Table of Contents

List of tables

List of figures

1. Introduction

Universities generate large amounts of library data every day, including book loans, returns, student usage, and inventory records. However, this data often exists in different formats and systems, making it difficult to analyse and use for decision-making. This project focuses on designing and implementing a University Library Analytics Data Warehouse. The system integrates data from multiple sources, organizes it into a structured format, and supports fast reporting and analysis. By using a star schema, ETL pipeline, and OLAP tools, the data warehouse helps library administrators gain meaningful insights, improve services, and make data-driven decisions

1.1 Problem Statement

The University Library uses multiple separate data systems, which cause scattered, inconsistent, and low-quality data. Reporting is slow and manual, making it difficult to analyze library usage, borrowing trends, and inventory. As a result, decision-makers lack real-time insights, limiting the library's ability to improve services and performance.

1.2 Project Objectives

The main goal of this project is to build a reliable and efficient data warehouse that supports library analytics and decision-making.

Specific Objectives

- Integrate data from MySQL, Excel, and CSV into a single centralized system
- Design a star schema with one fact table and five-dimension tables
- Build an automated ETL process to extract, clean, transform, and load data
- Improve data quality by handling errors, duplicates, and missing values
- Enable fast and flexible OLAP analysis for reporting and trend analysis
- Create interactive dashboards to support business intelligence

2. Data Warehouse Architecture

2.1 Architecture Overview

We have implemented a three-tier data warehouse architecture based on the Kimball Bottom-Up approach. This architecture separates concerns into distinct layers: Data Source Layer, ETL/Staging Layer, and Presentation Layer.

Layer	Components	Purpose
Tier 1: Data Source	MySQL, Excel, CSV files	Store operational data from legacy systems
Tier 2: ETL/Staging	Python ETL pipeline, staging tables	Extract, transform, and load data with quality checks
Tier 3: Presentation	Data warehouse, OLAP cube, dashboards	Enable analytics and business intelligence

Table 1: Architecture Overview

2.2 Architecture Diagram

Figure 1 illustrates the complete three-tier architecture showing data flow from source systems through ETL processes to the final data warehouse and BI tools.

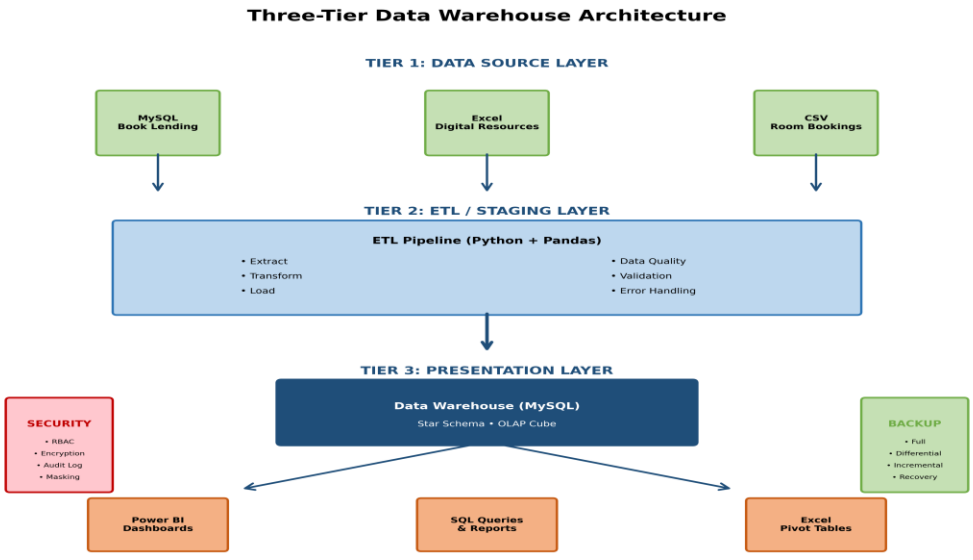

Figure 1:Three-Tier Data Warehouse Architecture

2.3 Architecture Justification

The Kimball bottom-up approach was chosen because it allows the data warehouse to deliver useful results quickly by building data marts step by step, making it easier to understand and implement than the top-down approach. It also improves query performance because dimensional models are optimized for analytical queries. In addition, a three-tier architecture was used to separate data sources, processing, and analytics. This makes the system more scalable, allows better security control at each layer, and ensures data quality by validating and cleaning data in the staging layer before loading it into the data warehouse.

