Introduction

The COVID-19 pandemic introduced unprecedented disruptions to global manufacturing and supply chain operations, forcing organizations to reevaluate the economic resilience and operational efficiency of their core processes. In response, the application of Data Science and Business Intelligence (BI) tools has become increasingly vital in modeling and understanding the economic impact of process improvements before and after the crisis. This research investigates how these technologies can quantify financial benefits, support strategic decision-making, and optimize continuous improvement initiatives across industrial sectors.

To support this exploration, a relational database was created using MySQL Workbench. The goal was to build a structured and scalable data environment to capture essential variables associated with improvement projects—including project metadata, key performance indicators (KPIs), and cost-saving metrics. This allows for clear analysis of economic outcomes both pre- and post-COVID-19, enabling insights into how business performance metrics evolved in response to crisis-induced transformations.

The following tasks were completed:

- A new schema named mydatabase was created.
- Three interrelated tables (Projects, KPI_Tracking, and Cost_Savings) were designed, each including a mix of string, numeric, and date/time fields.
- Foreign key relationships were used to ensure normalization and relational integrity.
- At least three sample data rows were inserted into each table.

- A schema diagram was generated via reverse engineering in MySQL Workbench to visualize the entity-relationship structure.
- Screenshots of the database, including sample data, were taken to document the implementation process.

This structured database not only provides the foundation for conducting robust economic modeling using BI tools, but also contributes to data-driven insights that support strategic planning and ROI analysis in post-pandemic manufacturing environments.

The file was clicked, and then a new query tab was selected as showed in Fig.1.

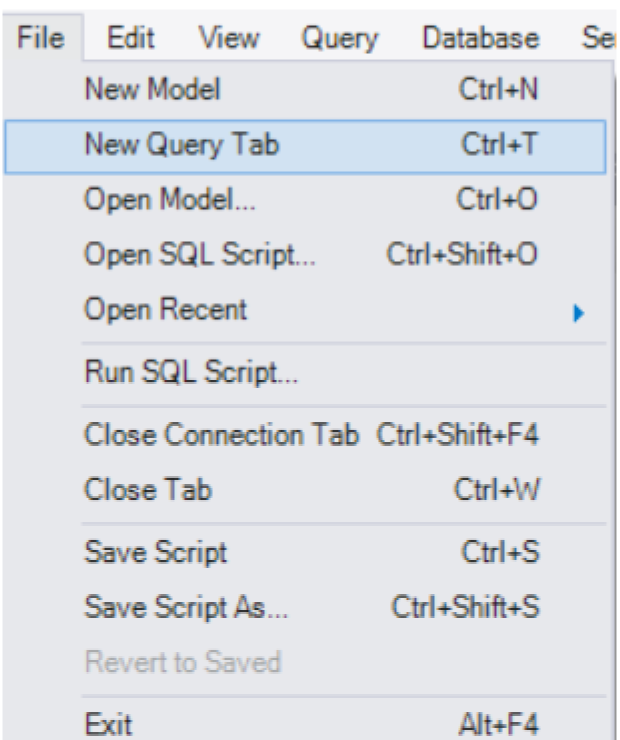

Fig. 1. File tab to get to the query tab.

On the query tab(SQL File 12 in this case), “create Schema mydatabase” was added, as shown in Fig. 2

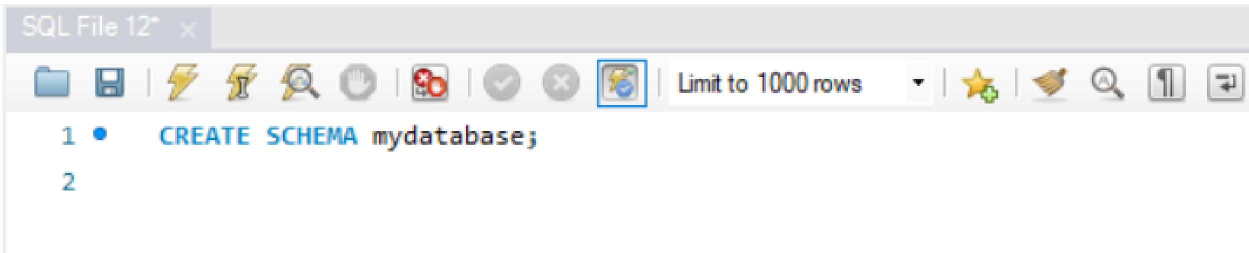

Fig. 2 Creation of “mydatabase” schema in the SQL File 12 query

The output was positive, as Fig. 3 indicates. Also, the green check mark shows that there was no error or issue.

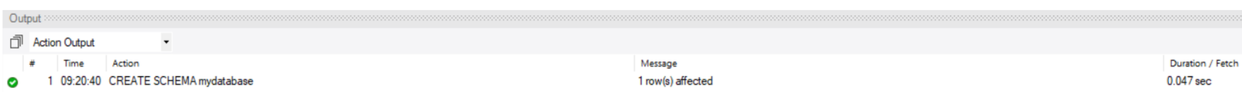

Fig. 3 The output showing Schema creation

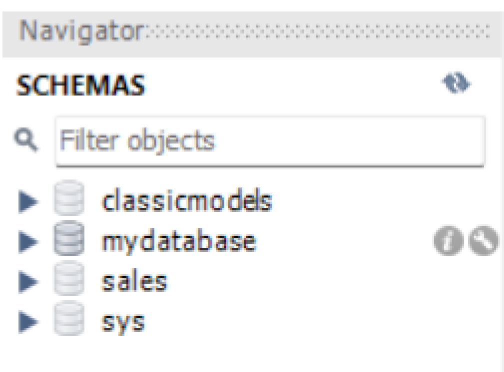

Fig. 4. Mydatabase appearance after refreshing  the schema

Mydatabse was expanded to see what it contained. As one can see in Fig. 5, there are tables, views, stored procedures, and functions

