MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 1960-1970

Design and Implementation of a Data Warehouse for Multi-Source Data Integration in the Open University Environment

M. Musyahid Abror
Universitas Esa Unggul, Indonesia
Email: musyahid2@student.esaunggul.ac.id

ABSTRACT

The The utilization of information technology at Universitas Terbuka to support distance learning generates a large and continuously growing volume of data. This data is spread across various units, each producing data in different formats and structures. The diversity of data sources and formats resulting from operational activities creates challenges in the data integration process, making it difficult to produce centralized data. To address this issue, a data warehouse is implemented to integrate data from various sources through the ETL process. This process includes data extraction from various sources, transformation to fit the predetermined schema, and loading data into a centralized storage system. With the implementation of the Data Warehouse, Universitas Terbuka can overcome data fragmentation issues and provide a platform that enables centralized data-driven analysis. Additionally, the Data Warehouse enhances data accessibility and integrity, making it easier for users to access the necessary information for decision-making and ensuring that the data used maintains its integrity. This implementation is expected to significantly contribute to improving operational efficiency, data accuracy, and the development of data-driven planning and policies to support distance learning at Universitas Terbuka.

KEYWORDS: data integration, data warehouse, database technology, ETL, data mart.

INTRODUCTION

The rapid growth of data in educational institutions, particularly at Universitas Terbuka, has created increasing demands for more effective data management. As a provider of distance learning, Universitas Terbuka generates large volumes of data from various units, including administration, academic affairs, learning activities, and human resources. The need to manage these heterogeneous data sources in an integrated manner highlights the urgency of implementing innovative data management solutions.

Each unit at both the central office and regional offices maintains its own datasets and independent dashboards that are used for decision-making. This fragmented data environment presents significant challenges in terms of coordination and monitoring data integrity. The existence of isolated systems increases the risk of data inconsistencies, inaccuracies, and difficulties in tracking relationships across organizational units.

The inability to access data in an integrated manner can hinder Universitas Terbuka's capacity to make timely and well-informed strategic decisions. Therefore, to improve the effectiveness of data management, concrete measures such as the implementation of a Data Warehouse (DW) are required. A Data Warehouse serves as an integrated data repository designed to support analytical processing [8], enabling Universitas Terbuka to consolidate data from multiple units into a single centralized data source. This approach allows stakeholders at various organizational levels to access consistent and reliable information, serving as an initial framework for integrating data from multiple databases and other information sources [9].

The implementation of a Data Warehouse provides a unified platform that integrates administrative, academic, learning, and human resources data. This integration facilitates cross-unit analysis, trend identification, and strategic decision-making based on accurate and up-to-date information.

In general, a data warehouse constitutes a fundamental component of decision support systems in both corporate and public administration contexts. Data stored in a data warehouse are typically analyzed using Online Analytical Processing (OLAP) tools, which offer advanced capabilities for data aggregation and comparison. Decision support applications are highly dependent on the reliability and accuracy of the underlying data. Moreover, a data warehouse does not only store current snapshot data but also maintains historical data, enabling longitudinal analysis over extended periods [4].

In addition, the centralized nature of a data warehouse provides significant advantages in monitoring data integrity. Centralized storage enables more efficient data quality management, including error detection, data correction, and the enforcement of data security policies.

These measures are expected to offer an innovative solution to address the growing demands of data management amid the continuously increasing data volume at Universitas Terbuka. Through the implementation of a Data Warehouse, Universitas Terbuka can enhance coordination, improve data integrity, and establish a strong foundation for informed and systematic decision-making.

To achieve these objectives, this study is grounded in a comprehensive understanding of the Data Warehouse life cycle, emphasizing the formalization of each design phase and the identification of interactions among these phases [11].

RESEARCH METHODOLOGY

This study employed the following research methods:

1. Literature Review

The literature review was conducted to collect and examine relevant scholarly sources related to the research problem and objectives. This method provided theoretical foundations and insights into data warehouse concepts, data integration, and system development methodologies.

2. Field Study

Data were collected through interviews, which involved direct interaction between the researcher and respondents. In this study, interviews were conducted with key stakeholders, including unit heads and database administrators from each organizational unit, to gather detailed information regarding data requirements and existing data management practices.

3. Research Site and Duration

The research was conducted at Universitas Terbuka. The data collection and system development activities began in November 2023.

4. Development Methodology

The system development methodology was divided into two interrelated components:

1) the data warehouse development methodology, which focuses on the backend system, and
2) the dashboard application development methodology, which focuses on the frontend system.

5. Data Warehouse Development Method

This study adopted a structured set of steps and methodologies to ensure the successful development of the data warehouse. The development process followed the Data Warehouse Development Methodologies approach [2], which emphasizes the creation of an integrated and centralized data source. The stages of this methodology are illustrated in Figure 1.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 1960-1970

Figure 1. Data Warehouse Development Methodologies

Figure 1 illustrates the initial stage of the research process, which begins with requirements identification. This stage aims to gather comprehensive information related to the needs and objectives of data warehouse development. Based on the requirements identification process, several key preparatory activities were defined as follows:

1) Collecting relevant information to understand user requirements, particularly regarding the expected data outputs and analytical needs.
2) Analyzing data structures from multiple data sources to understand storage formats and to determine an appropriate database schema.
3) Defining the development approach and selecting suitable technologies to be used in the data warehouse implementation.
4) Designing user interfaces to facilitate efficient and user-friendly access to data stored in the data warehouse.
5) Establishing parameters and criteria for data testing to ensure that the stored and managed data meet acceptable quality standards.
6) Identifying the needs of users and key stakeholders in order to define evaluation criteria for assessing the success of the data warehouse development.

6. Dashboard Application Development Method

The development of the dashboard application employed the Waterfall model. This model was selected because the project scope was clearly defined, and the data sources from the data warehouse were already available. As a result, the development process could focus entirely on interface design and data visualization.

The Waterfall model, often referred to as the classic life cycle, represents a systematic and sequential approach to software development. The process begins with the specification of user requirements and proceeds through successive phases, including planning, modeling, construction, and system delivery to users. The development cycle concludes with support for the completed software product (Pressman, 2012). The stages of the Waterfall model are illustrated in the following figure.