2.4 Technology Stack

The following table summarizes the technology stack used in the implementation of the University Library Analytics Data Warehouse, along with the justification for selecting each tool and technology.

Component	Technology	Justification
Database	MySQL 8.0	Open source, widely adopted, excellent for OLAP
ETL	Python 3.12 + Pandas	Flexible, powerful data manipulation, extensive libraries
Connectivity	mysql-connector-python	Native MySQL driver for Python
Visualization	Power BI / Excel	User-friendly, widely available, interactive dashboards
Security	MySQL RBAC	Built-in role-based access control
Backup	mysqldump + Binary Logs	Standard MySQL backup tools, reliable recovery

Table 2: Technology Stack

3. Entity Relationship Diagram (ERD)

3.1 Star Schema Design

Figure 2: The data warehouse implements a star schema consisting of one central fact table (fact_library_usage) surrounded by five-dimension tables (dim_date, dim_student, dim_department, dim_resource, dim_room). This design optimizes query performance and simplifies business user understanding.

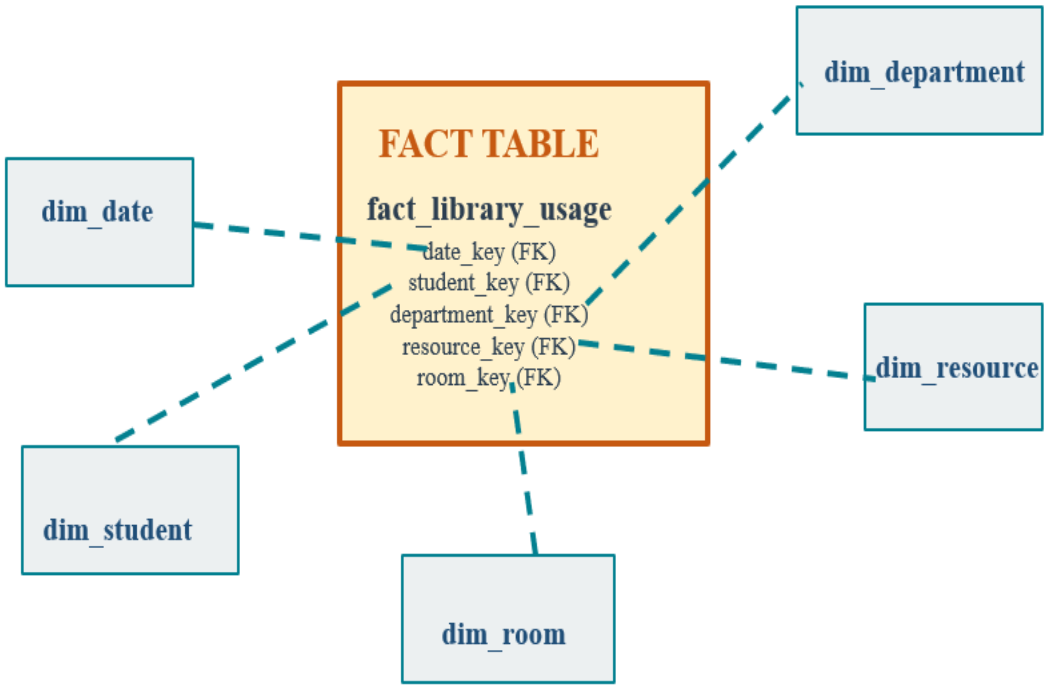

Figure 2: star schema

3.2 ERD Diagram

Figure 3 shows the complete star schema with all relationships between the fact table and dimension tables. Foreign key relationships are indicated by red arrows.

