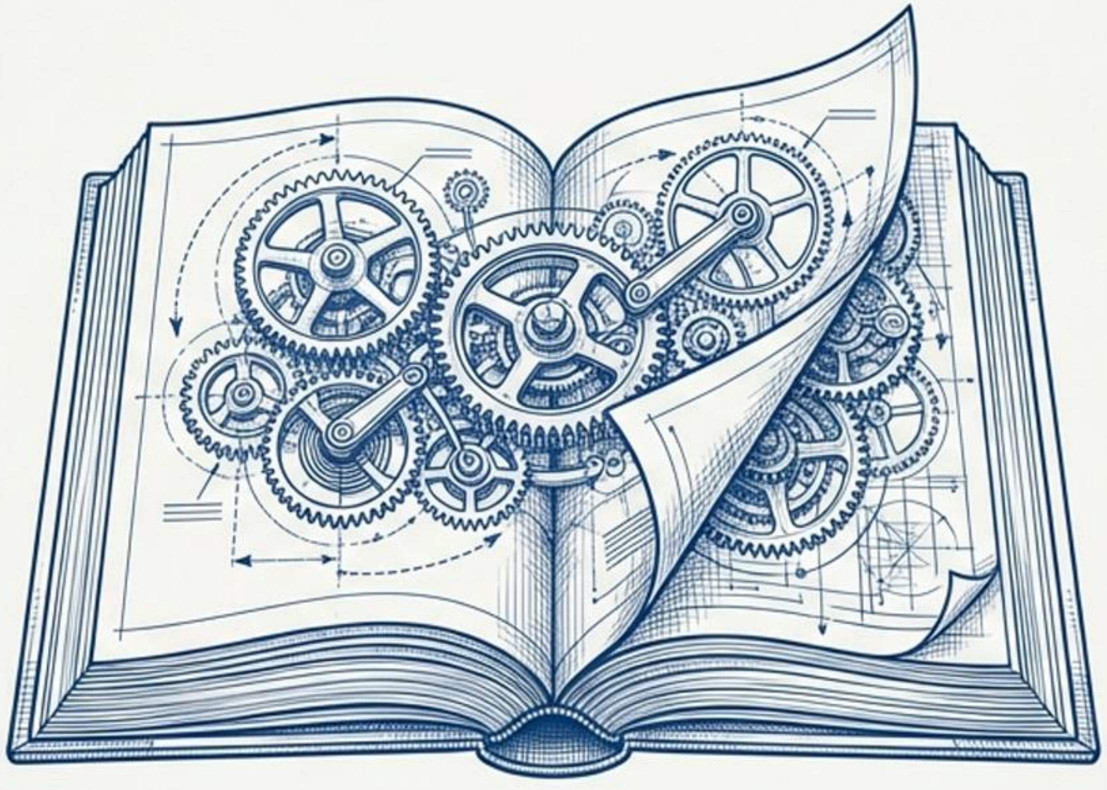


Building the Engine: A T-SQL Showcase for Library Management

A comprehensive database solution for automating workflows, analyzing performance, and driving business insights.

A Modern Library is a Complex Data Ecosystem

The challenge was to move beyond simple data storage and build a system to actively manage inventory, track member activity, process transactions, and generate financial insights in a simulated, real-world environment.

Inventory Management

How do we track thousands of books across multiple branches in real-time?

Member Activity

How can we monitor borrowing patterns, identify active users, and manage risk?

Operational Workflow

How can we automate routine tasks like book issuance and returns to reduce errors and improve efficiency?

Business Performance

How do we measure revenue, identify top-performing branches, and calculate overdue fines accurately?

The Mission: To Engineer a Solution for Four Core Objectives

The project was designed to solve 20 specific business problems by achieving four primary goals:

1. Normalize and Organize Data

Establish a robust relational schema to serve as a single source of truth for all library operations, connecting Books, Members, Employees, and Branches.

2. Automate Core Workflows

Utilize Stored Procedures to handle high-volume transactions like book issuance and returns, ensuring data integrity and real-time status updates.