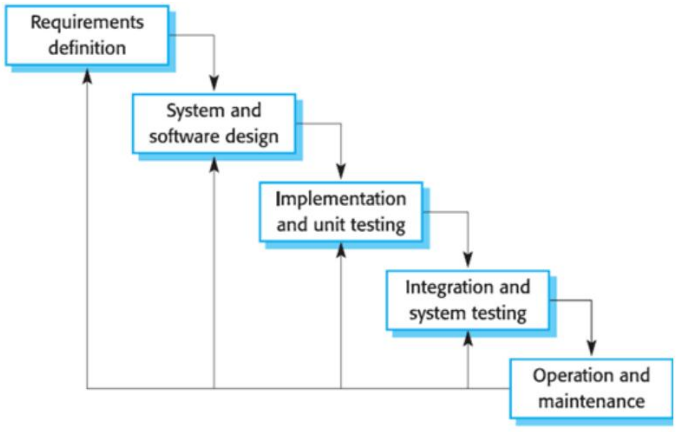


Figure 2. Waterfall Model

In its development, the Waterfall model consists of several sequential stages, namely: Requirements Definition, System and Software Design, Implementation and Unit Testing, Integration and System Testing, and Operation and Maintenance. Each stage must be completed before proceeding to the next. The stages of the Waterfall model applied in this study are described as follows:

1. Requirements Definition

At this stage, effective communication between the system developers and users is essential to understand the expected software functionalities and system constraints. In this study, a survey was conducted by distributing questionnaires to identify the types of information required by users prior to software development.

2. System and Software Design

During this phase, the requirements specifications obtained in the previous stage are analyzed, and the system design is prepared. This stage includes business process modeling, database design, and user interface design. The outputs of this phase consist of BPMN (Business Process Model and Notation), UML (Unified Modeling Language), Use Case diagrams, ERD (Entity-Relationship Diagram), and interface mockups.

3. Implementation and Unit Testing

In this stage, system coding is carried out based on the design and analysis results from the previous phase. The application was developed using the Laravel framework (version 11.44.2) and implemented with HTML, CSS, JavaScript, and PHP. PostgreSQL was used as the database management system. Unit testing was conducted to ensure that each component functions as intended.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 1960-1970

4. Integration and System Testing

All units developed during the implementation phase were integrated into a complete system after individual unit testing. Following integration, comprehensive system testing was conducted to identify any failures or errors. This testing process employed black-box testing techniques.

5. Operation and Maintenance

The operation and maintenance phase was not conducted in this study, as the application development process was completed up to the Integration and System Testing stage and did not extend to system deployment and long-term maintenance.

RESULTS AND DISCUSSION

Data Source Description

The data warehouse implementation at Universitas Terbuka utilizes four primary data sources. Each data source is represented in the data warehouse through a set of dimension tables and fact tables. These data sources were integrated to support comprehensive analytical processing. The data sources incorporated into the data warehouse are described as follows:

Table 1 Data Source Description

NO	Data Source	Format	Brightness
1	*Student Database*	SQL	Storing all student data from the beginning of registration to the end of the lecture
2	*Staffing Database*	SQL	Storing all employee data at the Open University
3	Learning Data	JSON	Keeping a record of students' learning history
4	Payment Data	JSON dan CSV	Store student payment data

Amount of Data by Data Source

The following is the amount of data based on the last 4 years

Table 2 Amount of data by data source

NO	Data Source	Year				
		2020	2021	2022	2023	2024
1	*Student Database*	853.446	877.152	1.203.801	1.533.372	1.533.372
2	*Staffing Database*	2.536	2.533	2.530	2.526	2.526
3	*Learning Rest* API	499.154	964.897	881.331	800.376	776.397
4	Payment Data	881.191	920.333	1.144.179	1.441.266	1.769.840

Information Requirements Analysis

The identification of data and information requirements for the data warehouse at Universitas Terbuka is a critical step, as it directly affects the quality of reports delivered to stakeholders. These reports must be accurate, reliable, and easy to interpret in order to support effective decision-making processes.

Based on an analysis of business processes and existing information systems at Universitas Terbuka, the required data and information are categorized as follows:

1. Student Database

Reports include information on prospective students who have completed the admission process, both those who have paid the admission fee and those who did not proceed further. In addition, the number of students registering for courses can be analyzed by period and disaggregated by Regional UT Office, Faculty, and Study Program.

2. Human Resources Database

Reports provide information on employee status and employee distribution based on active status, highest educational attainment, age, and employee type. These data can be analyzed and queried by year.

3. Learning Data

Reports include comprehensive information related to students, study programs, tutors, courses, tutorial locations, schedules, learning periods, start and end dates, and tutorial status. This information provides a complete overview of learning activities and instructional implementation.

4. Payment Data

Reports include information on total payments based on payment partners, payment methods used by students, and payment trends analyzed according to students' geographic locations.

Discussion of Research Results

1. Data Warehouse Architecture

The system and infrastructure architecture adopts a distributed approach, incorporating multiple components to support the implementation of the data warehouse. The ETL (Extract, Transform, Load) process begins by extracting data from various external sources, including SQL databases, REST APIs, and CSV files used by different units at Universitas Terbuka. The extracted data are then stored in a staging area, which serves as a temporary repository for filtering, cleansing, and preparing data before it is loaded into the main data warehouse.The use of a staging area ensures that the data loaded into the data warehouse are well-structured and consistent. Moreover, the staging area provides flexibility for data manipulation and transformation according to analytical requirements, while also reducing the processing burden on OLTP systems that support ongoing operational activities.This process includes data cleansing and transformation to ensure data consistency and integrity prior to loading into the data warehouse. Once the transformation process is completed, the data are loaded into the data warehouse, which functions as a centralized data repository. The data are then segmented into multiple data marts based on different functional units. This segmentation enables each data mart to provide focused access for users who require specialized analyses relevant to their respective units.The system infrastructure utilizes high-capacity servers equipped with 64-core processors and 128 GB of RAM to support large-scale data processing. PostgreSQL is used as the Database Management System (DBMS), providing robust support for large-scale data storage and analytical processing. In addition, the system is equipped with monitoring and security mechanisms to ensure data confidentiality and protection. Data encryption is implemented, and access is restricted to authorized users only.