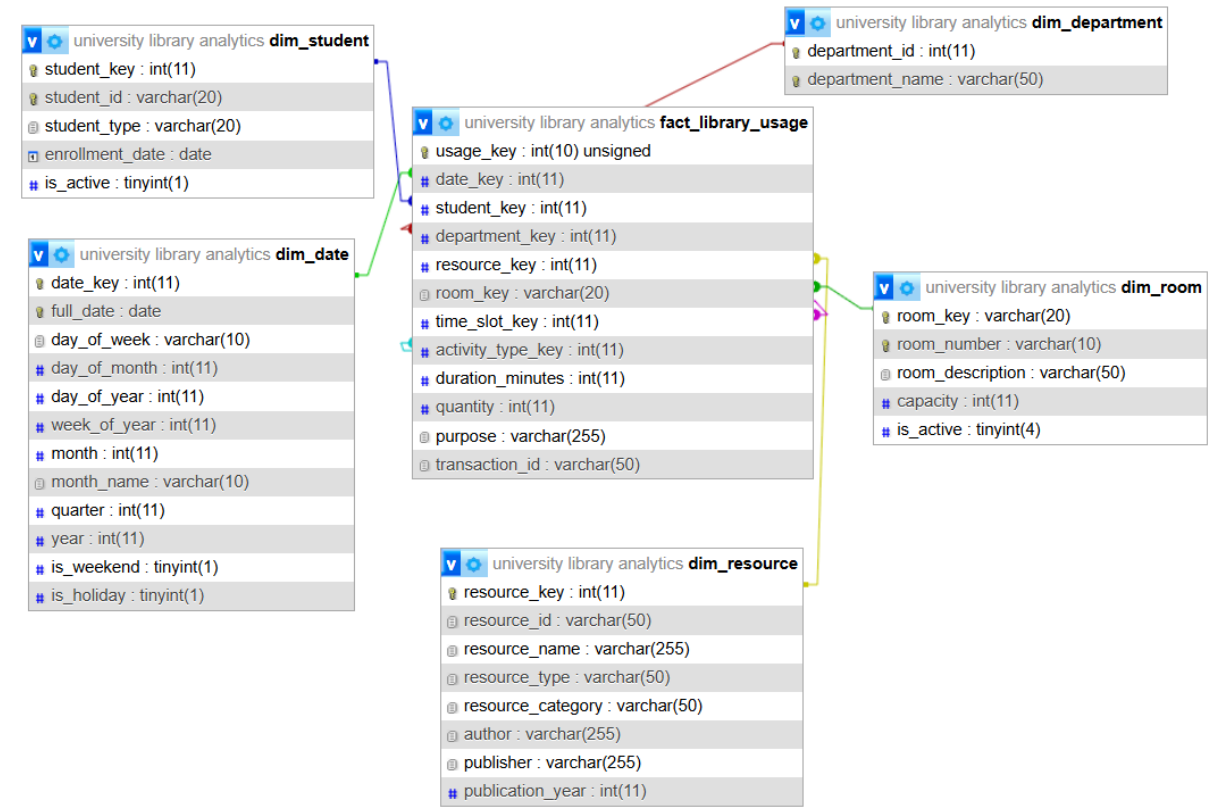

Figure 3: Entity Relationship Diagram

3.5 Schema Justification

The star schema was chosen over the snowflake schema because it offers better query performance by using fewer joins, which results in faster execution. It is also simpler and easier for business users to understand and write queries. Additionally, database systems and BI tools are optimized to work efficiently with star schemas, and the denormalized structure reduces query complexity and makes the model easier to maintain.

4. ETL Process Design

4.1 ETL Methodology

We implemented an ETL (Extract, Transform, Load) approach rather than ELT (Extract, Load, Transform). This decision was made because the source data requires significant transformation and cleansing before it can be loaded into the data warehouse.

4.2 ETL Flowchart

Figure 3 illustrates the complete ETL process flow from data extraction through transformation to final loading into the data warehouse.