3. Analyze Business Performance

Develop queries to track key metrics such as rental revenue by category, branch-level performance, and overdue fines.

4. Monitor Member Activity

Create systems to identify key member segments, including highly active users and those with overdue or damaged books.

The Blueprint: A Relational Schema for a Single Source of Truth

The system is built on a normalized database architecture to ensure data integrity and efficient querying.

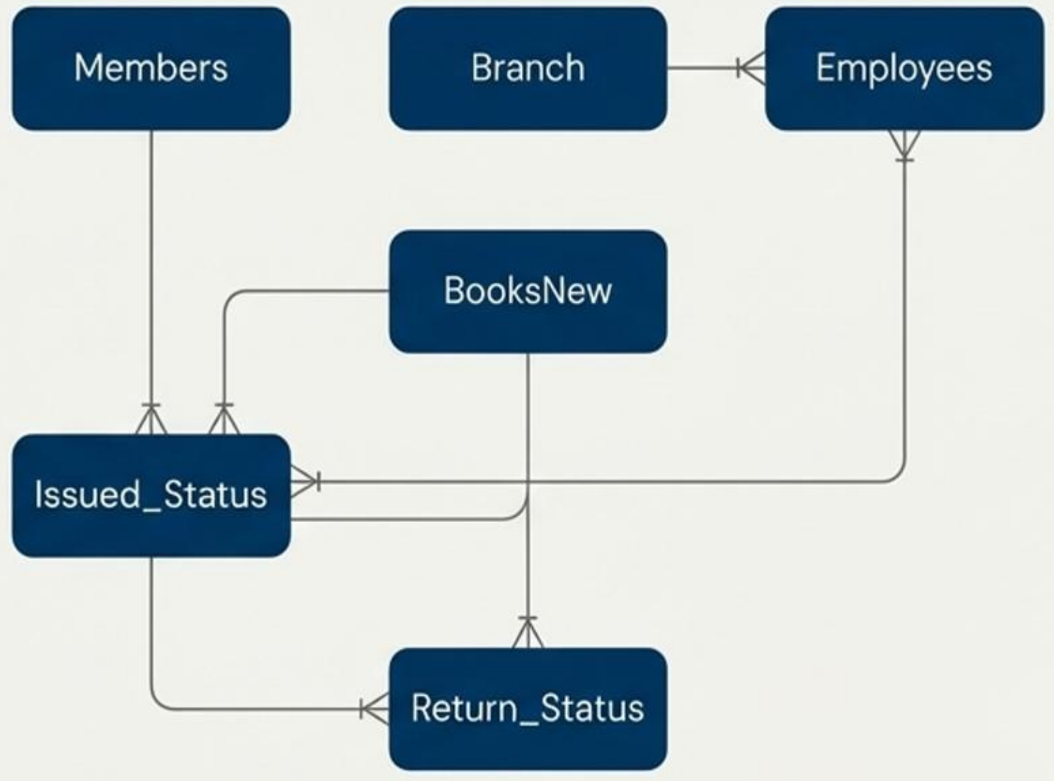
BooksNew: The central inventory (ISBN, title, category, price, status).

Members: Customer profiles and registration data.

Branch, Employees: Staff and location information.

