UNIVERSITY OF ECONOMICS AND LAW

FACULTY OF INFORMATION SYSTEMS

FINAL PROJECT REPORT

TOPIC

BUILDING BUSINESS INTELLIGENCE SOLUTION FOR PURCHASING MODULE FOR ADVENTURE WORKS CYCLES COMPANY

Course: BUSINESS INTELLIGENCE AND

DECISION SUPPORT SYSTEMS

CourseID: 243MI3301

Lecturer: Le Ba Thien, MSc.

Group 9

Ho Chi Minh City, June, 2025

MEMBERS OF GROUP 9

No	Full Name	Student code	Roles	Completion Level
1	Luong Chi Trung	K224060818	Leader	100%
2	Le Thanh Quy Hai	K224060780	Member	100%
3	Cao Phan Trung Hieu	K224060784	Member	100%
4	Tran Đuc Luong	K224060793	Member	100%
5	Nguyen Van Tai	K224060809	Member	100%

ACKNOWLEDGE

The successful completion of this project would not have been possible without the valuable support and guidance we received from various individuals and institutions. We are especially thankful to MSc. Le Ba Thien, lecturer at the Faculty of Information Systems, University of Economics and Law – VNU HCM, whose dedicated mentorship and thoughtful advice played a vital role in shaping our work. His continuous support helped us overcome many challenges throughout the development of the project.

We would also like to express our sincere appreciation to the authors and researchers whose academic work served as key references in our analysis. Their contributions provided essential insights and theoretical foundations, enabling us to better understand the subject and formulate meaningful conclusions. While we strived to ensure the highest quality in our efforts, we acknowledge that some imperfections may remain. Therefore, we respectfully welcome all constructive feedback and criticism to further improve our work and deepen our learning.

We are truly grateful for the opportunity to conduct this research and for the support received along the way. This experience has been invaluable in enhancing our skills and knowledge, and we look forward to future opportunities for growth.

Sincerely,

Group 9

Ho Chi Minh City, June, 2025

COMMITMENT

We hereby declare that this project was independently conducted by our team under the supervision of MSc. Le Ba Thien, and fully reflects our own efforts in researching, designing, and implementing the proposed solution. The process involved building a data-driven system based on reliable sources and prior studies, aiming to generate valuable insights and support informed decision-making.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

This project presents the design and implementation of a Business Intelligence (BI) solution focused on the Purchasing Module of the AdventureWorks database. Using Microsoft's BI ecosystem—SQL Server Integration Services (SSIS), SQL Server Analysis Services (SSAS), and Power BI—the team developed a data warehouse to streamline the extraction, transformation, and loading (ETL) of purchasing data. The solution includes a multidimensional cube for advanced analytics and visual dashboards that provide insights into supplier performance, purchasing trends, and key procurement indicators. This integrated system supports data-driven decision-making and enhances operational visibility in the procurement process.

CHAPTER 1. OVERVIEW OF THESIS

1.1. Overview of the Bicycle Industry, AdventureWorks Cycles Company, and the Business Case

1.1.1. Overview of the Global Bicycle Retail Industry

The bicycle industry has seen strong and steady growth due to global trends such as environmental awareness, health consciousness, and a shift towards sustainable transportation. Bicycles serve not only as eco-friendly means of transportation but also as a popular form of fitness and recreation, driving consistent market expansion.

Figure 1.1. Bicycle Market Size 2025 - 2030 (Source: Mordor Intelligence)

According to a report by Mordor Intelligence, the global bicycle market is expected to grow from USD 58.52 billion in 2025 to USD 71.88 billion by 2030, driven by a CAGR of 4.20% (Figure 1.1). This growth is supported by rising environmental awareness, technological innovation, and strong government support. Manufacturers are leveraging AI to improve product development and supply chain efficiency, while the adoption of smart technologies has given rise to connected

bicycles with GPS, mobile integration, and performance tracking. The electric bike segment, in particular, is seeing advancements in battery and power-assist systems. At the same time, infrastructure investments are accelerating, with countries like the Netherlands and France committing billions to cycling facilities. Product innovation remains strong, with brands introducing models tailored to specific user needs, such as Giant's Voya E+ and PakYak E+ in Germany. The retail landscape is also evolving, combining physical stores and online platforms with added services like repairs and rentals. Incentive programs, such as the Netherlands' policy of reimbursing cyclists USD 0.22 per kilometer, further stimulate demand and reinforce the market's positive outlook.

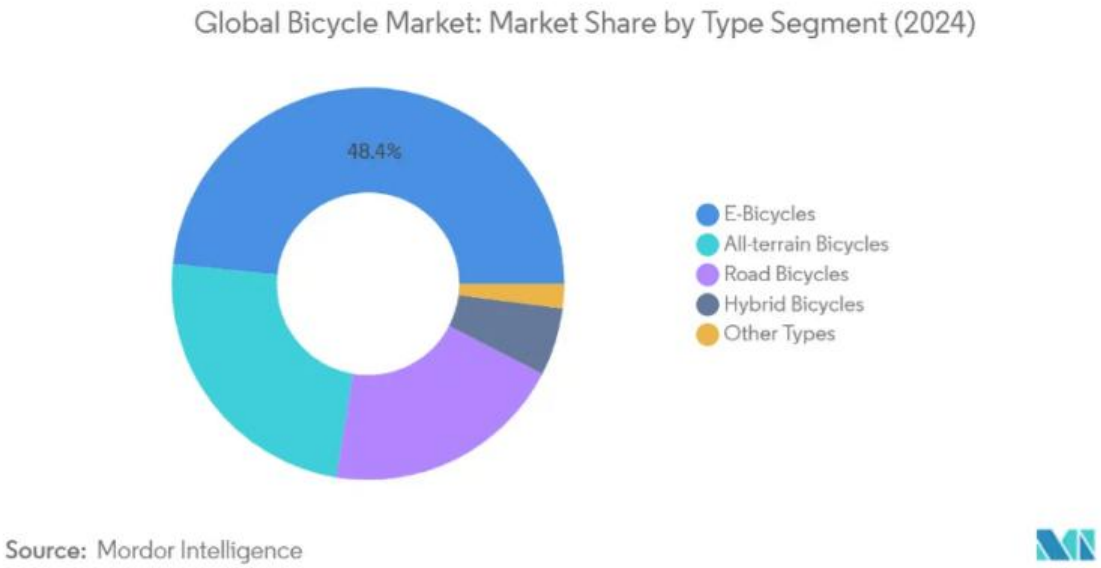


Figure 1.2. Global Bicycle Market: Market Share by Type Segment (2024) (Source: Mordor Intelligence)

As illustrated in Figure 1.2, e-bicycles hold the largest market share in the global bicycle market, accounting for 48.4% in 2024. This dominance reflects the growing consumer demand for convenient, eco-friendly mobility solutions and the increasing adoption of electric bikes in urban and recreational settings. The strong presence of all-terrain and road bicycles further indicates a diversified market catering to various lifestyle and sporting needs.