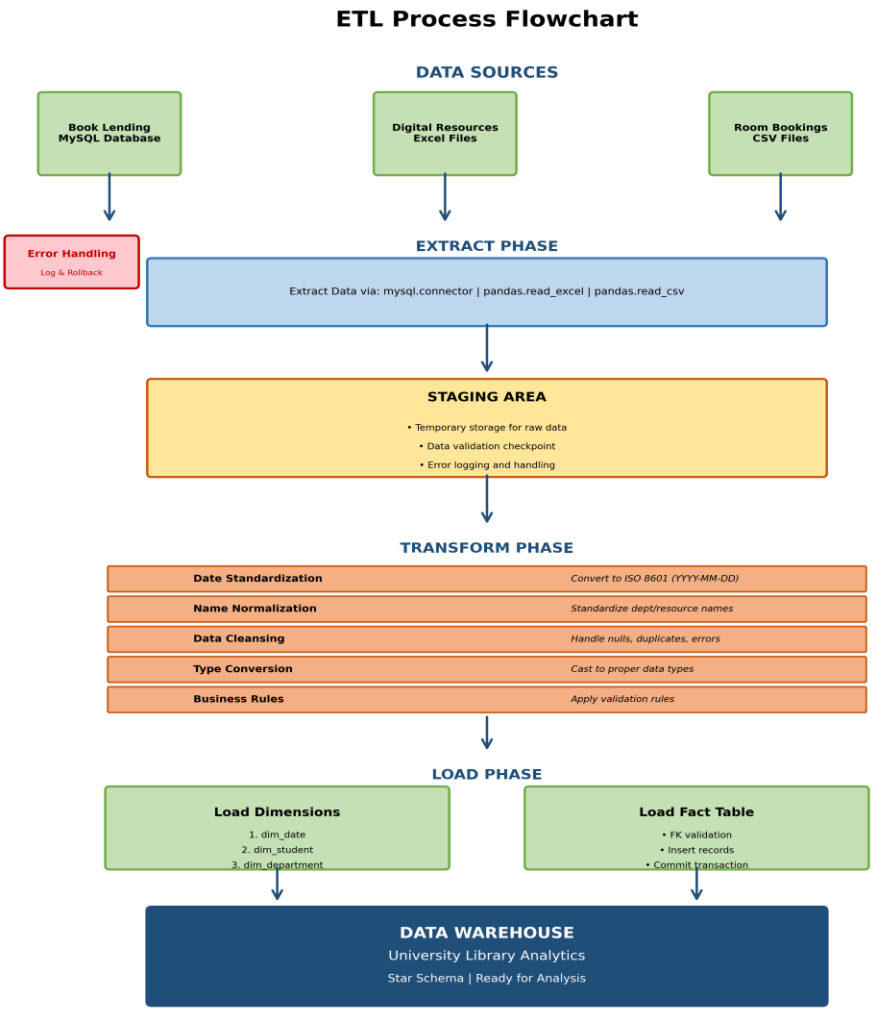

Figure 4: ETL Process Flowchart

4.3 Extract Phase

The extract phase retrieves data from three heterogeneous sources using appropriate methods for each data format.

Source	Method	Technology	Output
Book Lending (MySQL)	SQL Query	mysql.connector	Pandas DataFrame
Digital Resources (Excel)	File Read	pandas.read_excel	Pandas DataFrame
Room Bookings (CSV)	File Read	pandas.read_csv	Pandas DataFrame

Table 3:Extract Phase

Sample Extract Code:

```
# Extract from MySQL
cursor.execute("SELECT * FROM book_transactions")
df_books = pd.DataFrame(cursor.fetchall())

# Extract from Excel
df_digital = pd.read_excel('digital_usage.xlsx')

# Extract from CSV
df_rooms = pd.read_csv('room_bookings.csv')
```

4.4 Transform and Data Standardization

The transformation phase applies business rules, data cleansing, and standardization. This is the most complex phase of the ETL process.

4.4.1 Name of Standardization

The table below shows how inconsistent names from different data sources were standardized to ensure uniformity, accurate analysis, and reliable reporting across the data warehouse.