Through this integrated architecture, the system is capable of handling large data volumes, facilitating data-driven decision-making, and improving operational efficiency at Universitas Terbuka.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 1960-1970

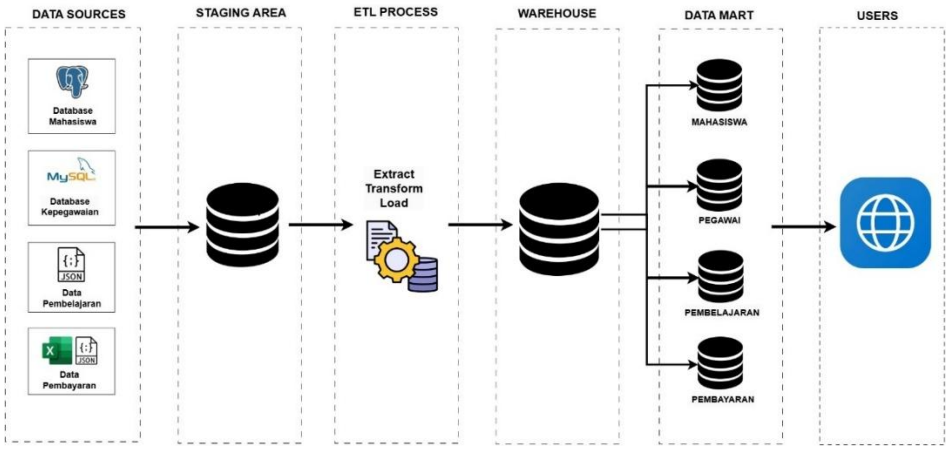


Figure 3 Data Warehouse Architecture

To effectively manage and utilize data, several key components operate in an integrated manner. Each component plays a specific role within the data warehouse architecture. The components are described as follows:

1) Operational Data

Operational data refer to data generated from daily system operations. These data are stored in transactional databases that support routine operational activities and business processes.

2) Staging Area

The staging area serves as a temporary storage layer used to prepare data before it is loaded into the data warehouse. Within this area, the ETL (Extract, Transform, Load) process is executed to extract data from operational data sources, transform it into a consistent and structured format, and load it into the data warehouse.

The decision to utilize a staging area prior to loading data from multiple sources into the data warehouse is driven by several considerations. Data originating from different systems often vary in format and structure, necessitating an intermediate layer for data integration and standardization to ensure consistency.

Data transformation processes including data cleansing, normalization, and the consolidation of data from multiple sources are performed within the staging area. Furthermore, executing ETL processes in the staging area ensures that only validated and high-quality data are loaded into the data warehouse. The staging area also supports batch processing, allowing data preparation activities to be conducted without disrupting core operational systems that may be concurrently serving users. This approach ensures that data are thoroughly processed and properly managed before final integration into the data warehouse.

3) Data Mart

Data marts are developed based on the specific analytical requirements of individual organizational units. Each data mart is tailored to particular analytical purposes, such as student data analysis, human resources, research, learning activities, and payment transactions. By focusing on domain-specific datasets, data marts provide relevant and targeted information to specific teams or functional units. Their structured and organized design enhances data accessibility and enables users to efficiently retrieve and differentiate information. The data marts implemented within the data warehouse are presented as follows:

Table 3 Data Mart Grouping

Data Mart	Purpose
Students	Provide data related to admission data, course registration data, recapitulation of admission and registration data based on UT Regions, faculties and study programs
Employees	Provide data related to social security such as active employees, employee placement, employment status, education level and data of staff and lecturers
Learning	Provide tutor data, tutorial locations and types of learning taken by students
Finance	Provide data related to the Open University's financial receipts of various types of payments

2. Data Warehouse Design

In this design, the nine-step methodology quoted from Connolly and Begg (2005, pp. 1187-1193) is used, as well as the methodology proposed by Kimball in the design of the data warehouse. These measures include:

1) Choosing the process

Choosing the right process or function is essential to building an effective data warehouse because it must be appropriate to the needs of the organization and be able to solve many important problems in the overall management and processing of data. For example, the process that is the main focus in the construction of this data warehouse is academic and non-academic data management, which includes various important aspects of education and administration.

In this context, the processes that will be analyzed in detail include student data management, staffing, finance, learning, and payments. This data warehouse is created by combining various relevant data sources to provide a more complete and comprehensive picture, enable more in-depth data analysis, and support more informed and informed decision-making. All of this is intended to improve operational efficiency and support the achievement of the organization's strategic objectives.

2) Choosing the grain

Grain is the level of detail obtained from the potential facts to be analyzed. Choosing a grain means determining exactly what each entry in the fact table indicates so that we can understand the context and relevance of the data collected. The selection of the right grain is very important because it determines how detailed the information will be presented in the data warehouse and how it can be used for further analysis. By clearly defining grains, the stored data will be easier to analyze according to the needs of each unit and allow for more effective integration between various data sources at the Open University.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 1960-1970

The grains from the Open University used to design the data warehouse are as follows:

1. Student Data

Analysis of student data includes the number of admissions, the number of paid admissions, the number of new student registrations, the number of old student registrations, the data graph of UT Regions, faculties and study programs with the highest number of admissions. data graph of UT Regions, faculties and study programs with the most registered students. The analysis will be carried out per specific period of time, such as days, months, academic periods, or years.

2. Staffing Data

Analysis of employee data includes the number of active employees, the number of central employees and regional employees, recapitulation of employee activity status, active employee status, employee data based on education level. The analysis will be carried out per certain period of time (months, years).

3. Learning Data

Analysis of learning data related to tutor data, tutorial location, face-to-face tutorials, webinar tutorials and the number of tutorials based on the status of tutorials that can be analyzed per specific time period (months, years)

4. Payment Data

Analysis of payment data based on transactions from each payment partner channel, payment mapping by payment type, location that can be analyzed per specific time period (months, years).