Figure 1.3. Bicycle Market: Forecasted Five-Year Growth Rate, By Region (Source: Mordor Intelligence)

Figure 1.3 illustrates the forecasted five-year growth rate of the global bicycle market by region, highlighting clear differences across geographic areas. Notably, countries in the Asia-Pacific region such as China, India, Vietnam, and Australia are expected to experience high growth rates. This trend is largely driven by rapid urbanization, growing environmental awareness, and government initiatives supporting green transportation. In Southeast Asia, demand for bicycles is rising significantly due to persistent traffic congestion and the cost-effectiveness of bicycles for daily commuting. In contrast, several regions in Africa, the Middle East, and parts of Europe are projected to see only moderate or low growth, possibly due to market saturation or economic conditions that are less favorable for shifting transportation habits. Meanwhile, countries in the Americas, including the United States and Brazil, fall into the medium growth category, where investments in cycling infrastructure and public health awareness continue to support market expansion. These regional disparities emphasize the need for businesses in the bicycle industry to tailor their investment and market entry strategies to each specific region. According to a report by Mordor Intelligence, this approach is essential for maximizing opportunities and minimizing risks in the global market.

The global bicycle market presents a fragmented and highly competitive landscape, with a wide mix of global corporations and strong regional manufacturers. While large players benefit from scale and brand reach, many smaller or localized companies gain market share through region-specific designs, agile operations, and strong dealer networks. The industry is undergoing steady consolidation, especially through acquisitions in the electric bicycle segment, as companies seek to expand their product offerings and enter new markets. A strong emphasis is being placed on sustainability, with manufacturers adopting eco-friendly materials and production practices to meet environmental standards. Meanwhile, digital transformation continues to shape the market through the development of smart bicycles and direct-to-consumer channels. New business models like subscriptions and urban mobility services also create opportunities for innovative companies to grow. In this environment, success increasingly depends on a company's ability to adapt to shifting consumer preferences, comply with safety and environmental regulations, and manage supply chains effectively. (Source: Mordor Intelligence)

1.1.2. Overview of Adventure Works Cycles Company and the Business Case

Figure 1.4. Adventure Works Cycles (Source: Word Press)

Adventure Works Cycles is a simulated multinational company specializing in the manufacturing and distribution of bicycles and related products. Alongside its core product line of bicycles made from various materials, the company also offers accessories, spare parts, and cycling apparel. A significant portion of these products is procured from third-party suppliers, which makes the company not only a manufacturer but also a reseller.

The company serves both individual customers and retail bike stores through online channels and wholesale distribution. Although it does not operate physical storefronts, Adventure Works relies on digital platforms and business-to-business sales to reach the market.

The Purchasing Department plays a vital role in ensuring the availability of materials and resale products. It oversees the procurement of components needed for bicycle production as well as finished goods sourced for resale, such as helmets, water bottles, and clothing. The available data within the AdventureWorks database contains comprehensive records of purchase orders, vendors, product categories, and transaction histories—providing a strong foundation for analysis.

This project is centered on building a Business Intelligence (BI) solution tailored to the Purchasing module. Tools such as SQL Server Integration Services (SSIS), SQL Server Analysis Services (SSAS), and interactive data visualizations are used to extract, transform, analyze, and present meaningful insights.

By analyzing purchasing trends, supplier performance, and cost patterns, the BI solution is expected to support the department in improving procurement strategies, reducing delays, and managing supplier relationships more effectively. The ultimate goal is to turn raw purchasing data into actionable insights that contribute to better decisions and higher operational efficiency.

To achieve this, our team initiated the project titled "Business Intelligence-Based Procurement Optimization for Adventure Works Cycles", focusing on data-driven improvements that support the company’s growth and competitiveness.

1.2. Objectives

1.2.1. General Objective

The main goal of this project is to develop a Business Intelligence (BI) solution for the Purchasing Department at AdventureWorks Company. This involves researching and applying SSIS and SSAS techniques to build a data warehouse, leveraging OLAP methods and MDX language for data analysis, and utilizing Power

BI to visualize the results. The proposed BI system is expected to provide purchasing staff and company executives with a comprehensive view of the procurement process, enabling them to identify unresolved issues and make better-informed decisions that enhance operational efficiency.

1.2.2. Specific Objectives

Use the AdventureWorks database as a sample database to conduct a comprehensive analysis, understand the data structure and quality, and gain insights into the purchasing process.

Analyze the purchasing workflow and develop a KPI system to effectively evaluate performance.

Propose a suitable data warehouse design that aligns with business requirements and ensures efficient data management.

Develop a new and appropriate data warehouse tailored to the needs of the purchasing subsystem.

Implement the ETL process to optimally integrate source data into the data warehouse.

Apply SSAS to convert raw data into meaningful and actionable business insights.

Use MDX (Multidimensional Expressions) to query data from OLAP cubes to support detailed data analysis.

Build a reporting and visualization system using Power BI to deliver an overall view of inventory status and purchasing trends within the company.

1.3. Subject and research scope of the project

Subject: Developing a Business Intelligence solution for the Purchasing Department of AdventureWorks Company.

Scope: The project scope includes processing and analyzing data from the Purchasing Department in the AdventureWorks dataset, constructing a centralized data warehouse through the ETL process, conducting descriptive data analysis using OLAP cubes, and developing interactive dashboards to visualize sales performance, customer segmentation, and key business metrics. The dataset used in this project includes purchasing data collected between April 16, 2011 and September 22, 2014, providing a sufficient historical range for analyzing procurement trends and supplier performance.

1.4. Tools used

SQL Server Management Studio (SSMS): SSMS is a powerful database management tool developed by Microsoft, used for managing, configuring, and querying SQL Server databases. In this project, we used SSMS to explore the structure of the original data, execute queries for validation, and assist in preparing the source data for the data warehouse.

SQL Server Integration Services (SSIS): SSIS is a specialized tool for handling ETL (Extract, Transform, Load) processes. It was used to design automated workflows that extracted data from transactional (OLTP) systems, transformed and standardized it—including the implementation of Slowly Changing Dimensions (SCD Type 1 and Type 2)—and loaded it into the centralized data warehouse.