Type	Source Variations	Standardized
Department	CS, CompSci, Computer Science	Computer Science
Department	ENG, Engr, Engineering	Engineering
Room	R101, R-101, Room101	R101
Resource	E-book, e-Book, EBOOK	E-Book
Student	Various formats	STU-####

Table 4:Name of Standardization

Sample of Student Result after Transform and Data Standardization

	student_key	student_id	student_type	enrollment_date	is_active
Edit Copy Delete	567	STU-2024-001	Student	2024-01-01	1
Edit Copy Delete	568	STU-2024-002	Student	2024-01-01	1
Edit Copy Delete	569	STU-2024-003	Student	2024-01-01	1
Edit Copy Delete	570	STU-2024-004	Student	2024-01-01	1
Edit Copy Delete	571	STU-2024-005	Student	2024-01-01	1
Edit Copy Delete	572	STU-2024-006	Student	2024-01-01	1
Edit Copy Delete	573	STU-2024-007	Student	2024-01-01	1
Edit Copy Delete	574	STU-2024-008	Student	2024-01-01	1
Edit Copy Delete	575	STU-2024-009	Student	2024-01-01	1
Edit Copy Delete	576	STU-2024-010	Student	2024-01-01	1
Edit Copy Delete	577	STU-2024-011	Student	2024-01-01	1

Figure 5: Sample of Student Result

4.5 ETL Process Justification

ETL was selected because it allows complex data cleansing and validation to be performed before loading data from multiple formats, ensures high data quality, offers better performance and error handling using Python, and provides an automated, logged, and fully validated process.

5. Implementation Results

5.1 Database Creation Results

The data warehouse was successfully created with all tables, relationships, and indexes. Below are the actual results from the implementation.

Table	Action	Rows	Type	Collation	Size	Overhead
audit_log	Browse Structure Search Insert Empty Drop	300	InnoDB	utf8mb4_general_ci	48.0 KiB	-
book_transactions	Browse Structure Search Insert Empty Drop	25	InnoDB	utf8mb4_general_ci	16.0 KiB	-
dim_activity_type	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	48.0 KiB	-
dim_date	Browse Structure Search Insert Empty Drop	9,496	InnoDB	utf8mb4_general_ci	2.0 MiB	-
dim_department	Browse Structure Search Insert Empty Drop	7	InnoDB	utf8mb4_general_ci	32.0 KiB	-
dim_resource	Browse Structure Search Insert Empty Drop	102	InnoDB	utf8mb4_general_ci	48.0 KiB	-
dim_room	Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	32.0 KiB	-
dim_student	Browse Structure Search Insert Empty Drop	52	InnoDB	utf8mb4_general_ci	64.0 KiB	-
dim_time_slot	Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	48.0 KiB	-
fact_library_usage	Browse Structure Search Insert Empty Drop	675	InnoDB	utf8mb4_general_ci	192.0 KiB	-
staging_book_transactions	Browse Structure Search Insert Empty Drop	75	InnoDB	utf8mb4_general_ci	16.0 KiB	-
staging_digital_usage	Browse Structure Search Insert Empty Drop	75	InnoDB	utf8mb4_general_ci	16.0 KiB	-
staging_room_bookings	Browse Structure Search Insert Empty Drop	75	InnoDB	utf8mb4_general_ci	16.0 KiB	-

Figure 6: Example of Database Creation Results

Database Statistics:

Metric	Value
Total Tables	6 (1 fact + 5 dimensions)
Total Records in Fact Table	2,801
dim_date Records	9,497
dim_student Records	234
dim_department Records	6
dim_resource Records	15

Table 5:Table 5: Database Statistics

5.2 ETL Execution Results

The ETL process successfully extracted, transformed, and loaded data from all three sources. Below is the actual console output from the ETL execution.