5. Identify and Conform the Dimensions

6. Student Registration Data

In the following table, the relationship between dimensions and grains of registration facts is shown in the form of a matrix for further analysis.

Table 4 Grain Student Registration Data from Student Data Facts

Dimensions	*Grain*					
	Admission Based on UT Region	Admissions by Faculty	Admission by Study Program	Registration Based on UT Region	Registration by Faculty	Registration by Study Program
Personal Data				x	x	x
Academic Period	x	x	x	x	x	x
Prospective Students	x	x	x			
UT Region	x			x		
Study Programs		x	x		x	x

Student Payment Data

The following table shows the relationship between dimensions and grains of payment facts in the form of a matrix:

Table 5 Grain Payment Data from Payment Data Facts

Dimensions	*Grain*		
	Payments by Payment Type	Payment based on UT Region	Payments by transaction partner
Personal Data		x	
Academic Period			
Payment Types	x		
UT Region		x	
Transaction Partners			x

Staffing Data

In the following table, the relationship between dimensions and grains of employee facts is shown in the form of a matrix:

Table 6 Grains of Staffing Data from Staffing Data Facts

Dimensions		*Grain*		
	Active Employees	Central and District Officers	Employee Activity Status	Education Level
Employee Status	x			
Activity Status			x	
Employee Location		x		
Education Level				x

Learning Data

In the following table, the relationship between dimensions and grains of learning facts in the form of a matrix is shown:

Table 7 Grain Learning Data from Learning Facts

Dimensions		*Grain*		
	Active Tutor	Tutorial Location	Face-to-face tutorials	Tutorial Webinar
Tutor	x			
Location		x		
Status Tutorial			x	x

Choosing the facts

To determine the facts to be used in a data mart, each fact must have data that can be calculated. This data will later be displayed in the form of reports, graphs, or various types of diagrams. Here are the facts that will be displayed in the data warehouse:

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 1960-1970

Table 8 Selection of Fact Table

Business Process	*Measure*	Table Facts
Student Enrollment Analysis	Credits Taken, Tuition Fees, Number of Courses Taken, Semester	Student Registration Facts
Employee Analysis	TMT CPNS, TMT PNS, TMT UT, TMT START, Year of Graduation	Employee Facts
Learning Analysis	Tutorial date, Tutorial status	Learning Facts

Storing Pre-Calculations in Fact Tables

Pre-calculated measures are stored in fact tables to improve query performance and support efficient analytical processing. The pre-calculations implemented in this study are categorized based on data domains as follows.

1. Student Data

The following pre-calculations are stored in the student registration fact table:

1) Number of Students by Gender
   Calculated based on the total number of registered students grouped by gender as recorded in the registration data.
2) Number of Students by Academic Status
   Calculated based on student academic status, such as active, graduated, on leave, or other statuses.
3) Number of Students by Study Motivation
   Calculated based on students' stated purposes for enrollment, such as employment preparation, personal development, or other objectives.
4) Number of Students by Study Program
   Calculated based on the total number of students enrolled in each study program at Universitas Terbuka.
5) Number of Students by Registration Information Source
   Calculated based on the source of registration information, such as advertisements, peer recommendations, or other channels.
6) Number of Students by City/Regency
7) Calculated based on the city or regency of origin of registered students.
8) Number of Students by Regional UT Office
   Calculated based on the Regional UT Office where students registered or attended academic activities.

Number of Courses Registered by Students

Calculated based on the total number of courses registered by students within a specific academic period.

Total Payments by Payment Partner per Regional UT Office

Calculated based on the total payment amounts processed by each payment partner and grouped by Regional UT Office.

Human Resources Data

The following pre-calculations are stored in the employee fact table:

1) Number of Employees by Gender
2) Calculated based on employee gender information recorded in the human resources database.
3) Number of Employees by Employment Status
4) Calculated based on employment status, such as permanent or contract employees.
5) Number of Employees by Employee Type
   Calculated based on employee categories, such as administrative staff, technical staff, or other classifications.

Number of Employees by Educational Level

Calculated based on the highest educational qualification attained (e.g., bachelor's, master's, or diploma).

1. Number of Employees by Work Location
2. Calculated based on employee work locations, such as headquarters or regional offices.
3. Number of Employees by Activity Status
4. Calculated based on activity status, including active, on leave, or inactive employees.
5. Number of Employees by Organizational Unit
6. Calculated based on organizational units, such as finance, marketing, faculties, or human resources.
7. Number of Employees by Competency Type
8. Calculated based on competency classifications, including managerial, technical, and soft skills.

Learning Data

The following pre-calculations are stored in the learning fact table:

1) Number of Active Tutors
2) Calculated based on tutors recorded as active within a specific time period.
3) Number of Tutorial Locations
   Calculated based on the total number of tutorial locations, including face-to-face and webinar-based tutorials.

Number of Face-to-Face Tutorials

1. Calculated based on the number of in-person tutorials conducted during a given period.
2. Number of Webinar Tutorials
3. Calculated based on the number of tutorials conducted via webinars within a specific period.
4. Number of Tutorials by Status per Academic Term
5. Calculated based on tutorial status (e.g., active, completed, canceled) for each academic term.

Payment Data

The following pre-calculations are stored in the payment fact table:

1. Number and Total Amount of Transactions by Payment Partner
2. Calculated based on transaction counts and total payment amounts processed by each payment partner within a given period.
3. Payments by Year and Academic Term
   Calculated based on total payments and transaction volumes grouped by year and academic term.

Payments by Payment Type

Calculated based on payment categories, such as tuition fees, administrative fees, and other payment types.

Payments by Payment Method

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 1960-1970

Calculated based on payment methods used, including bank transfers, credit cards, and other methods.

Transactions by Payment Partner per Regional UT Office

Calculated based on transaction counts and payment totals processed by payment partners within each Regional UT Office.

Completing Dimension Tables (Rounding Out the Dimension Tables)

To ensure clarity and usability, each dimension table is supplemented with descriptive attributes that explain the contextual information represented by the dimension. These descriptions enable users to better interpret analytical results derived from the data warehouse. The following section describes each dimension table in detail.