SQL Server Analysis Services (SSAS): SSAS is a tool for building OLAP (Online Analytical Processing) cubes, which allow for multidimensional data analysis. In the project, SSAS was used to create analytical cubes that enabled quick retrieval of KPIs such as Received Rate, Return Rate, and Order Quantity across dimensions like time, vendor, and product.

Microsoft Power BI: Power BI is a robust data visualization platform used to create interactive reports and dashboards. We utilized Power BI to present the analysis results through intuitive visuals, making it easier for end users—such as purchasing managers—to monitor key metrics and make timely decisions.

Python: Python, known for its flexibility and efficiency, was used to support data processing tasks such as data cleaning, anomaly detection, and automation of certain preprocessing steps before loading data into the main BI system.

1.5. Research implications

1.5.1. Project Value

Throughout the project implementation, our team gained several valuable experiences and insights, including:

Understanding how to build and operate a data warehouse system: From designing data models and identifying fact and dimension tables to integrating and managing data from multiple sources.

Becoming familiar with and applying Business Intelligence (BI) tools: Including SSIS for ETL processes, SSAS for building analytical cubes, and Power BI for data visualization.

Grasping the mindset required to implement a complete BI solution: Not only technical aspects but also understanding how to define problems, gather user requirements, build KPIs, and design practical solutions.

Learning how to evaluate and measure performance through KPIs: By building indicators such as Received Rate, Return Rate, Order Quantity, etc., and understanding the practical significance of each.

Enhancing analytical thinking and problem-solving skills: Through data handling, error detection, ETL optimization, and drawing meaningful insights to support decision-making.

Developing teamwork and professional communication skills: gained through task delegation, technical collaboration, and presenting the final report to instructors.

1.5.2. Desired Outcomes

With a solid technological foundation and an appropriate implementation methodology, the project aims to deliver in-depth analytical reports that help the company uncover hidden insights and translate them into actionable recommendations. These outcomes are expected to provide management with a clearer understanding of customer segments and support the development of strategies for customer retention, growth, and market expansion.

The project also seeks to improve performance measurement capabilities through standardized KPIs, allowing the business to identify consumer trends promptly and respond effectively to market fluctuations. Most importantly, by offering a multi-dimensional analytical perspective, the project enables the company to discover and capitalize on emerging business opportunities, while optimizing operational processes and long-term strategic planning.

1.5.3. Project Execution and Deliverables

Step 1: Planning

The first phase focuses on gathering and defining requirements from the Purchasing Department. This includes outlining the purchasing workflow, identifying key business needs, and developing a system of Key Performance Indicators (KPIs). Functional requirements are also clarified to ensure the BI solution aligns with operational demands.

Step 2: Data Preparation

Raw data from OLTP (Online Transaction Processing) systems is collected and transformed into a Purchasing Data Mart. Exploratory Data Analysis (EDA) is conducted to understand data characteristics and design a suitable structure for the upcoming processing. The objective is to organize data effectively to support the ETL process.

Step 3: ETL Processing

In this step, data is extracted, transformed, and loaded into a centralized Data Warehouse. Key tasks include updating existing records (using Slowly Changing Dimensions types 1 and 2) and inserting new ones. Lookup tables are applied for data matching and validation to ensure accuracy before analysis. After this phase, the data is ready for in-depth analytics.

Step 4: Data Analytics

Processed data is analyzed using OLAP cubes built with SQL Server Analysis Services (SSAS). A KPI system is also developed to monitor performance against predefined criteria. This provides a solid foundation for drawing insights and supporting decision-making.

Step 5: Visualization and Reporting

Analyzed data is visualized through dynamic, interactive reports built in Power BI. These reports are deployed on the Microsoft Fabric platform, making them easily accessible and scalable. As a result, stakeholders can quickly grasp essential information and make informed decisions.

1.6. Structure of the report

Chapter 1: Introduction

This chapter provides an overview of the bicycle market and introduces AdventureWorks—the primary subject of this study. It identifies existing issues within the purchasing process, analyzes the need for improvement, and proposes a solution based on Business Intelligence (BI) technologies. Additionally, the chapter outlines the research objectives, scope, target users, tools employed, and the practical significance of the proposed BI solution for the company.

Chapter 2: Theoretical Foundation

This chapter establishes the theoretical basis for implementing a BI solution in the purchasing department. It covers the concepts and benefits of Business Intelligence, the process of building a BI system, and detailed explanations of the

ETL (Extract, Transform, Load) process. Furthermore, it introduces the use of Key Performance Indicators (KPIs) to measure performance, along with the fundamentals of MDX language, OLAP techniques, and Power BI as tools to support effective data analysis.

Chapter 3: User Requirement Analysis and Data Description

This chapter analyzes the business requirements of the purchasing department at AdventureWorks and develops a set of suitable KPIs accordingly. It also presents the data preparation steps, including data cleaning, data warehouse design, and data modeling. The chapter concludes with a detailed description of the ETL process implemented using SSIS, incorporating Slowly Changing Dimension (SCD) techniques to manage historical data changes effectively.

Chapter 4: Data Analysis and Modeling

This chapter explains how the team utilized SQL Server Analysis Services (SSAS) to build a data analysis model (cube) and applied MDX language to query the data warehouse. It continues with the development of KPIs and performs data analysis through dashboards in Power BI, allowing the team to derive insights, identify trends, and propose specific strategic recommendations for the business.

Chapter 5: Conclusion

This final chapter summarizes the key outcomes of the project and highlights the significance of the BI system in enhancing the purchasing department's operations. The chapter also discusses the limitations encountered during the project and proposes future directions, such as integrating ERP systems or leveraging AI/ML technologies for demand forecasting.

CHAPTER 2. THEORETICAL BASIS

2.1. Overview of BI

2.1.1. BI Models and Solutions

Business Intelligence is a collection of processes that includes business analysis, data mining, data visualization, data infrastructure and tools, along with best practices, to help businesses analyze their business information data. This process requires the collection, storage, analysis, and visualization of data from various sources within the organization.

Figure 2.1. Modern BI Models and Solutions (Source: Authors, 2025)

In this project, the system was designed based on the modern Business Intelligence (BI) architecture, which encompasses several key layers, including data sources, ETL processing, data warehousing, data modeling, and visualization. To meet the complex analytical requirements of the purchasing domain, we adopted the Galaxy Schema (also known as the Fact Constellation model) as the foundation for data organization.

This schema allows multiple fact tables to share common dimension tables, providing a flexible and scalable structure for multidimensional analysis. In our implementation, the data warehouse includes two main fact tables: one for recording purchasing transactions and another for capturing periodic inventory snapshots. Both fact tables are linked to shared dimensions such as product, vendor, employee, time, and location. The use of this schema facilitates integrated analysis across processes such as procurement and inventory management, while ensuring consistency in data interpretation.