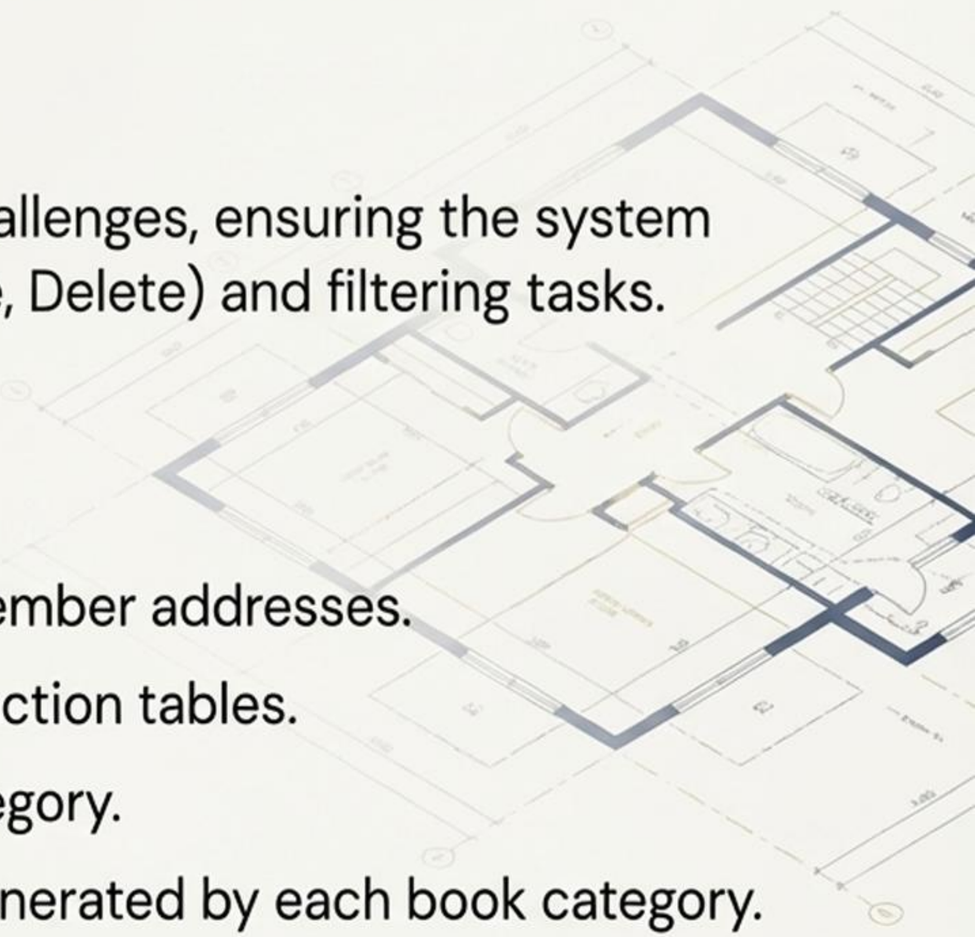
Issued_Status: The primary transaction table linking members, books, and employees for every loan.

Return_Status: Tracks returns, dates, and the condition of the book.



Solution Layer 1: Establishing Foundational Control with Basic Operations

The first phase involved solving core data management challenges, ensuring the system could reliably handle essential CRUD (Create, Read, Update, Delete) and filtering tasks.



Key Examples of Problems Solved

- ✓ **Data Entry:** Inserted new book records and updated member addresses.
- ✓ **Data Integrity:** Removed erroneous records from transaction tables.
- ✓ **Basic Reporting:** Retrieved all books in the ‘Classic’ category.
- ✓ **Financial Calculation:** Calculated total rental income generated by each book category.
- ✓ **Staff Management:** Listed employee details alongside their branch manager’s name.