1) Student Data

Table 9 Description of Student Dimension Table

Dimensions	*Field*	Description
Time	- Year - Moon - Day	Data reports are viewed by year, month, and day
UT Region	- Code - Name	Reports can be viewed based on UT Regions
Academic Status	- Code - Status	Reports can be viewed by code and academic status
Province	- Code - Name	Reports can be viewed by Provincial Region
Study Programs	- Code - Name - Levels - Faculty	Reports can be viewed by study program, level and faculty
Jobs	- Remarks	Reports can be viewed by job
Transaction Partners	- Name	Reports can be viewed by student payment transaction partners
Courses	- Code - Name - Credits	Reports can be viewed by course
Time	- Time - Academic year	Reports can be viewed based on time and academic year
Kabupaten Kota	- Code - Name	Reports can be viewed by student district or city
Payment Types	- Code - Remarks	Reports can be viewed by type of student payment

2) Choosing the duration of the database

The duration of the Open University program that is entered into the data warehouse is as follows:

Table 10 Selection of Duration from the Database

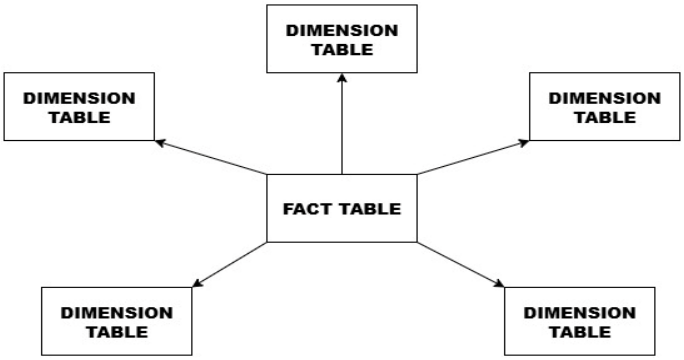
Data Name	Data Source	*The database* has been around since	Time Data Enters Data to Data *Warehouse*	Data Length in Data *Warehouse*
Student Data	*Student Record System* (SRS)	2003	2019 - 2025	6 years
Employee Data	Personnel Information System (SIMPEG)	2003	2019-2005	6 Years
Research Data	-	-	-	-
Learning Data	Tutorial Apps	2018	2021 - 2025	4 Years
Payment Data	Financial Applications and *Web Banking Services*	2020	2020 - 2025	5 Years

3) Tracking slowly changing dimensions

The Slowly Changing Dimensions (SCD) Type 2 method is used in the design of the Open University data warehouse to handle dimensional changes. This change led to the addition of new dimension data, which allowed the old data to remain available. This ensures that dimensional changes, such as changes in student data or changes in employee data, can be tracked by adding new data while keeping the old data. Therefore, the data history is maintained and analysis over time can be performed.

Star Schema and Metadata

In this design, a star scheme is chosen as the form of the scheme used. This choice is based on ease of understanding and use by users compared to other schemes. Star schemas have a less complicated structure, making it easy to write queries. This advantage makes star schemes more efficient and practical to implement in the designed system. A star schema is useful for illustrating the relationship between a dimension table and a fact table.



Gambar 4 Design Star Schema

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 1960-1970

Relationship of Registration Facts Table and Dimensions

This table outlines how each foreign key in the table fact_registrasi_mahasiswa connected to the dimension table to give context to the data.

Table 11 Relationship Table of Registration Facts and Dimensions

Table Facts (*Foreign Key*)	Connect to the Dimension Table (*Primary Key*)	Context of the Data Provided (Sample Analysis Questions)
id_mahasiswa	dim_mahasiswa (id)	Provide all the details about the students who registered. Example: • "How many students from the Informatics Engineering study program have registered?" • "How many students from UT Jakarta have registered? "
id_masa_akademik	dim_masa_akademik (id)	Describe the academic period (academic year and semester) of registration. Example: "Show the trend in the number of credits students took in odd semesters over the last 3 years."
id_dim_waktu_*source_created*	dim_waktu (id)	Provide details of the time (date, month, day) when the registration data was created. Example: "On what day of the week does the most registration activity occur?"
id_dim_waktu_etl_*inserted*	dim_waktu (id)	Indicates when this data was entered into the data warehouse. Essential for data audits. Example: "Verify the registration data entered into the system on August 15, 2024."

Student Payment Data Star Scheme

This star scheme shows the relationship between the dimension table table and the student payment fact table. The dimensions used are the time, payment type and transaction partner.