The Galaxy Schema model proves to be well-suited for enterprise BI systems, especially when business processes are interconnected and require simultaneous monitoring across several analytical dimensions.

2.1.2. Benefits of BI in Business

Business Intelligence (BI) offers a range of significant benefits for businesses that choose to implement and leverage its capabilities. Below are insights into the advantages of BI in business operations:

Informed, Data-Driven Decisions: Business Intelligence (BI) enables timely and reliable decision-making by providing real-time, organized data. Instead of relying on intuition, managers use data insights to evaluate strategies, reallocate resources, and minimize risks. For example, a retailer can accurately forecast restocking needs by analyzing consumer trends, improving both efficiency and customer satisfaction.

Improved customer satisfaction: BI enables businesses to gain deeper insights into customer needs and behavior, allowing for personalized experiences and services. When organizations understand customer preferences and reactions, they can tailor messaging, products, or promotions to enhance satisfaction. Platforms like Netflix and Shopee leverage this capability effectively to improve user retention.

Improved data access: Modern BI systems integrate data from various sources and offer role-based access to users. Instead of relying on the IT department, business units can directly extract the information they need through intuitive dashboards, boosting responsiveness and reducing delays in data flow.

Enhanced risk management: BI supports risk mitigation by monitoring performance indicators and detecting anomalies. From finance to operations, BI systems can provide early warnings when a metric exceeds thresholds or deviates from norms. For example, banks use BI to identify suspicious transaction patterns and activate fraud prevention mechanisms.

Cost savings: BI offers in-depth visibility into financial operations, production costs, logistics, or marketing efficiency, enabling businesses to eliminate wasteful spending. By analyzing historical data and performance metrics, organizations can spot inefficiencies, streamline operations, or negotiate better supplier terms.

Improved operational efficiency: By analyzing workflows and measuring internal KPIs, BI helps identify non-value-adding steps or bottlenecks in processes. This enables process improvement, better workforce allocation, and automation of repetitive tasks, leading to smoother daily operations.

Data visualization: The ability to convert raw data into charts, interactive tables, and heatmaps makes trends and insights more accessible. Data visualization not only simplifies data interpretation but also improves internal communication by presenting complex information clearly and persuasively.

Identifying market trends: BI supports market trend analysis by processing customer and market data to uncover new patterns or shifts in behavior. Instead of reacting passively, businesses can proactively adjust product strategy, marketing direction, or market expansion plans—gaining a competitive edge.

Better forecasting: Forecasting powered by BI is based on historical data and statistical or machine learning models. This allows companies to plan more accurately for finance, procurement, production, or workforce needs, minimizing understocking or overcapacity during peak periods.

Business Intelligence offers substantial benefits to organizations by transforming raw operational data into meaningful insights that support timely and informed decision-making. In our project, BI was applied to empower the purchasing department of AdventureWorks with enhanced visibility and analytical capabilities.

Through the use of real-time dashboards and historical trend analysis, managers can now monitor procurement costs, order volumes, vendor performance, inventory availability, and stockout risks with greater accuracy. Instead of relying on intuition, decision-makers are equipped with data-driven insights to detect supply

chain inefficiencies, assess supplier reliability, and identify replenishment needs. The integration of BI into daily operations enables more proactive management, better resource allocation, and strategic improvements across the procurement function.

By embedding BI into the purchasing workflow, the organization benefits from increased operational transparency, improved responsiveness, and a stronger foundation for long-term planning.

2.1.3. BI Solution Implementation Process

Step 1: Define Business Requirements

Implementing a BI system must start with clearly capturing the business requirements of the enterprise, especially from departments directly using data like Purchasing. In this step, the project team works closely with stakeholders to understand the information they need to track, the specific goals they aim to achieve through the BI system, and current difficulties in accessing data. Business expectations, risks, and priorities will be documented in a Business Requirements Document (BRD), forming the foundation for subsequent design steps.

Step 2: Assess Current Environment:

Once business requirements are clear, the project team needs to conduct a comprehensive assessment of the current state of the technology infrastructure and existing data systems. This includes reviewing input data sources, checking data quality and completeness, identifying existing analytical tools (if any), and understanding existing processing flows within the Purchasing department. This practical assessment helps identify strengths to leverage and data gaps or technical limitations to address when building the BI solution.

Step 3: Conceptualize the BI Solution:

Based on the gathered information and assessment, the implementation team will conceptualize the BI solution – specifically defining what the system needs to do and who it will serve. This phase focuses on identifying essential functions (such as

data filtering, time-based analysis, drill-down, report exporting) and non-functional requirements (security, performance, scalability). These requirements will be clearly categorized and prioritized, forming a complete System Requirements Specification (SRS) as the basis for technical design.

Step 4: Design the BI Solution Architecture:

After defining the goals and required functions, the technical team will begin designing the overall architecture for the BI system. This design must clearly illustrate the data flow from extraction to presentation for end-users, including ETL/ELT processing architecture, data warehouse model, analytical tools, and reporting interfaces. The choice of data model (star schema, snowflake, or flat table) will also be decided at this step to ensure efficiency and maintainability.

Step 5: Choose Deployment Strategy:

At this stage, the business needs to decide how the BI solution will be deployed: on-premise, cloud-based, or a hybrid of both. This decision depends on budget, security requirements, infrastructure management capabilities, and long-term development direction. For example, if data is highly sensitive or subject to legal constraints, an on-premise deployment model may be more suitable; while businesses desiring flexibility and rapid scalability will prioritize cloud.

Step 6: Select Appropriate Technologies:

After determining the deployment strategy, the next step is selecting the technology stack to implement the system. This includes choosing ETL tools, data warehouse platforms, visualization software, and related supporting technologies like programming languages, integration APIs. Factors such as compatibility with existing systems, licensing costs, internal capabilities, and scalability will play crucial roles in the final decision.

Step 7: Develop Detailed Implementation Plan:

With the technical components defined, the project team will build a specific implementation plan, including timelines, task assignments, budget estimates, and risk management scenarios. A good strategy often starts with deploying a Minimum Viable Product (MVP) – focusing on critical functions first – before gradually expanding. Clearly defining priorities and resources at this step ensures the project deploys on schedule and achieves high efficiency.

Step 8: Develop BI Components:

This is the stage of realizing the system through building data processing pipelines, designing data models, developing reports, dashboards, and user interface components that interact with users. The technical team will connect data sources, write processing logic (ETL), build the Data Warehouse, and ensure the final data is presented clearly, visually, and is easy to use. Naming conventions, data testing procedures, and quality control methods are also established here.