Solution Layer 2: Unlocking Insights with Intermediate Analysis

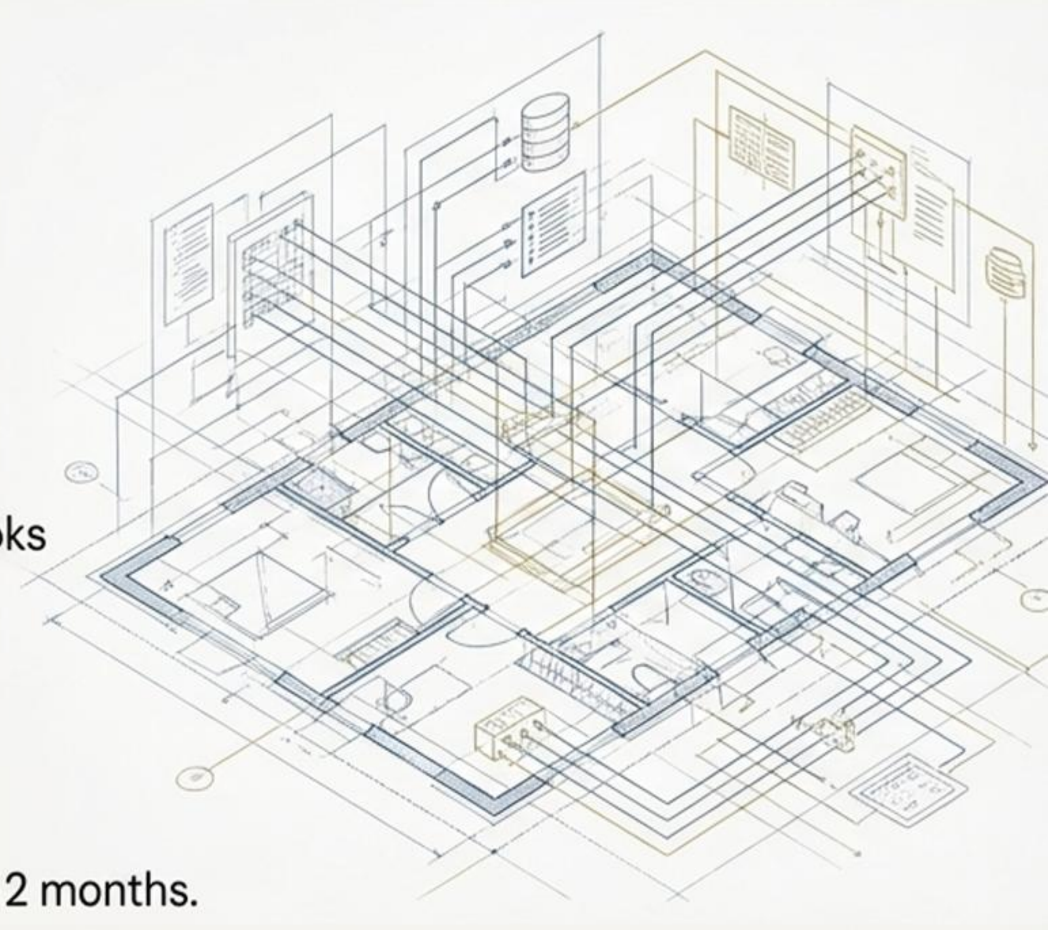
Moving beyond basic queries, this layer focused on using Joins, Aggregations, and CTAS (Create Table As Select) to generate performance reports and analytical snapshots.

Key Examples of Analysis Performed

- ✓ Member Activity Analysis: Identified members who have issued more than one book.
- ✓ Branch Performance Reporting: Created a report showing total books issued, returned, and revenue per branch.
- ✓ Overdue Book Identification: Isolated all transactions where books were not returned within the 30-day window.

Creating Analytical Tables (CTAS):

- **`book_issued_cnt`**: A summary table of issuance frequency.
- **`active_members`**: A pre-filtered table of users active in the last 2 months.



Solution Layer 3: Automating Intelligence with Advanced SQL

The highest level of implementation involved using Stored Procedures, CTEs, and Window Functions to automate complex business logic, manage risk, and identify top performers.