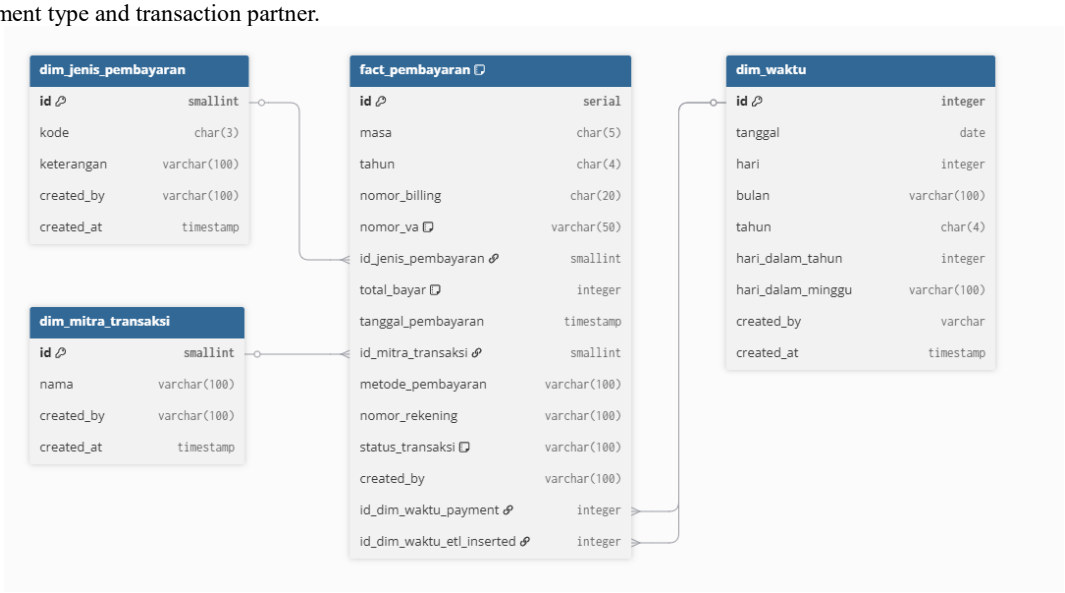


Figure 5 ERD Scheme Star Scheme Student Payment Data

Payment Facts Table Relationship and Dimensions

Table 12 Payment Facts Table Relationships and Dimensions

Table Facts (*Foreign Key*)	Connect to the Dimension Table (*Primary Key*)	Context of the Data Provided (Sample Analysis Questions)
id_jenis_pembayaran	dim_jenis_pembayaran (id)	Provide details about the type of payment made. Example: "What was the total income from the 'Tuition Fee' payment type last month?"
id_mitra_transaksi	dim_mitra_transaksi (id)	Describe the partner or channel where the payment transaction is processed. Example: "Which transaction partner processed the highest payment volume this quarter?"
id_dim_waktu_*payment*	dim_waktu (id)	Provide details of the time (date, month, day) when the payment was made by the customer. Example: "What is the trend of total daily payments over the last month?"
id_dim_waktu_etl_*inserted*	dim_waktu (id)	Indicates when this payment data was entered into the data warehouse. Essential for data audits. Example: "View all payment transactions whose data was entered into the system yesterday.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 1960-1970

Relationship of Employee Facts Table and Dimensions

Table 13 Relationship Employee Fact Table and Dimensions

Table Facts (*Foreign Key*)	Connect to the Dimension Table (*Primary Key*)	Context of the Data Provided (Sample Analysis Questions)
kode_jenis_kelamin	dim_jenis_kelamin (code)	Demographics: Provides information on the gender of employees. Example: "How is the distribution of employees by gender in each work unit?"
kode_agama	dim_agama (id)	Demographics: Provide information about employees' religions. Example: "Show the religious composition of all employees."
kode_status_pegawai	dim_status_pegawai (id)	Staffing: Explains the status of the employee (e.g., civil servant, honorary). Example: "How many permanent employees are compared to non-permanent employees?"
kode_jenis_pegawai	dim_jenis_pegawai (code)	Staffing: Describes the type of employee (e.g., Lecturers, Education Personnel). Example: "Compare the number of lecturers with education staff."
kode_lokasi_pegawai	dim_lokasi_pegawai (code)	Location: Describes the location of the employee's work placement. Example: "Which location has the most employees?"
kode_status_aktivitas	dim_status_aktivitas (id)	Status: Describes the employee's current activity status (for example, Active, Time Off, Study Assignment). Example: "How many employees are currently in the status of a study assignment?"
kode_unit & kode_unit_satminkal	dim_unit (code) & dim_unit_satminkal (code)	Structural: Explain the work unit and the base administrative unit (satminkal) of employees. Example: "Display a list of employees working in the 'Faculty of Engineering' unit."
kode_golongan	dim_pangkat_golongan (code)	Position & Rank: Provide details of the rank and class of employees. Example: "How many employees are in Group III/d?"
kode_jabatan, kode_jabatan_struktural, kode_jabatan_fungsional_umum	dim_jabatan, dim_jabatan_struktural, dim_jabatan_fungsional_umum	Position & Rank: Provides complete information about the position held by the officer. Example: "Who are the employees who hold the structural position of 'Head of Department'?"
kode_jenjang_pendidikan	dim_jenjang_pendidikan (code)	Education: Explain the employee's last level of education. Example: "How is the distribution of employee education levels (S1, S2, S3)?"
id_dim_waktu_tmt_cpns, id_dim_waktu_tmt_pns, id_dim_waktu_tmt_ut, id_dim_waktu_tmt_mulai, id_dim_waktu_*source_created,* id_dim_waktu_etl_*inserted*	dim_waktu (id)	TIME (Various Contexts): Provides date details for various important events in an employee's career (TMT CPNS, TMT PNS, etc.) as well as data processing time. Example: "Show employees who were appointed as civil servants in 2020." or "How long is the average employee's tenure?"

ETL Workflow Management and Monitoring Using Apache Airflow

Apache Airflow was utilized to manage and monitor the ETL workflow. Apache Airflow is an open-source platform designed for scheduling, orchestrating, and monitoring ETL processes using a Directed Acyclic Graph (DAG) approach. DAGs are employed to define task execution order and manage dependencies among tasks, ensuring that the ETL pipeline operates in a structured and controlled manner. The core characteristics of DAGs in Apache Airflow are explained as follows:

1. Directed

Each task in the workflow follows a defined execution path that specifies the processing order. For example, Task A must be completed before Task B, which is represented by a directional arrow connecting the two tasks. This directional relationship indicates that Task A is a prerequisite for Task B, ensuring that the ETL process executes sequentially and preventing tasks from being skipped or executed prematurely.

2. Acyclic

A Directed Acyclic Graph (DAG) does not contain any cyclic dependencies. Each task is executed only once, and no task is allowed to repeat in a loop. This acyclic structure guarantees that all tasks can be completed efficiently without the risk of infinite execution cycles.