Step 9: Testing and User Training:

Before the official deployment, the system needs comprehensive testing to ensure stability and accuracy. Testing focuses not only on technical functionality but also on data accuracy and user experience. Subsequently, the organization will conduct training for end-users – those who directly utilize data from the system – to ensure they can use BI effectively, understand reports clearly, and face no difficulties accessing data.

Step 10: Official Deployment and Operation:

The final phase is putting the BI system into actual operation. Data will be updated periodically, processing workflows are automated, and the system begins serving daily business activities. Simultaneously, the business needs to establish monitoring, maintenance, data update, and incident handling procedures. This is also when user feedback will be collected for continuous system improvement over time, ensuring BI becomes not just a technical tool but truly the organization's "data brain".

2.2. Overview of ETL

ETL Process Concept

ETL, an acronym for Extract, Transform, and Load, is a three-step data integration process that combines data from various sources into a unified and consistent data warehouse. This process consists of three main stages: extracting data from multiple sources, transforming the data to meet specific standards, and loading it into the target system, such as a data warehouse.

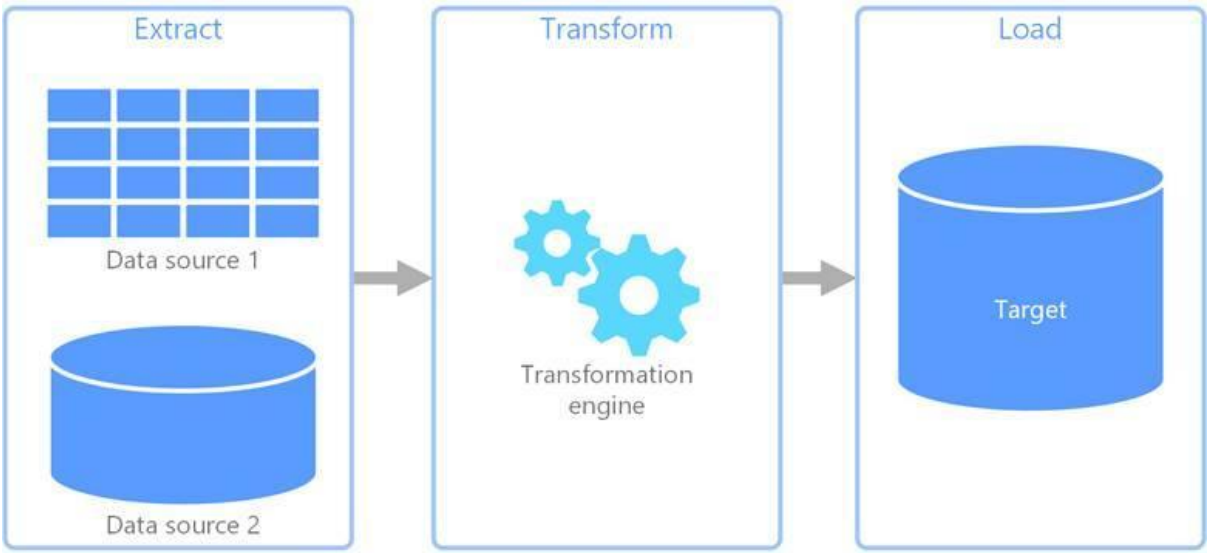


Figure 2.2. Steps in ETL Process (Source: Microsoft Learn)

Key Steps in the ETL Process:

Extract (Data Extraction)

The first stage in ETL is extracting data from source systems. The goal is to collect all necessary data while ensuring completeness and avoiding disruption to the source system's operations. Input data may come from:

- Relational databases (SQL Server, MySQL, PostgreSQL...)
- Non-relational databases (NoSQL like MongoDB, Cassandra)
- Raw files (CSV, Excel, JSON, XML)
- APIs from other systems (REST, SOAP)

- Cloud services (Google Cloud, AWS S3, Azure Blob)
- Legacy systems still in operation

Extraction can be performed in various modes: scheduled (batch), event-triggered, or real-time (streaming), depending on data freshness and speed requirements.

Transform (Data Transformation)

Once collected, data requires processing to become usable in the target system. This stage demands deep technical and business understanding. Typical operations include:

- Data cleansing: Removing missing, duplicate, or malformed data.
- Value standardization: Unifying date formats, measurement units, product codes.
- Format conversion: e.g., XML to JSON or relational tables.
- Applying business logic: Calculating profit margins, classifying customers by lifetime value, risk labeling based on behavior…
- Integrating data from multiple sources: e.g., joining transaction records with customer profiles.

This step converts raw data into meaningful information while restructuring it for analytical readiness. It also establishes a unified data format for future storage and utilization.

Load (Data Loading)

The final stage loads processed data into the target system. Depending on system scale and purpose, enterprises may choose these strategies:

- Full Load: Suitable for initial loads or small systems.
- Incremental Load: Only new or changed data (delta).

- Real-time Load: For systems requiring continuous updates (e.g., fraud detection, customer behavior tracking).
- Batch Processing: Periodic aggregation (hourly/daily/weekly).

Target systems are typically Data Warehouses (Snowflake, Amazon Redshift, Google BigQuery) or Data Lakes for more flexible, diverse data storage.

The Extract – Transform – Load (ETL) process serves as a vital component of any BI system, ensuring that data is accurate, reliable, and consistently structured. In this project, we used SQL Server Integration Services (SSIS) to develop automated ETL workflows that retrieve data from various operational sources, including purchase orders, product catalogs, vendor information, and employee assignments.

Once extracted, the data undergoes multiple transformation steps such as cleaning invalid values, mapping codes through lookup tables, enriching records with calculated fields like inventory status or date keys, and preparing historical tracking for dimension tables. For attributes that evolve over time—such as vendor ratings or addresses—the project implements Slowly Changing Dimensions, allowing us to preserve data history and support accurate time-based analysis.

After transformation, the processed data is loaded into the warehouse under the Galaxy Schema structure, where it becomes immediately accessible for analytical modeling in SSAS and visualization in Power BI. The ETL process, as implemented, is not only technically robust but also business-aware, aligning closely with procurement workflows and ensuring that the BI system remains dependable and relevant over time.

2.3. Data warehouse and data marmart

2.3.1. Data warehouse

A data warehouse (DW) is a digital storage system that connects and harmonizes large amounts of data from many different sources. Its purpose is to feed business intelligence (BI), reporting, and analytics, and support regulatory

requirements – so companies can turn their data into insight and make smart, data-driven decisions. Data warehouses store current and historical data in one place and act as the single source of truth for an organization.