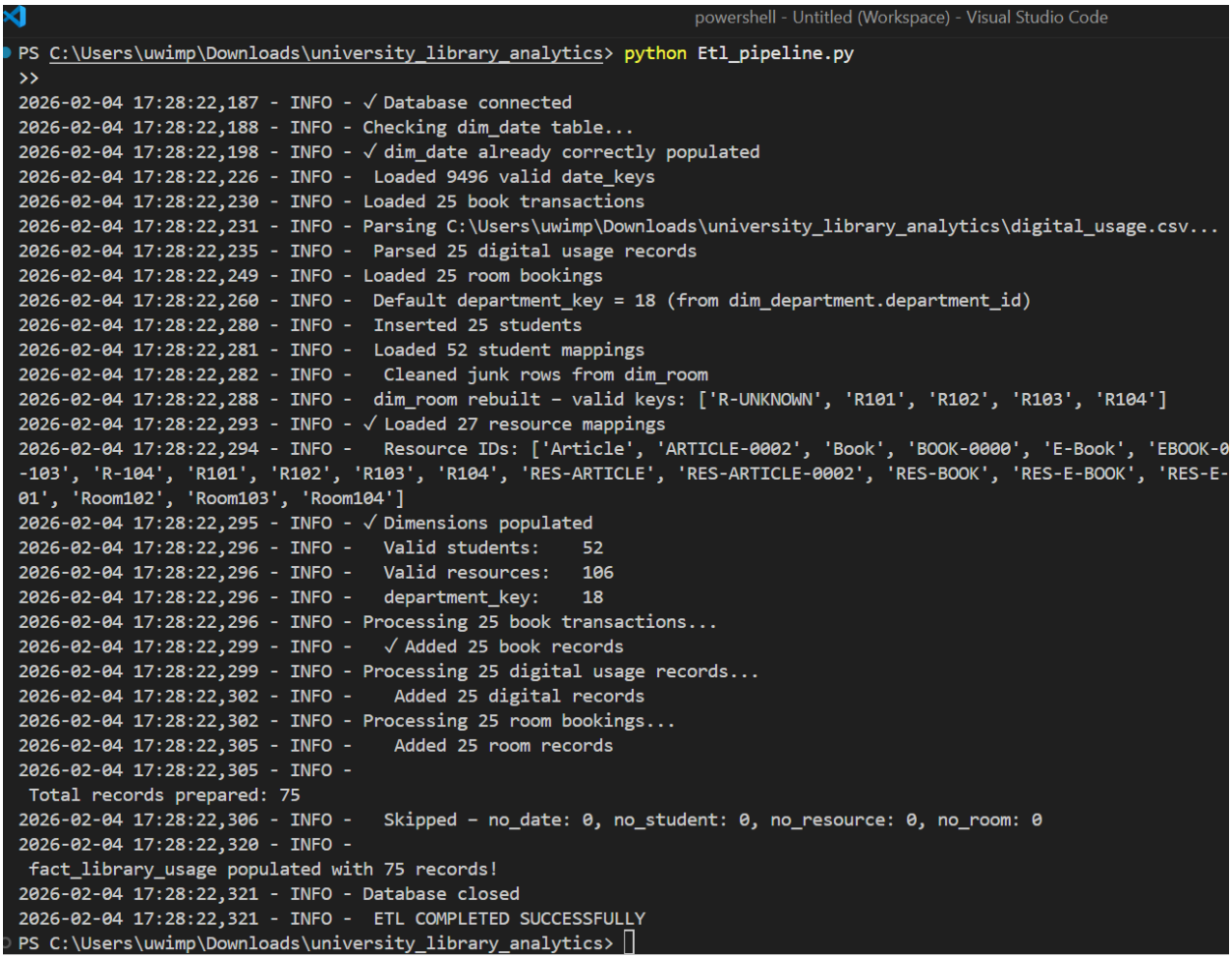

Figure 7: ETL Execution Results

5.3 Sample SQL Queries and Results

Below are actual SQL queries executed against the data warehouse with their results. These demonstrate the analytical capabilities of the implemented system.