3. Graph

The graph structure of a DAG consists of nodes and edges. Nodes represent individual tasks within the ETL workflow, while edges represent dependencies or relationships between tasks. This structure provides a clear and visual representation of the ETL pipeline, enabling effective orchestration and monitoring of task execution.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 1960-1970

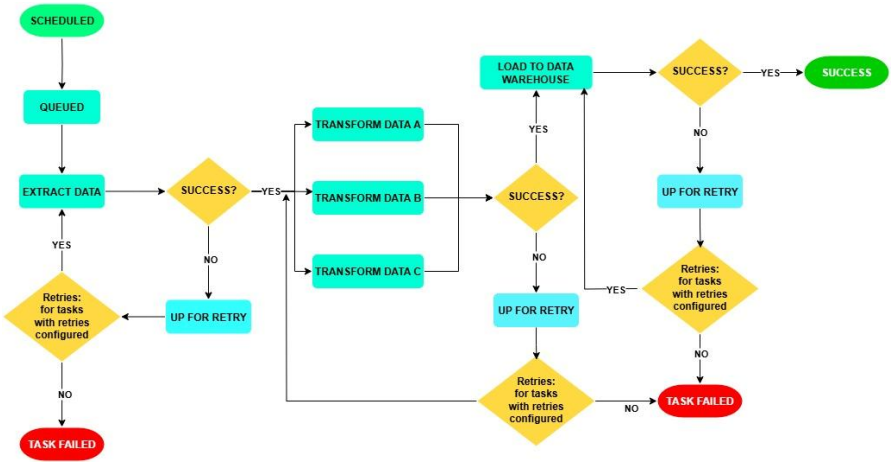


Figure 6 ETL Process Workflow

The selection of Apache Airflow was motivated by its strong capabilities in managing and orchestrating complex workflows. Apache Airflow enables effective dependency management among tasks through the use of Directed Acyclic Graphs (DAGs) within data pipelines. Compared to alternative platforms such as Apache NiFi or Talend, Apache Airflow offers superior scalability and greater flexibility for highly customized integrations.In addition, Apache Airflow provides built-in monitoring and automated scheduling features that ensure the reliability of ETL processes. The platform also supports robust error handling mechanisms, including task retry functionality, which helps minimize workflow disruptions and enhances overall system resilience [20].

Testing Results and Analysis

The following section presents a summary and analysis of the two testing scenarios that were conducted to evaluate system performance and reliability.

Table 14 Stress Test Scenario Analysis Menu Dashboard and Reports

Testing Metrics	Performance Targets	Scenario A Result (Dashboard Menu)	Scenario B Results (Report Menu)
Maximum Virtual Users	100 VUs	100 VUs	100 VUs
Failure Rate	< 3-5%	0.00% (✓ Pass)	0.00% (✓ Pass)
Response Time p (95)	< 5.0 seconds	4.90 seconds (✓ Pass)	3.83 seconds (✓ Pass)
Average Response Time	Informational	3.09 seconds	2.00 seconds

Outcome of Scenario A: Dashboard Menu Stress Test Test

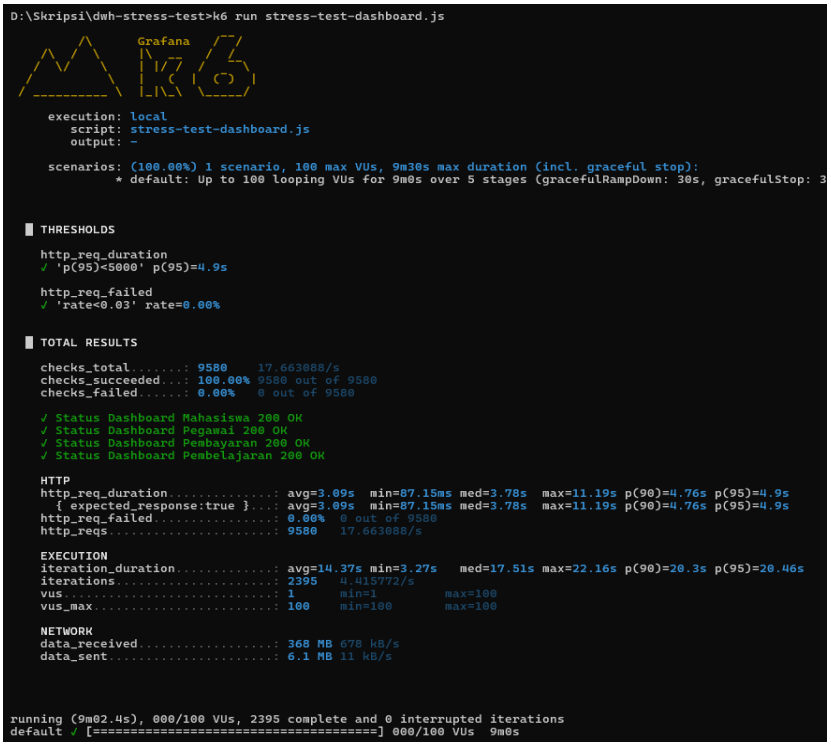


Figure 7 Stress Test Menu Dashboard

The test results show that the dashboard module successfully passed all the set criteria. A 0.00% failure rate proves the application is running stably. The response time of p(95) was recorded at 4.9 seconds, which is below the 5 second threshold. With an average response time of 3.09 seconds, it can be concluded that the dashboard module is capable of handling the load from the predetermined one.

MSW MANAGEMENT -Multidisciplinary, Scientific Work and Management Journal
ISSN: 1053-7899
Vol. 36 Issue 1, 2026, Pages: 1960-1970