Data flows into a data warehouse from operational systems (like ERP and CRM), databases, and external sources such as partner systems, Internet of Things (IoT) devices, weather apps, and social media – usually on a regular cadence. The emergence of cloud computing has caused a shift in the landscape. In recent years, data storage locations have moved away from traditional on-premises infrastructure to multiple locations, including in private and public clouds. A typical data warehouse often includes the following elements:

- A relational database to store and manage data
- An extraction, loading, and transformation (ELT) solution for preparing the data for analysis
- Statistical analysis, reporting, and data mining capabilities
- Client analysis tools for visualizing and presenting data to business users
- Other, more sophisticated analytical applications that generate actionable information by applying data science and artificial intelligence (AI) algorithms, or graph and spatial features that enable more kinds of analysis of data at scale.

Benefits of a Data Warehouse

Data warehouses offer the overarching and unique benefit of allowing organizations to analyze large amounts of variant data and extract significant value from it, as well as to keep a historical record.

Four unique characteristics (described by computer scientist William Inmon, who is considered the father of the data warehouse) allow data warehouses to deliver this overarching benefit. According to this definition, data warehouses are

- *Subject-oriented*: They can analyze data about a particular subject or functional area (such as sales).
- *Integrated:* Data warehouses create consistency among different data types from disparate sources.
- *Nonvolatile:* Once data is in a data warehouse, it's stable and doesn't change.
- *Time-variant:* Data warehouse analysis looks at change over time.

A well-designed data warehouse will perform queries very quickly, deliver high data throughput, and provide enough flexibility for end users to "slice and dice" or reduce the volume of data for closer examination to meet a variety of demands—whether at a high level or at a very fine, detailed level. The data warehouse serves as the functional foundation for middleware BI environments that provide end users with reports, dashboards, and other interfaces.

In data warehousing, a schema defines how data is logically organized and how the relationships between different tables are structured. Choosing the right schema design is crucial for ensuring efficient data storage, retrieval, and analysis. The most commonly used schemas in data warehouse design include:

- *Star Schema:* This is the simplest and most widely used schema. It consists of a central fact table connected directly to several denormalized dimension tables. The structure resembles a star, which makes it easy to understand and query. It's ideal for straightforward reporting and quick data retrieval.
- *Snowflake Schema:* In this schema, dimension tables are normalized into multiple related tables, forming a structure that looks like a snowflake. While it reduces data redundancy and improves data integrity, it requires more complex queries due to additional joins.

- *Galaxy Schema (Fact Constellation Schema):* A more complex schema that involves multiple fact tables sharing dimension tables. It is used in large-scale data warehouses where different business processes (e.g., sales and inventory) are tracked simultaneously. It provides flexibility and supports more complex analysis.

These schemas serve different purposes depending on the business needs, query complexity, and data volume. Understanding their differences helps organizations design a data warehouse that balances performance, scalability, and maintainability.

2.3.2. Data Mart

A Data Mart is a type of data warehouse that contains data *specific* to a particular business line or department rather than an entire enterprise.

Because data marts contain a smaller subset of data, they enable a department or business line to discover more focused insights more quickly than is possible when working with the broader data warehouse data set.

For example, a marketing team might use a data mart to define ideal target demographics, while a product team might use one to analyze inventory patterns.

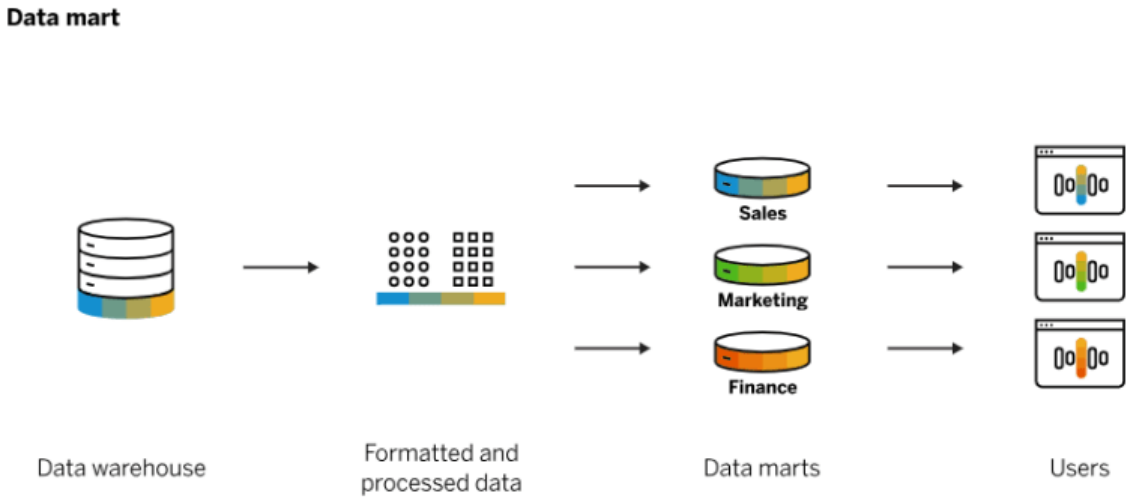


Figure 2.3. Diagram of a data mart and how it works (Source: Velog)

In our project, we built a data warehouse using the Galaxy Schema model, with data sourced from the AdventureWorks 2019 database. The main objective is to analyze the performance of the Purchasing Department, providing insights to support better decision-making and operational efficiency.

2.4. KPIs

A performance indicator or key performance indicator (KPI) is a type of performance measurement. KPIs evaluate the success of an organization or of a particular activity (such as projects, programs, products and other initiatives) in which it engages. KPIs provide a focus for strategic and operational improvement, create an analytical basis for decision making and help focus attention on what matters most.

Key performance indicators are used in business to judge performance and progress toward specific, measurable goals. As such, they provide owners and managers with an overview of how their business, or an aspect of their business, is performing at a certain point in time. They may be compared to:

- A predetermined benchmark
- Other competitors within the industry
- The performance of the business over time

Also referred to as key success indicators (KSIs), KPIs vary between companies and between industries, depending on performance criteria. For example, a software company striving to attain the fastest growth in its industry may consider year-over-year (YOY) revenue growth as its chief performance indicator. Conversely, a retail chain might place more value on same-store sales as the best KPI metric for gauging growth.

At the heart of KPIs lie data collection, storage, cleaning, and synthesizing. The KPI data is gathered and compared to the target that has been set. The results of that comparison are analyzed and used to draw conclusions about how well current

systems or recent changes to those systems are working to achieve the department's or business's goals. This allows management to determine whether the current systems are effective and whether adjustments are needed to improve those outcomes and meet future goals.