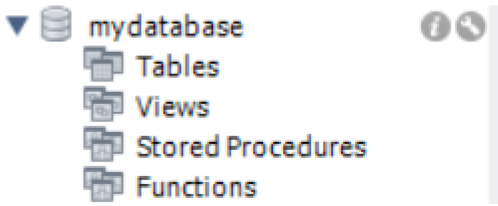

Fig. 5 mydatabase content

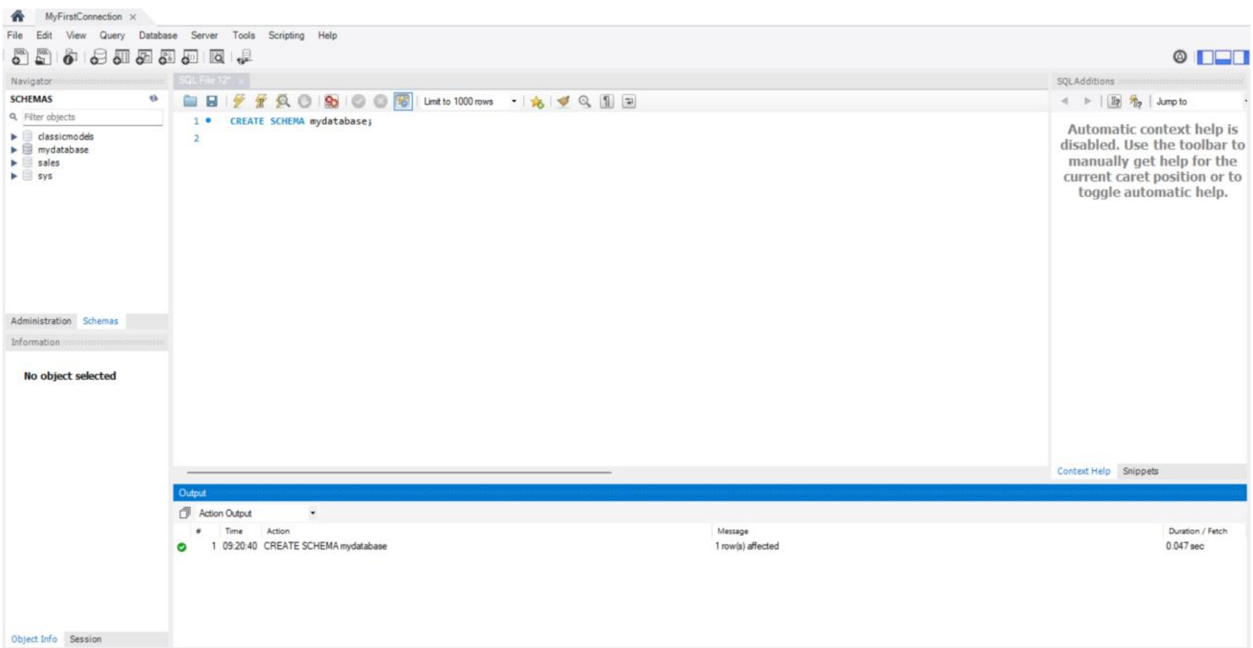

Fig. 6. First schema creation

The use of my database and then the creation of Projects

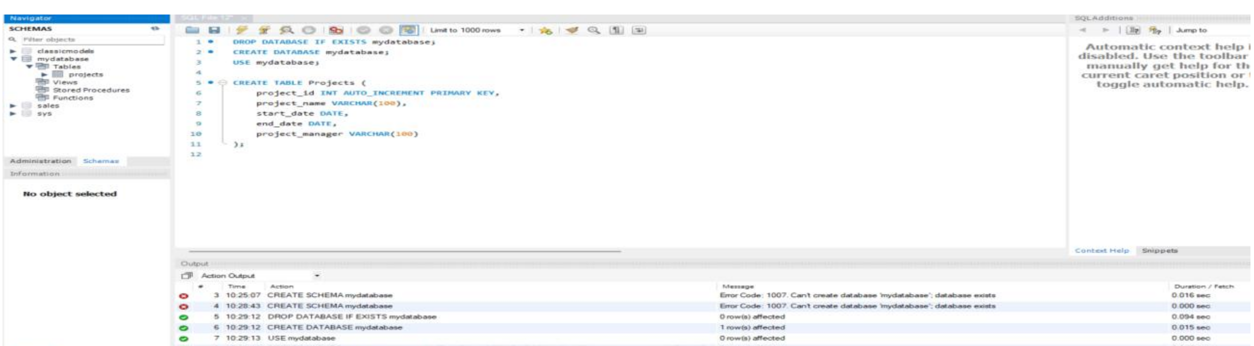

Fig 7.