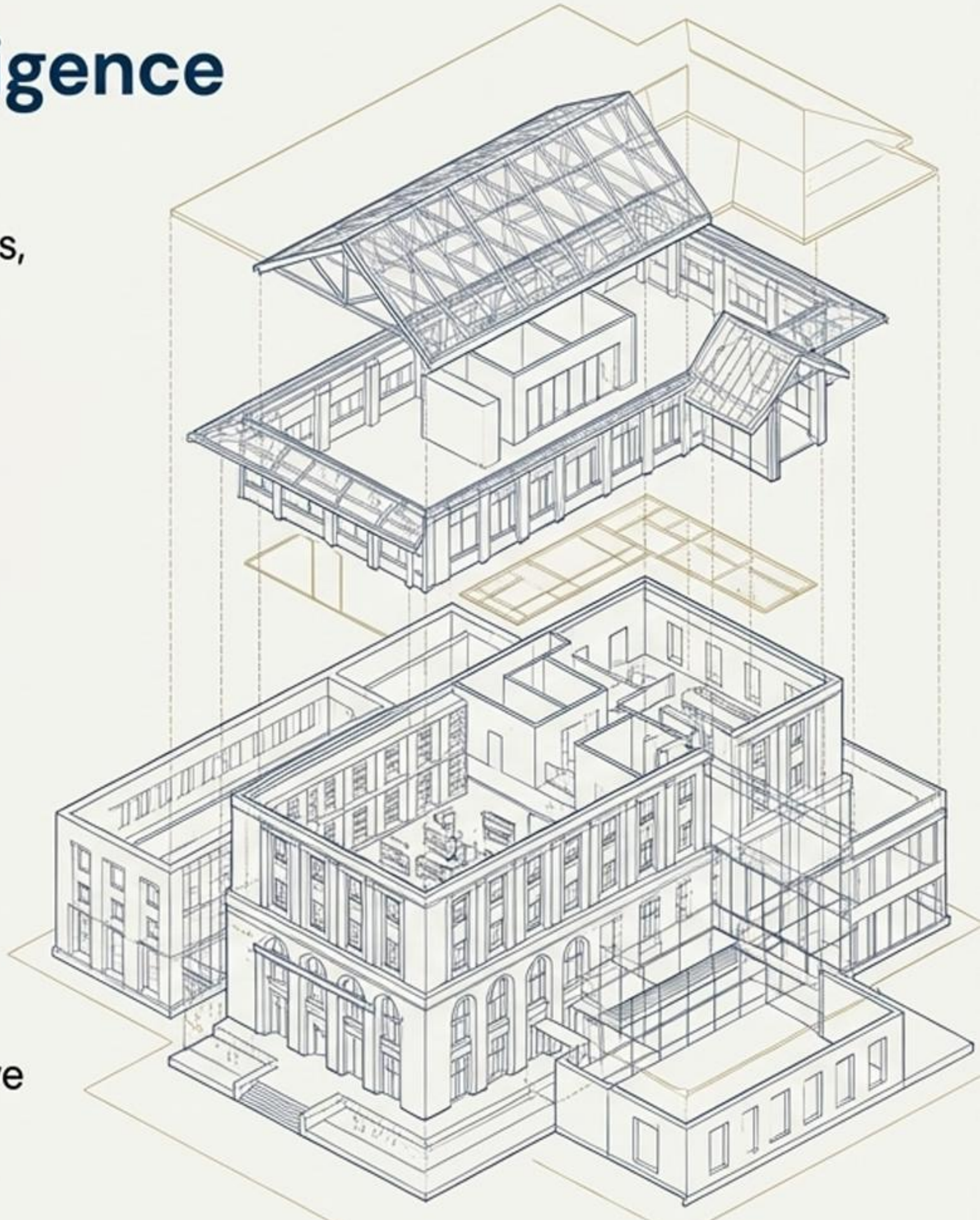
Key Advanced Implementations

Automated Transactions: Created Stored Procedures to safely issue and return books.

Performance Ranking: Used window functions to find the top 3 employees by number of book issues.

Proactive Risk Management: Built a CTE to identify 'High-Risk Members' who repeatedly return damaged books.

Automated Financial Penalties: Developed a comprehensive query to automatically calculate and report overdue fines.



The Engine Room, Part 1: An Automated, Error-Proof Book Issuance Procedure

To ensure data integrity and prevent double-booking, a Stored Procedure was created to govern the entire book issuance process. It acts as a gatekeeper for the inventory.

```
CREATE OR ALTER PROCEDURE issue_book (...)
AS
BEGIN
   DECLARE @v_status VARCHAR(10);
   -- Check status
   SELECT @v_status = status FROM BooksNew
     WHERE isbn = @p_issued_book_isbn;

   IF @v_status = 'yes'
   BEGIN
      INSERT INTO issued_status(...)
      VALUES (...);

      UPDATE BooksNew SET status = 'no'
      WHERE isbn = @p_issued_book_isbn;

      PRINT 'Book issued successfully.';
   END
ELSE
   BEGIN
     PRINT 'Book is currently unavailable.';
   END
END;
```

Verification Step: Checks the live status of the book *before* any action is taken.