The goal of KPIs is to communicate results succinctly to allow management to make more informed strategic decisions. They are often measured using analytics software and reporting tools.

As part of the project, the team designed and implemented a Key Performance Indicator (KPI) system to evaluate the operational effectiveness of the Purchasing Department at AdventureWorks, using data from the AdventureWorks 2019 sample database. The selected KPIs are practical and closely aligned with real-world procurement and supply chain processes. These KPIs were implemented within an OLAP cube using SQL Server Analysis Services (SSAS), with calculations written in MDX language. The results were visualized through interactive Power BI dashboards, enabling purchasing managers and team members to monitor performance trends, evaluate suppliers, and make timely, data-driven decisions. This KPI system plays a crucial role in improving procurement efficiency, enhancing supply chain visibility, and supporting strategic decision-making in a data-driven environment.

2.5. OLAP technique and MDX language for analyzing multi-dimension data

2.5.1. OLAP technique

OLAP (Online Analytical Processing) is an advanced data analysis technique that enables users to efficiently retrieve, process, and analyze data across multiple dimensions. It is a fundamental component of data warehouse systems, where data is structured into multidimensional formats to support in-depth analytical needs.

Figure 2.4 Online Analytical Processing (Source: GeeksforGeeks)

At the core of OLAP lies the OLAP cube, a multidimensional structure that extends the traditional table format by incorporating multiple analytical dimensions such as time, product, vendor, etc. Each dimension can be broken down into various hierarchical levels (e.g., year → quarter → month). While OLAP cubes are theoretically capable of supporting unlimited dimensions (hypercube), in practice, designers implement only the necessary dimensions to optimize both analytical capability and system performance.

Among the OLAP architectures, MOLAP (Multidimensional OLAP) stands out as the most efficient and widely used. It pre-processes, aggregates, and stores data directly in physical cubes, allowing for extremely fast query response times. MOLAP supports a range of analytical operations, including:

- Drill-down: Navigating from summary data to more detailed levels within a hierarchy, allowing users to explore data granularity.
- Roll-up: The reverse of drill-down; it summarizes detailed data to higher-level aggregates to reveal broader trends.
- Slice: Refers to extracting a subset of the cube by fixing a single value from one dimension.
- Dice: Involves selecting a range of values across two or more dimensions.

- Pivot: Rotating the cube's perspective to view the data from different angles and uncover new relationships.

The key advantage of MOLAP lies in its speed and responsiveness. Since the data is already aggregated and indexed within the cube, complex queries and visualizations can be processed in near real time. This makes MOLAP highly suitable for Business Intelligence (BI) applications such as KPI analysis, interactive dashboards, and real-time decision support.

While ROLAP and HOLAP are also mentioned in OLAP theory:

- ROLAP queries relational databases directly without physical cubes, offering better scalability but slower performance.
- HOLAP combines MOLAP and ROLAP strengths but involves higher complexity and maintenance costs.

In this project, the team implemented MOLAP using SQL Server Analysis Services (SSAS) to build and analyze data cubes for the Purchasing Department of AdventureWorks. The use of MDX (Multidimensional Expressions) to query the cubes allows for flexible and accurate access to multidimensional insights, optimizing the overall BI effectiveness.

2.5.2. MDX language

First introduced in the late 1990s by Panorama Software and Microsoft, MDX (Multidimensional Expressions) has since evolved into a widely adopted query language, now considered a non-proprietary standard by many multidimensional database vendors. It was designed to facilitate the querying and analysis of data stored in OLAP cubes, using a syntax that is both powerful and user-friendly.

MDX is widely used in SQL Server Analysis Services (SSAS) and has the capability to generate multidimensional query results, including both axis data and cell data – which represent the intersection of selected dimensions. Compared to traditional SQL, which operates on two-dimensional tables, MDX enables analysis

from multiple perspectives simultaneously, making it highly suitable for complex business intelligence requirements.

A basic MDX query typically consists of the following components:

- WITH clause: Allows the definition of calculated expressions or named sets that can be reused within the query. This improves logical handling and enhances query performance.
- SELECT clause: Specifies which dimension members will be displayed on the axes — usually ROWS and COLUMNS. Each axis contains a set of members to display results in a multidimensional format.
- FROM clause: Identifies the name of the cube to be queried. This cube serves as the data source, containing all preprocessed and structured multidimensional data.
- WHERE clause (also known as a slicer): Used to filter data by one or more specific members of a dimension. It restricts the analytical context without altering the axis layout. Note that the WHERE clause in MDX does not remove rows like in SQL; instead, it slices the data based on the selected dimension member.

By correctly applying and adhering to the role of each component in the query structure, MDX enables queries to be flexible, readable, and optimized for multidimensional data analysis. This structure not only reflects how data is organized within the cube but also ensures that users can retrieve information accurately and extend their analysis according to practical needs.

2.6. Power BI

2.6.1. Definition of Power BI

Power BI is a powerful data analytics and visualization platform developed by Microsoft. It allows users to connect to a wide range of data sources, process and model that data, and transform it into interactive reports and visual dashboards. This

tool supports businesses in making data-driven decisions swiftly while also facilitating the digitalization of reporting and performance monitoring processes.

Thanks to its user-friendly interface and high level of interactivity, Power BI enables even non-technical users to easily build dynamic dashboards and explore data in real time. Its interactive features allow for flexible filtering, data slicing, and in-depth analysis, helping users uncover valuable insights to guide managerial and operational decisions.

Unlike traditional static reporting tools, Power BI offers a modern, interactive working environment. It is deeply integrated with Microsoft's ecosystem—such as Excel, Azure, SharePoint, and Teams—as well as popular data platforms like SQL Server, Google Analytics, Salesforce, and SAP. Its high compatibility and ease of deployment have made Power BI a widely adopted solution across modern organizations, from analysts to executives.

Figure 2.5. Power BI (Source: Adtimin)

2.6.2. Core Components of Power BI

The Power BI ecosystem is built around three core components: Power BI Desktop, Power BI Service, and Power BI Mobile. Each plays a distinct role within the data analysis and visualization pipeline. In addition, Microsoft provides

supporting tools to enhance the overall reporting workflow within business environments.

Power BI Desktop is a desktop application considered the central workspace for building reports. Users can:

- Connect to various data sources, including Excel, SQL Server, PostgreSQL, web APIs, and more.
- Perform ETL (Extract – Transform – Load) processes to clean and prepare data.
- Design data models by defining relationships between tables and creating calculated columns or measures using the DAX language.
- Generate interactive reports using charts, dashboards, KPIs, maps, and other visuals.