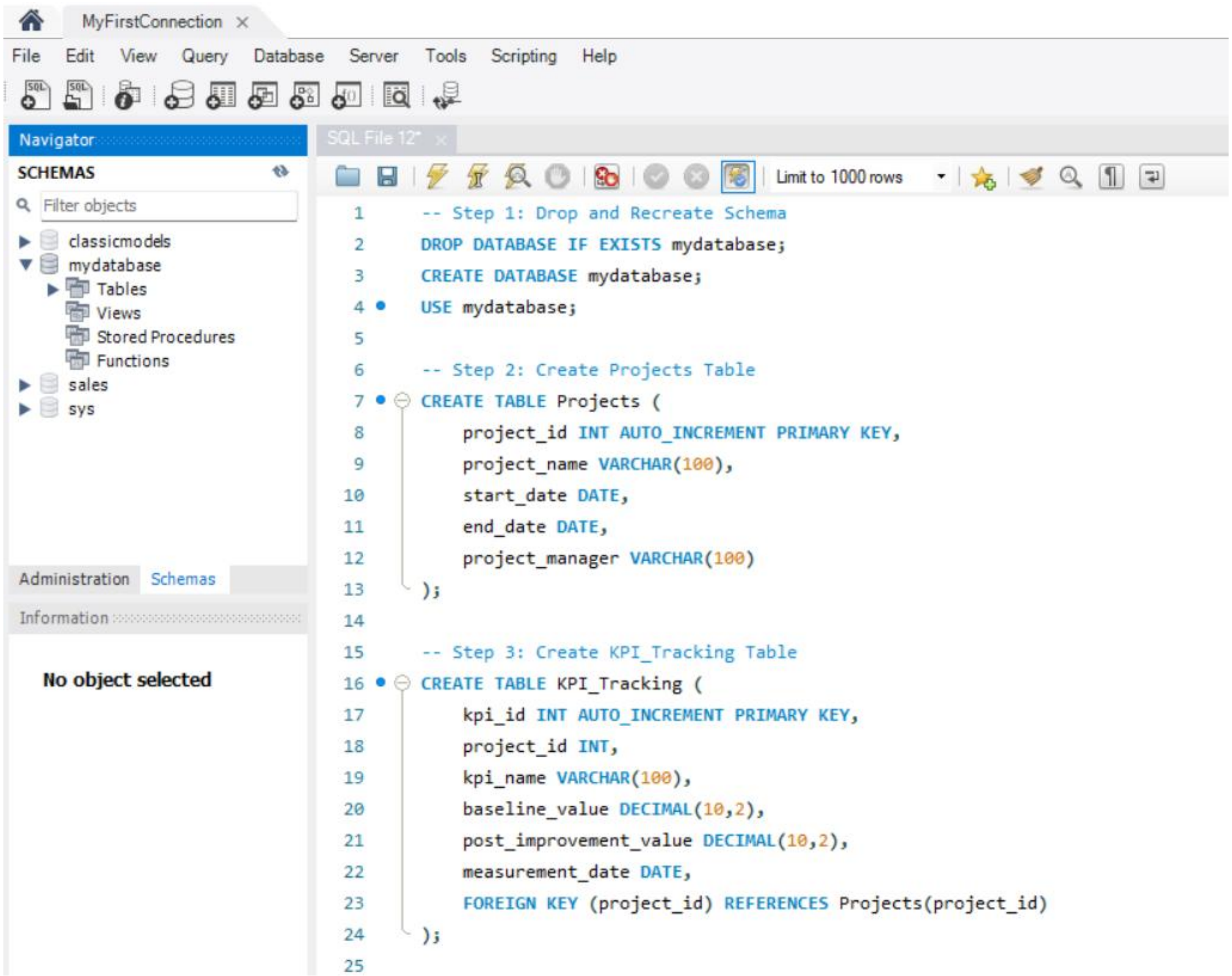

Fig 8a. addition of KPI_tracking table

```sql
-- Step 4: Create Cost_Savings Table
CREATE TABLE Cost_Savings (
    savings_id INT AUTO_INCREMENT PRIMARY KEY,
    project_id INT,
    estimated_savings DECIMAL(12,2),
    actual_savings DECIMAL(12,2),
    roi_percent DECIMAL(5,2),
    recorded_date DATE,
    FOREIGN KEY (project_id) REFERENCES Projects(project_id)
);

-- Step 5: Insert Sample Data into Projects
INSERT INTO Projects (project_name, start_date, end_date, project_manager) VALUES
('Lean Line Balancing', '2023-01-15', '2023-06-30', 'Alice Johnson'),
('Waste Reduction Initiative', '2023-03-01', '2023-08-15', 'Bob Smith'),
('Cycle Time Optimization', '2023-04-10', '2023-12-01', 'Claire Adams');

-- Step 6: Insert Sample Data into KPI_Tracking
INSERT INTO KPI_Tracking (project_id, kpi_name, baseline_value, post_improvement_value, measurement_date) VALUES
(1, 'Production Rate (units/hr)', 120.00, 150.00, '2023-07-01'),
(2, 'Defect Rate (%)', 5.50, 2.30, '2023-08-20'),
(3, 'Cycle Time (min)', 45.00, 30.00, '2023-12-10');

-- Step 7: Insert Sample Data into Cost_Savings
INSERT INTO Cost_Savings (project_id, estimated_savings, actual_savings, roi_percent, recorded_date) VALUES
(1, 50000.00, 52000.00, 18.50, '2023-07-05'),
(2, 30000.00, 28000.00, 12.00, '2023-08-25'),
(3, 70000.00, 74000.00, 25.00, '2023-12-15');
```

Fig 8b. Addition of cost_savings and the sample data

After running the code, refreshing the schema we were able to expand the database as described in Fig 9.