Scenario B Result: Stress Test Report Menu

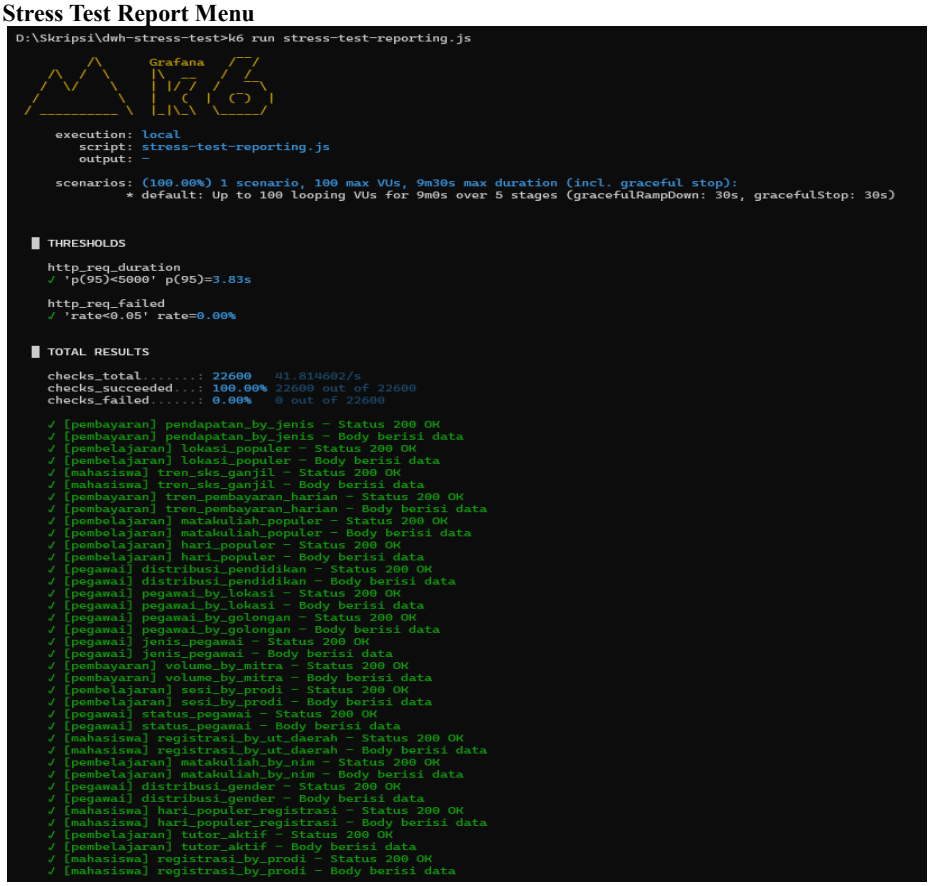


Figure 8 Stress Test Report Menu

The report module also successfully passed the test with better results. The response time of p(95) is only 3.83 seconds with an average response time of 2 seconds. These results show that the application for the report is running responsively. This good performance proves that optimization on the database query and tuning on the database side is going well.

Conclusion of Load Testing

Based on a series of Load Tests conducted, it can be concluded that the data warehouse application developed meets performance and stability standards to handle the expected peak loads.Both the dashboard and report modules are proven to be able to serve up to 100 concurrent users without experiencing failures and with response times within the predetermined threshold limits. This success validates the effectiveness of the optimizations that have been made at the application and database layers. Thus, the system is declared ready for use by users at the Open University under normal conditions of use up to simultaneous access with a high number.

CONCLUSION

The implementation of data warehouses at the Open University can overcome problems in the management and integration of data spread across each unit. Managing ETL (Extract, Transform, Load) processes using Apache Airflow ensures that data loaded into the data warehouse remains consistent, clean, and structured. The selection of a star schema for this data warehouse makes it easier to analyze the data by depicting a clear relationship between the dimension table and the facts. Data marts are built to provide a more specific subset of data on specific units, thus facilitating cross-unit decision-making and analysis.In addition, real-time monitoring and management of ETL psoes flows through Apache Airflow provides an advantage in ensuring the accuracy and sustainability of integrated data. It also allows for an automatic retry process in the event of a failure, thus minimizing disruption to the system.

REFERENSI

A. S, Rosa dan Shalahuddin, M., Rekayasa Perangkat Lunak Terstruktur. Dan Berorientasi Objek, Bandung: Informatika, 2013
Aberer, K., & Hemm, K. (1996, June). A methodology for building a data warehouse in a scientific environment. In Proceedings First IFCIS International Conferenceon Cooperative Information Systems (pp. 90-101).IEEE
B. Sidik, Pemrograman Web dengan PHP, Bandung: Informatika Bandung, 2014
Eder, J., & Wiggisser, K. (2011). Data warehouse maintenance, evolution and versioning.In Enterprise Information Systems: Concepts, Methodologies, Tools and Applications (pp. 566-583). IGI Global.
F. Martin, Panduan Singkat Bahasa Pemodelan Objek Standar, Yogyakarta: Andi, 2005.
Gupta, H., Harinarayan, V., and Rajaraman, A. Index selection for OLAP, in Proc. Int. Conf. Data Engineering (Binghamton, UK, 1997).
Inmon, W. H. (2005). Managing the lifecycle of data. Available on: http://www. sun.com/solutions/documents/whitepapers/bidw_ManagingTheLifecyleOfData.pdf.
J. Widom, editor. Data Engineering, Special Issue on Materialized Views and Data Warehousing, volume 18(2). IEEE, 1995.
J. Widom. Research problems in data warehousing. In Proc. CIKM, pages 25-30, Nov. 1995.
Imhoff, C., Galemmo, N., & G. Geiger, J. (2003). Mastering Data Warehouse Design. Wiley Publishing.
Khouri, S., & Bellatreche, L. (2017). Design life-cycle-driven approach for data warehouse systems configurability. Journal on Data Semantics, 6, 83-111.
Kimball, R., & Ross, M. (2002). The Data Warehouse Toolkit: The Complete. Guide to Dimensional Modeling. John Wiley & Sons
Kimball, R. The data warehouse toolkit. John Wiley & Sons, 1996
Madcoms, Program PHP dan MYSQL untuk Pemula, Yogyakarta: C.V Andi OFFSET, 2016
M. Andi Sunyoto, Ajax Membangun Web dengan Teknologi
Asynchronouse Javascript dan XML, Yogyakarta: Penerbit Andi Yogyakarta, 2007
Redmond, E., & Wilson, J. (2019). Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement. Pragmatic Bookshelf.
Supono dan V. Putratama, Pemrograma WEB dengan menggunakan PHP dan framework CodeIgniter, Yogyakarta: Deepublish, 2016.
Vassiliadis, P., & Simitsis, A. (2009). Extraction, Transformation, and Loading. Encyclopedia of Database Systems, 10.
Watson, H. J. (2002). Recent developments in data warehousing. Communications of the Association for Information Systems, 8(1), 1.
Yasmin, J., Wang, J., Tian, Y., & Adams, B. (2024). An empirical study of developers' challenges in implementing workflows as code: A case study on Apache Airflow. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2406.00180