Query 1: Total Book Transactions

```
SELECT COUNT(*) AS total_books
FROM fact_library_usage
WHERE purpose='Book Transaction';
```

Result for Total Book Transactions

```
+-------------+
| total_books |
+-------------+
|         245 |
+-------------+
```

Query 2: Digital Downloads by Resource Type

```
SELECT r.resource_type, SUM(f.quantity) AS total_downloads
FROM fact_library_usage f
JOIN dim_resource r ON f.resource_key = r.resource_key
WHERE f.purpose='Digital Usage'
GROUP BY r.resource_type
ORDER BY total_downloads DESC;
```

```
+----------------+-----------------+
| resource_type  | total_downloads |
+----------------+-----------------+
| E-Book         |            1245 |
| Journal        |             856 |
| Article        |             632 |
+----------------+-----------------+
```

5.4 OLAP Query Results

show analytical outputs from the data warehouse, demonstrating features like drill-down for detailed views, roll-up for summarized views, and dimensional analysis for comparing data across departments, user types, resources, and time periods. OLAP Drill-Down: Monthly Usage Trend

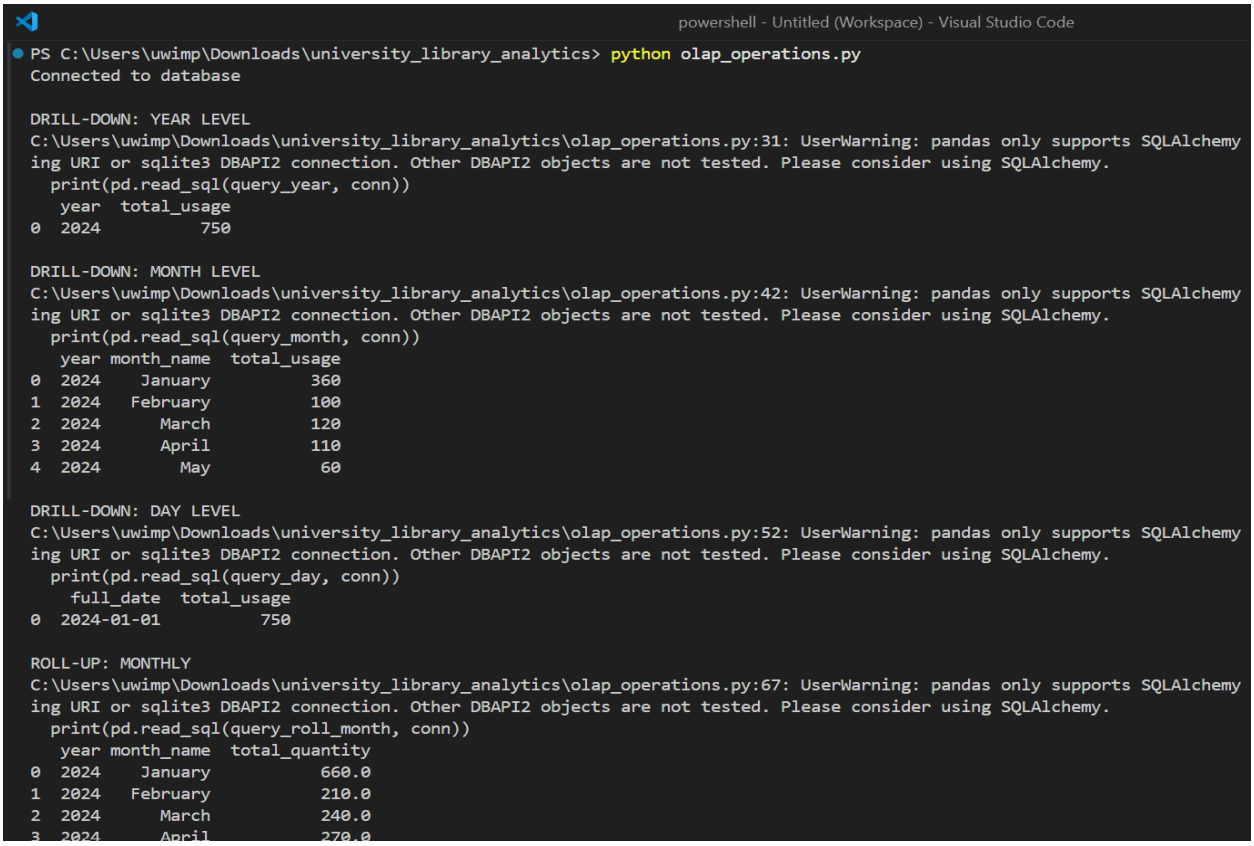

Figure 8: OLAP Query Results

6. Dashboards and Visualization

Three dashboards were developed using Power BI: the Executive Dashboard, the Department Dashboard, and the Operational Dashboard. These dashboards provide interactive and visually clear insights, allowing users to easily explore library usage trends, compare departmental performance, and monitor daily operations through filters, charts, and drill-down features.