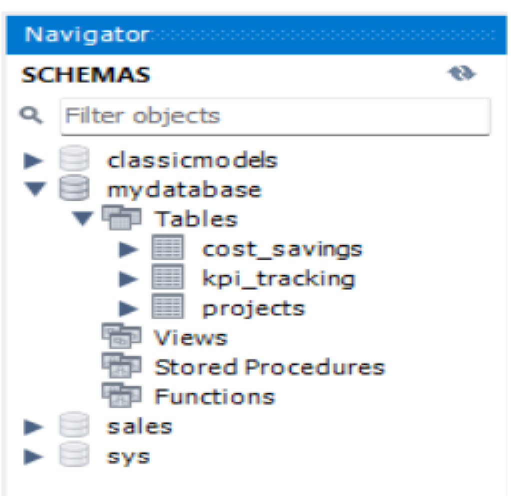

Fig 9. Content of tables in mydatabase

Next Step: Reverse Engineer for Diagram

1. Go to Database → Reverse Engineer
2. Select your local MySQL connection
3. Select mydatabase
4. Click Next → Execute → Next → Finish
5. You'll see the EER (Enhanced Entity Relationship Diagram)

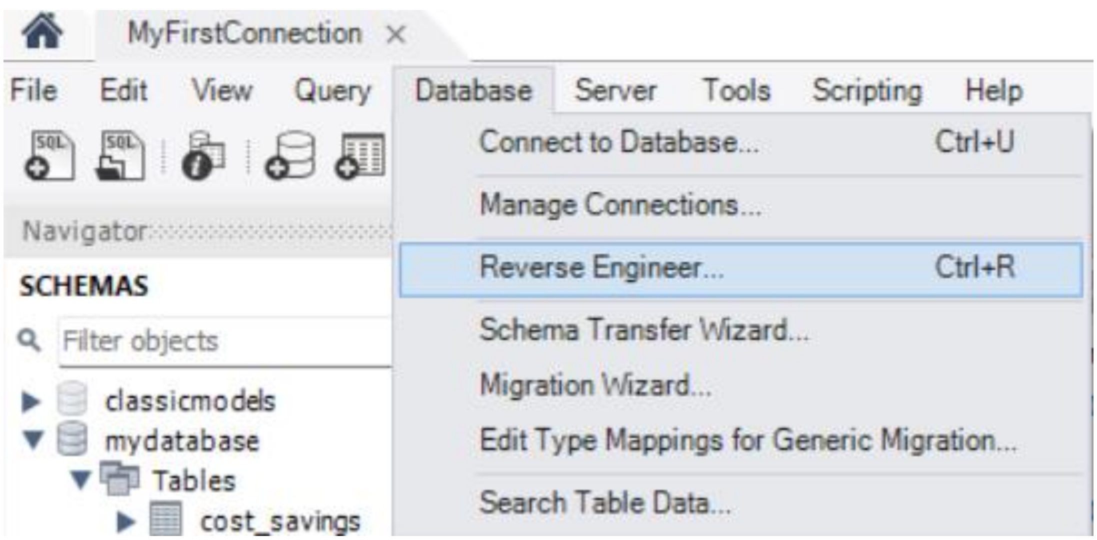

Fig 10. A way to visualize reverse engineering tab

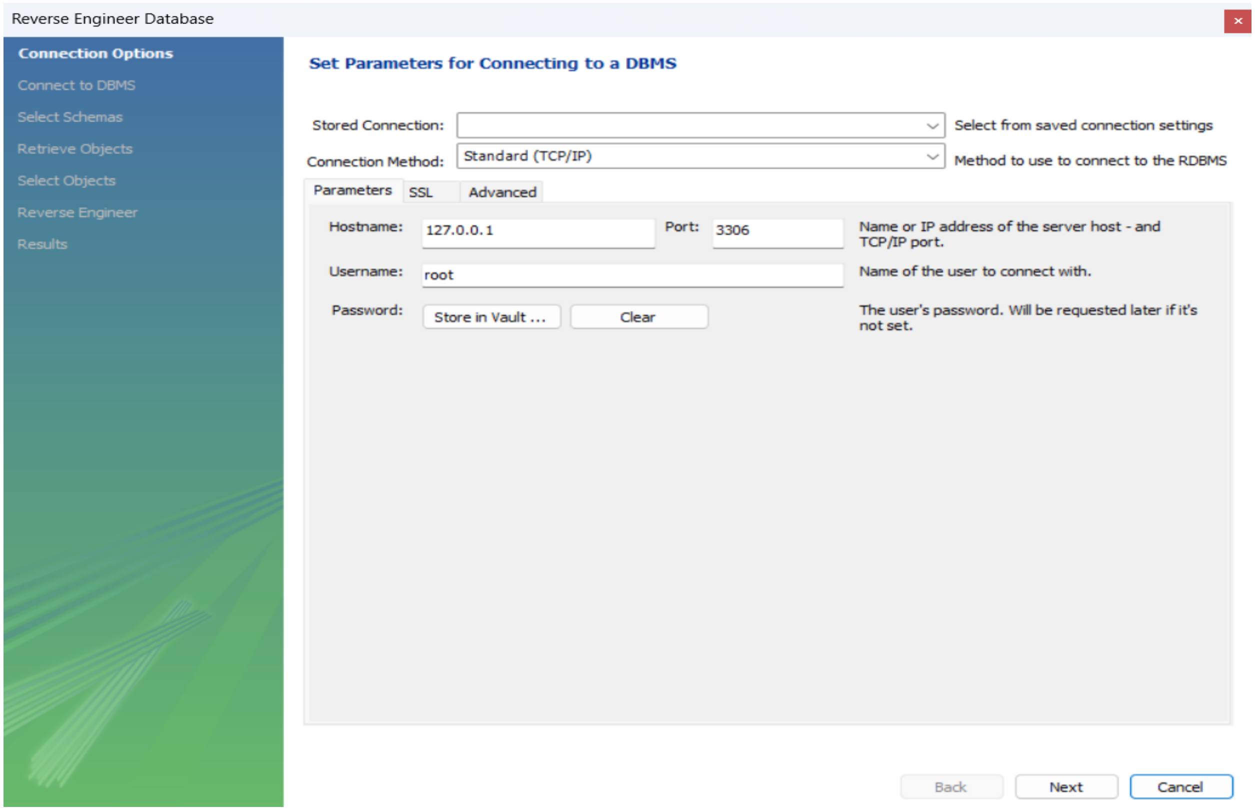

Fig 11. Reverse engineering database visualization

After the use of stored connection, there was "MyFirstConnection" was selected since the location where the query tab was created.