Business Rule: The core logic gate. The transaction only proceeds if the book is available.

Record Creation: Atomically creates the new loan record.

Real-Time Inventory Update: Immediately marks the book as unavailable to prevent conflicts.

Error Handling: Provides clear feedback if the business rule is not met.

TheEngine Room, Part 2: Proactively Identifying High-Risk Members with a CTE

A Common Table Expression (CTE) was used to create a temporary, logical result set identifying members who consistently return books in damaged condition. This allows the library to move from reactive fees to proactive policy enforcement.

Scan `Return_Status` for “Damaged” entries.

→

Group by `member_id` and count occurrences.

→

Join with `Members` table to get member details.

→

Filter for members with more than a set threshold of damaged returns.

This demonstrates using SQL not just for reporting what *has* happened, but for identifying patterns that predict future risk.

The Engine Room, Part 3: A Precise Engine for Automated Fine Calculation

An automated query was built to generate a **comprehensive report of all overdue books**, calculating the precise fine based on a rate of $0.50 per day past the 30-day lending window.

```sql
SELECT m.member_id,
  SUM((DATEDIFF(DAY, ist.issued_date,
        GETDATE()) - 30) * 0.50)
    AS total_fines
FROM
  issued_status ist
  JOIN members m
    ON ist.issued_member_id = m.member_id
  LEFT JOIN return_status rs
    ON ist.issued_id = rs.issued_id
WHERE
    rs.issued_id IS NULL
  AND DATEDIFF(DAY, ist.issued_date,
       GETDATE()) > 30
GROUP BY
  m.member_id;
```

Calculates Financial Penalty: The core calculation logic, precisely applying the daily fine to the number of overdue days.

Finds Unreturned Books: Isolates only the loans that are still open.

Applies Business Rule: Filters for books that are officially overdue.

Aggregates Fines: Rolls up all overdue books to a single total fine per member.

The Performance Dashboard: Translating Technical Execution into Business Impact

The implementation of the T-SQL engine delivered measurable improvements and actionable insights across four key areas of library operations.

Revenue Optimization

Identified most profitable genres to guide purchasing strategy.

Operational Efficiency

Reduced manual errors and provided real-time inventory accuracy.

Risk Management

Enabled proactive policies for high-risk members.

Performance Monitoring

Highlighted disparities in branch workload and revenue.

Impact Delivered: Optimized Revenue and Enhanced Operational Efficiency

Revenue Optimization

Insight:

Analysis of rental income by category revealed which book genres are the most profitable.

Actionable Outcome:

The library can now make data-driven decisions on budget allocation for future book purchases, maximizing return on investment.

Operational Efficiency

Insight:

The Stored Procedures for issuing and returning books eliminated the potential for manual data entry errors.

Actionable Outcome:

The system maintains 100% real-time inventory accuracy, preventing situations where unavailable books are promised to members.

Impact Delivered: Mitigated Risk and Enabled Strategic Resource Allocation

Risk Management

Insight:

By identifying members who frequently return damaged books or hold overdue items, a clear high-risk profile was established.

Actionable Outcome:

The library can enforce stricter lending policies or require security deposits for these specific accounts, protecting its assets.

Branch Performance

Insight:

The branch performance report highlighted significant disparities in workload and revenue generation between locations.

Actionable Outcome:

Management now has the data needed to justify resource redistribution (e.g., staffing, inventory) between high-traffic and low-traffic branches.