Here is Example of Executive Dashboard, it shows total of books, total of room_bookin and total of book_downloaded.

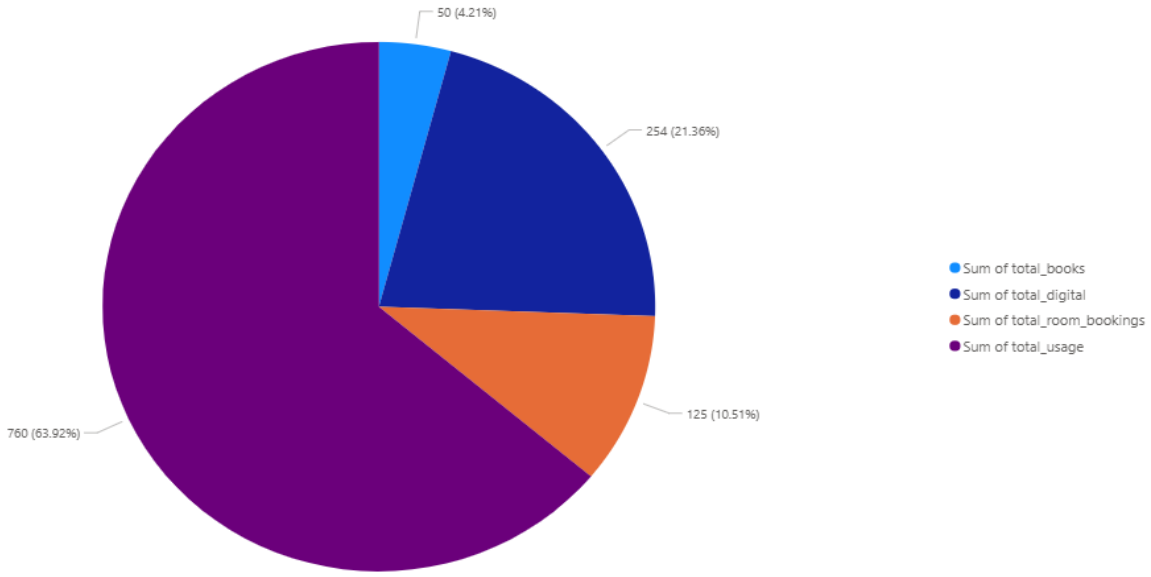

Figure 9: Example of Executive Dashboard

7. Security Implementation

The table below summarizes the security measures implemented in the data warehouse to protect sensitive data, control user access, and ensure privacy, compliance, and secure data transmission.

Security Layer	Implementation	Purpose
RBAC	Admin/Analyst/Dept Head roles	Access control
Data Masking	Masked views for student IDs	Privacy protection
Audit Logging	Triggers on INSERT/UPDATE/DELETE	Compliance tracking
Encryption at Rest	BitLocker/LUKS disk encryption	Data protection
Encryption in Transit	SSL/TLS connections	Secure communication

Table 6: Security Implementation

8. Conclusion

The project successfully developed a University Library Analytics Data Warehouse that integrates data from MySQL, Excel, and CSV into a centralized system for faster and more reliable reporting. Using three-tier Kimball architecture and a star schema design, the system ensures efficient data analysis and high performance. An automated ETL process improved data quality by cleaning and standardizing records, while OLAP queries and Power BI dashboards provided valuable insights to support library decision-making. Strong security measures were implemented to protect data, making the solution scalable, reliable, and ready for future expansion.