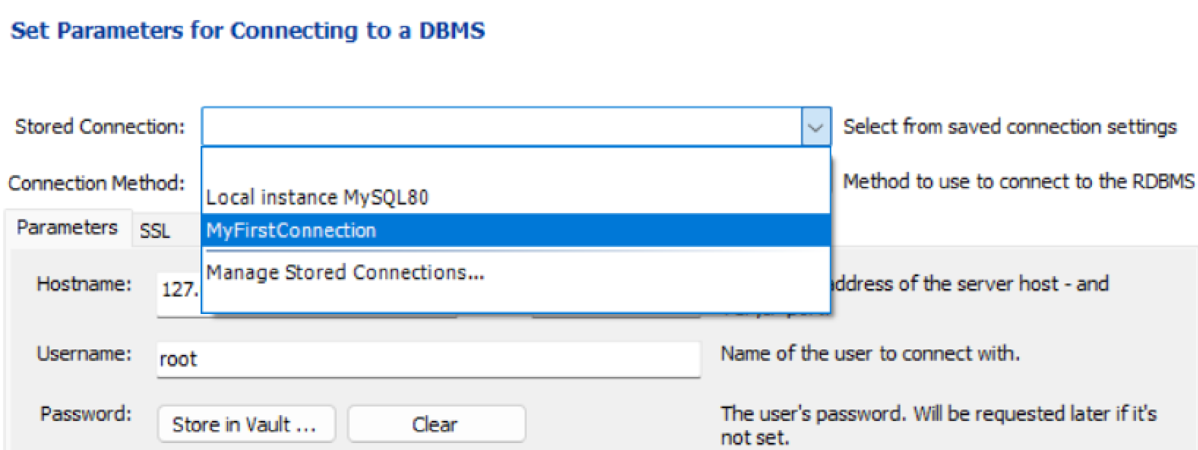

Fig.12 Set Parameters for connecting to a DBMS

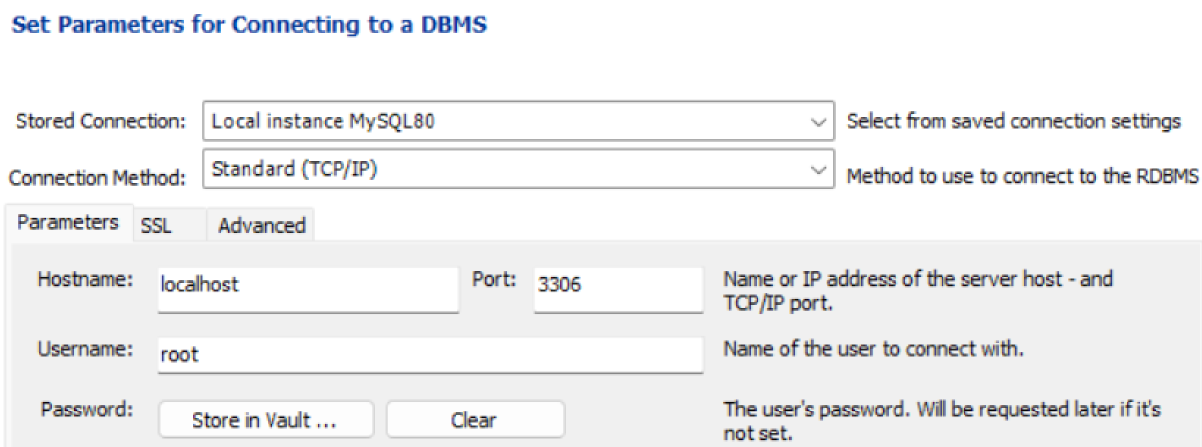

Fig 13. Local instance MySQL80 selection

After the selection, the Next option was presses and we got:

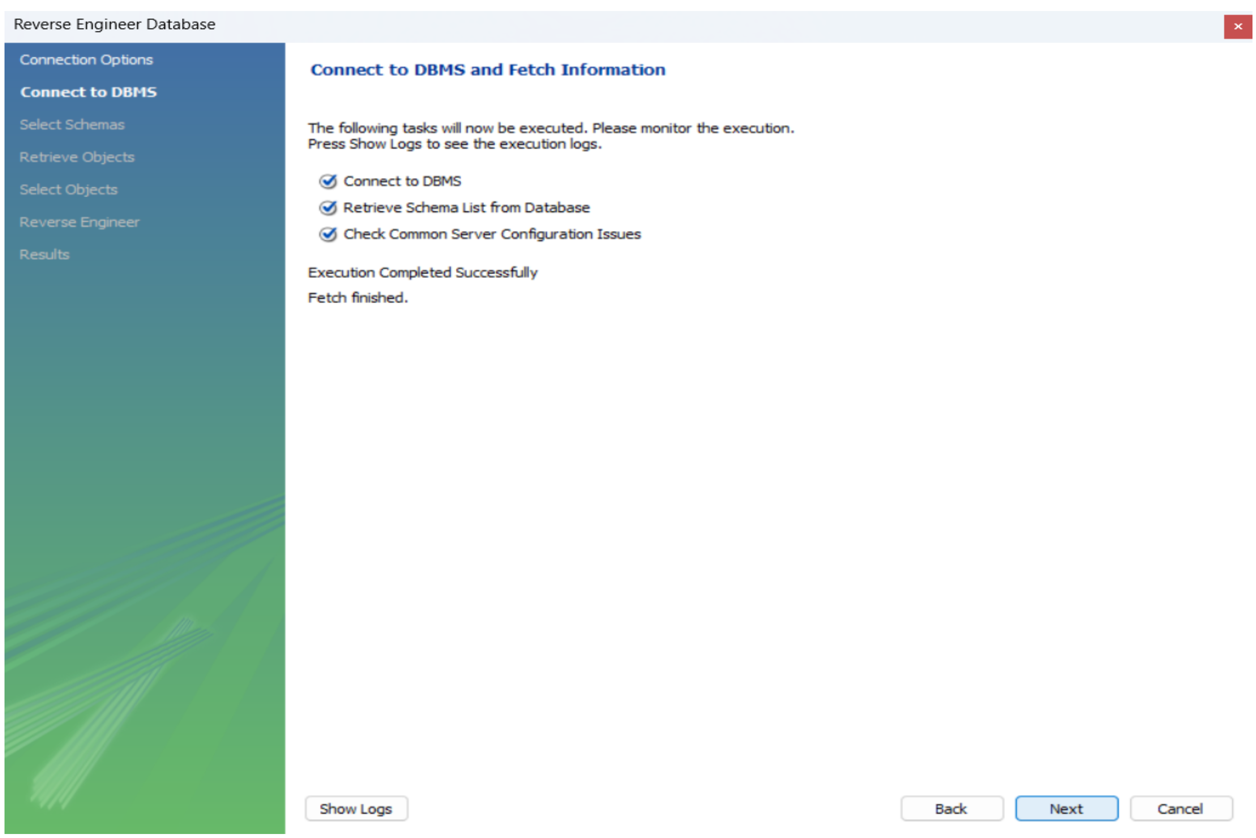

Fig 14. Way to connect with the DataBase Management System (DBMS)

The selection of the schema helped checked on mydatabase as shown in Fig 15.

Fig. 15. Mydatabase selection

Next, the retrieval was completed successfully as described in Fig. 16.

Fig 16. The retrieve and reverse engineer Schema Objects

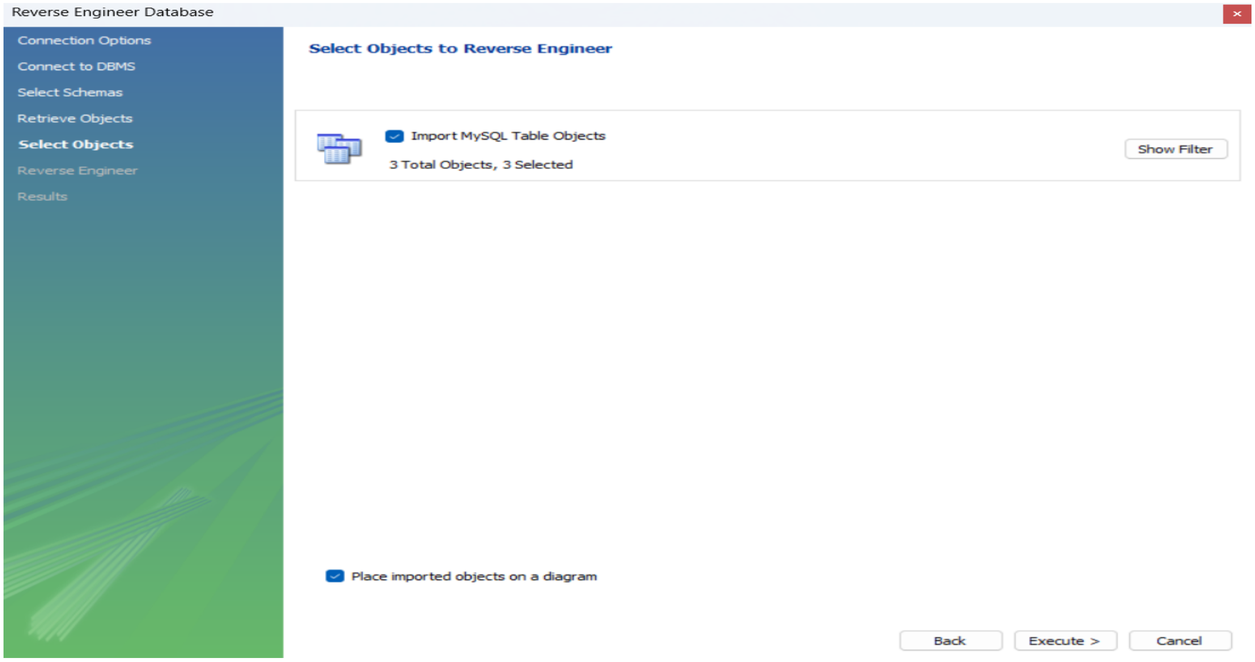

Fig 17. Way to Import MySql Table Object

The Execute tab at the bottom of Fig 17 was clicked next to get to the reverse engineer progress as seen in Fig 18.