Core Competencies and Technology Stack

Database Engine

Microsoft SQL Server (T-SQL)

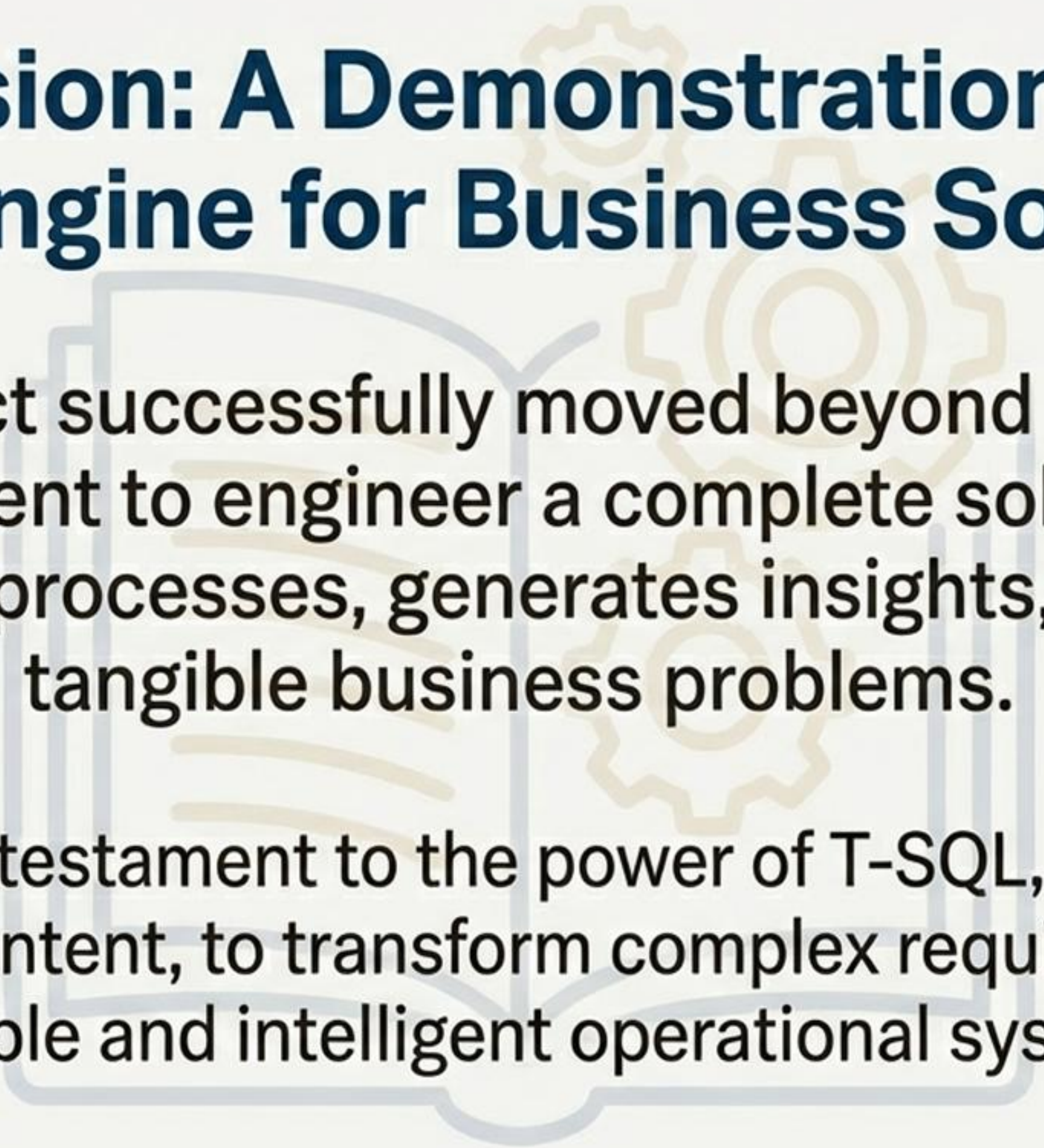
Development Tools

SQL Server Management Studio (SSMS)

Key Concepts Demonstrated

- Stored Procedures
- Common Table Expressions (CTEs)
- Create Table As Select (CTAS)
- Window Functions
- Complex Joins & Aggregations
- Database Schema Design
- Triggers

Conclusion: A Demonstration of SQL as an Engine for Business Solutions



This project successfully moved beyond basic data management to engineer a complete solution that automates processes, generates insights, and solves tangible business problems.

It stands as a testament to the power of T-SQL, when applied with strategic intent, to transform complex requirements into a reliable and intelligent operational system.

THANK YOU