changes (e.g., completing all subtasks → completing the parent task) are reliably enforced.

### **Reporting Needs**: Faculty and Admin dashboards require structured reports, which are easier to implement in SQL.

### **Prototype Scope**: The project is not targeting production-scale performance; hence, the scalability benefits of NoSQL are not critical.

### 

## **5. Conclusion**

### Both SQL and NoSQL have distinct strengths. For highly scalable, flexible, and unstructured applications, **NoSQL** provides advantages in handling nested data. However, for the **ResearchNest prototype**, which prioritizes **data integrity, reporting, academic evaluation, and team efficiency**, **SQL is the most suitable choice**.

### Thus, the project will be implemented using a **SQL relational database**, ensuring structured relationships, reliable progress tracking, and clear reporting for both students and faculty.

### 