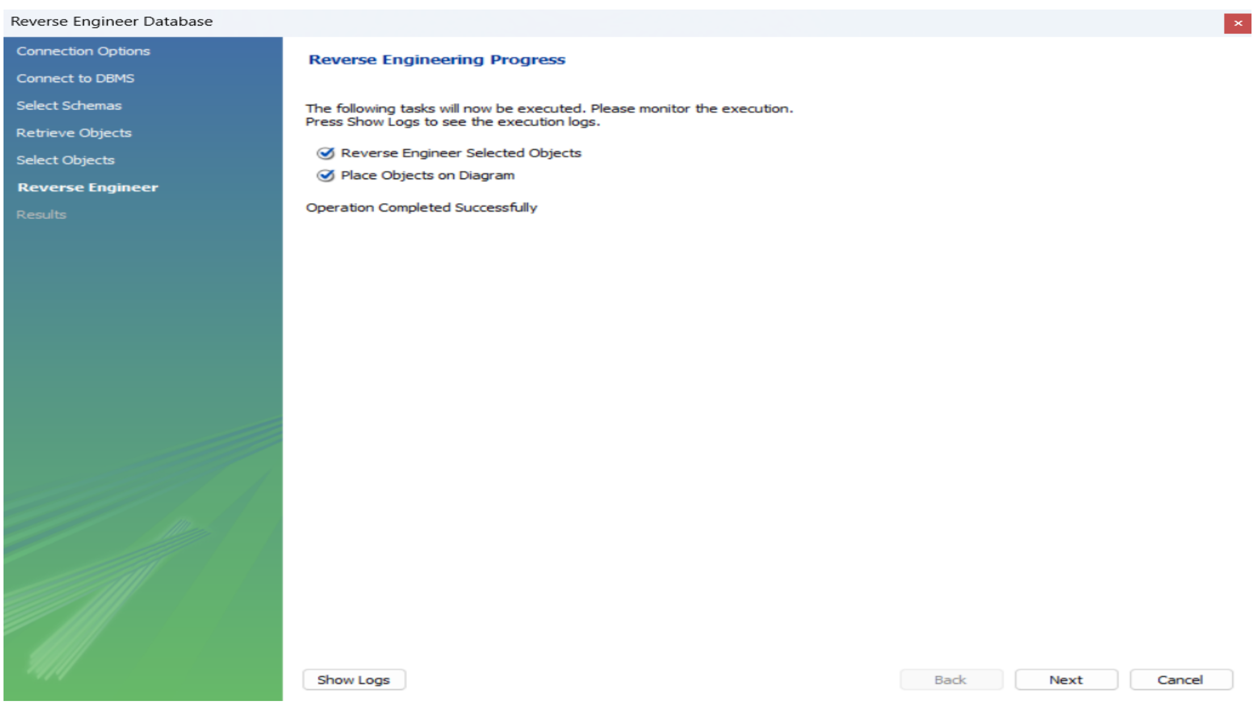

Fig 18. Reverse engineer progress completion

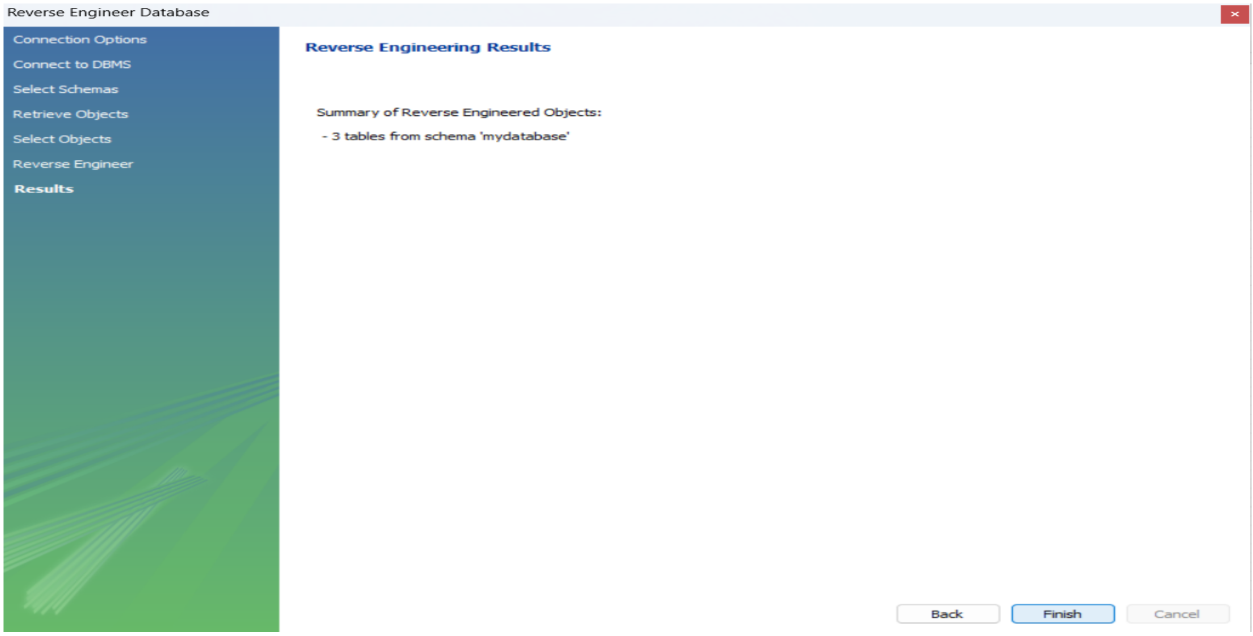

Fig. 19 the results

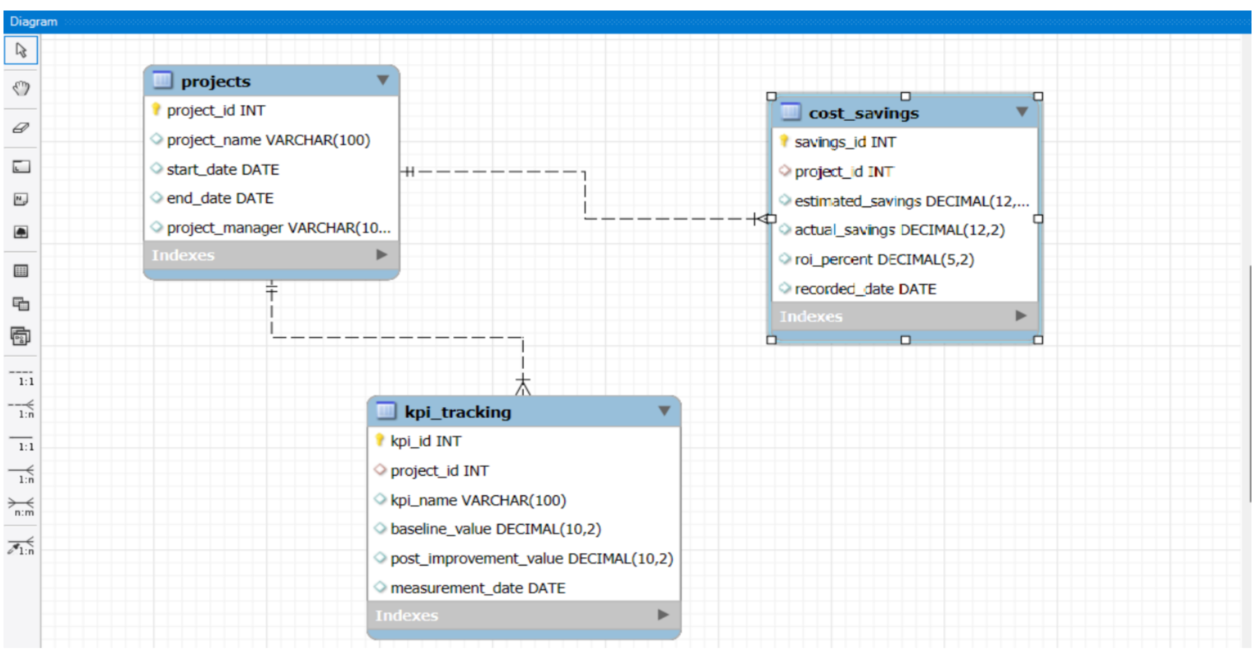

Fig. 20 EER(Enhanced Entity Relationship) Diagram

Summary and Reflection on Database Design in MySQL Workbench

The EER (Enhanced Entity Relationship) diagram shown in the image represents the logical structure of a relational database designed to analyze the economic impact of process improvements using data science and business intelligence tools. The diagram consists of three normalized tables—projects, kpi_tracking, and cost_savings—linked through foreign key relationships. Each table captures distinct and complementary dimensions of improvement initiatives, including project metadata, performance metrics (KPIs), and financial savings. This structure aligns with best practices in database design by reducing redundancy and promoting referential integrity (Elmasri & Navathe, 2017).

The projects table serves as the central entity, providing a foundational reference for both performance and cost outcomes. The kpi_tracking table enables the capture of measurable changes in key indicators such as production rate or defect rate—critical for Six Sigma and Lean analysis (George et al., 2005). The cost_savings table quantifies financial performance, including estimated and actual savings, and calculates ROI, making it suitable for post-implementation review and strategic financial planning (Eckerson, 2010).

Importance of Databases in Business Intelligence and Process Improvement

Databases are essential to business intelligence because they serve as structured repositories for storing, retrieving, and analyzing large volumes of data. Without properly normalized databases, organizations would face data silos, redundancy, and a lack of analytical coherence. MySQL Workbench, a widely adopted open-source relational database management tool, supports both forward and reverse engineering, allowing developers to transition seamlessly between conceptual design and practical implementation (Widenius et al., 2002).

In the context of this project, the use of MySQL Workbench enabled schema creation, table definition, and relationship enforcement—all crucial for developing accurate analytical models. Structured Query Language (SQL) was used to populate and manipulate data, reflecting real-world tasks expected in data-driven organizations.

Moreover, the ability to reverse engineer the schema into a visual format (EER diagram) provided a valuable tool for both documentation and communication, satisfying data governance and reproducibility expectations in professional environments (Inmon, 2005).

Academic Integration and Scholarly Support

This database implementation demonstrates thoughtful application of course concepts, especially as they relate to data modeling, schema design, and relational integrity. It also reinforces the academic principle of using data science to drive strategic improvement in post-COVID organizational recovery—where agility and evidence-based decisions are paramount (Brynjolfsson et al., 2020). The layered structure of the database supports analysis across temporal periods (before and after process changes), which is essential for causal inference and performance benchmarking in industrial engineering.

References

Brynjolfsson, E., Rock, D., & Syverson, C. (2020). *COVID-19 and remote work: An early look at US data*. National Bureau of Economic Research. https://doi.org/10.3386/w27344

Eckerson, W. (2010). *Performance dashboards: Measuring, monitoring, and managing your business*. John Wiley & Sons.

Elmasri, R., & Navathe, S. B. (2017). *Fundamentals of database systems* (7th ed.). Pearson.

George, M. L., Rowlands, D., Price, M., & Maxey, J. (2005). *The Lean Six Sigma pocket toolbook*. McGraw-Hill Education.

Inmon, W. H. (2005). *Building the data warehouse* (4th ed.). Wiley.

Widenius, M., Axmark, D., & DuBois, P. (2002). *MySQL reference manual: Documentation from the source*. O'Reilly Media.