Power BI Service (also known as Power BI Online) is a web-based platform that allows users to publish their reports from Power BI Desktop to the cloud. Within this environment, users can:

- Store, present, and share reports in real time.
- Schedule data refreshes to ensure up-to-date information.
- Manage user access and distribute content securely across departments.

Power BI Mobile is an application available on Android and iOS devices. It provides quick access to dashboards and reports stored on Power BI Service, allowing users to monitor key metrics and make decisions anytime, anywhere. The mobile interface is optimized for touch interaction and supports data alerts when thresholds are breached.

In addition to these core components, Power BI includes several supplementary tools that complete the data analysis lifecycle:

- *Power BI Gateway* acts as a bridge between on-premises data sources and Power BI Service. It ensures continuous and secure synchronization of internal data to the cloud.

- *Power BI Report Server* is an on-premises deployment option designed for organizations with strict data security requirements or those not yet ready for cloud adoption. It enables storage and access to reports within a company's local network.

- *Power Query and Power Pivot* are integrated tools within Power BI Desktop. Power Query facilitates data cleansing and transformation, while Power Pivot supports high-performance data modeling and advanced calculations using DAX.

2.6.3. How Power BI Works

The operation of Power BI can be described through four sequential phases, forming a closed-loop workflow for data analysis and visualization. Each phase plays a vital role in transforming raw data into meaningful insights that support decision-making.

Figure 2.6. Power BI Data Integration and Visualization Flow (Source: CRMVIET)

2.6.3.1. Data Connection

The first stage in the Power BI data workflow involves establishing connections to data sources. Power BI supports a wide variety of connectors—over one hundred in total—ranging from simple file-based inputs such as Excel and CSV

to more advanced relational database systems like SQL Server, Oracle, and PostgreSQL. In addition, it offers seamless integration with various cloud-based platforms including Google Analytics, Salesforce, Azure, and Dynamics 365, as well as Microsoft's own enterprise tools such as SQL Server Analysis Services (SSAS) and SQL Server Integration Services (SSIS). Power BI can directly query analytical models built in SSAS or consume output data that has been processed and prepared through SSIS ETL packages.

Thanks to this high level of interoperability, Power BI serves as an effective solution for unifying data from fragmented systems, laying a strong foundation for analysis, visualization, and data-driven decision-making.

2.6.3.2. Data Cleaning and Transformation

Once data is connected, it typically requires cleaning and transformation before it becomes usable. Power BI leverages the Power Query Editor to facilitate data preparation in an intuitive, code-free environment.

Users can eliminate invalid records, normalize data formats, split or merge columns, convert data types, and define relationships between tables. This step is crucial in ensuring data consistency and accuracy, which directly affects the reliability of analytical results.

2.6.3.3. Data Visualization

Cleaned data is then used to design reports. Power BI offers an extensive library of visualization options, including bar charts, line graphs, pie charts, geographic maps, waterfall charts, combo charts, and KPI indicators.

Users can create visually appealing and interactive dashboards through simple drag-and-drop actions. Notably, features such as *drill-down analysis* and *dynamic filtering* allow users to explore data hierarchically and extract insights based on specific contexts, all within the report interface.

2.6.3.4. Publishing and Sharing

Once completed, reports can be published to the Power BI Service platform. This is a cloud-based workspace where reports are stored, synchronized, and made accessible to end users.

Power BI Service supports *scheduled data refresh*, ensuring that reports reflect the most recent data available. A robust permission system allows for precise control over who can view, edit, or export reports. Furthermore, Power BI reports can be *embedded* into internal platforms like SharePoint, Microsoft Teams, or corporate websites—broadening accessibility while maintaining data governance.

CHAPTER 3: USER REQUIREMENTS ANALYSIS AND DATA DESCRIPTION

This chapter comprehensively presents the process of building a Business Intelligence (BI) solution for the Purchasing Department at AdventureWorks. It begins with an analysis of the roles, organizational structure, and specific operational processes of the Purchasing Department – the foundation for clearly defining business requirements, key performance indicators (KPIs), and necessary system functionalities. Based on this foundation, the chapter continues by discussing BI use cases and analysis types suitable for needs such as cost tracking, supplier performance management, and order management. The data preparation and understanding process is carried out through data source analysis and the construction of a specialized Data Mart, leading to the design of a star-schema data warehouse with fact and dimension tables. Finally, the chapter describes the ETL process with data integration strategies, Slowly Changing Dimension (SCD) handling, and execution using SSIS tools, ensuring data is transformed and loaded into the analytical system stably, consistently, and efficiently.

3.1. Purchasing Department

3.1.1. Purchasing Department at AdventureWorks

In the context of modern industrialization, the Purchasing Department at AdventureWorks plays an essential role in ensuring a stable, efficient, and production-requirement-aligned input supply chain. As a large-scale manufacturing enterprise specializing in sports equipment, mechanical components, and bicycles, AdventureWorks heavily relies on the availability, quality, and cost of externally sourced raw materials and components. Therefore, the Purchasing Department does not merely perform ordering functions but also acts as a strategic hub helping to control input risks, optimize costs, and enhance the overall competitiveness of the enterprise.

Firstly, the Purchasing Department is responsible for planning and coordinating supply activities aligned with the company's production plans and

development strategies. Purchasing decisions must be based on analysis of actual demand, storage capacity, market trends, and raw material price fluctuations. The goal is not only to "purchase enough" but also to "purchase correctly": the right type, right quality, right time, and right supplier. This requires the purchasing team to maintain a reliable supplier network while constantly updating market trends to flexibly adjust purchasing policies.

Furthermore, the Purchasing Department acts as a bridge in building sustainable partnerships with strategic suppliers. Instead of a short-term transactional approach, AdventureWorks' modern purchasing model focuses on long-term collaboration, where both parties share information, improve quality, and develop products. In some cases, the Purchasing Department participates in new product design by coordinating with the R&D department to source better-quality or lower-cost alternative materials.

From an internal control perspective, the Purchasing Department is also the frontline in ensuring compliance with the company's financial policies, business ethics, and sustainability standards. Supplier selection is not based solely on cost factors but also requires evaluation of production capacity, social responsibility, environmental commitments, and compliance with international laws – especially within the increasingly complex globalized supply chain context.

In summary, the Purchasing Department at AdventureWorks holds a core role in ensuring continuity, efficiency, and quality for production activities – from supply chain strategy planning, cost control, and partner development to risk management. Organizing this function well not only helps the enterprise avoid production disruptions but also creates a sustainable competitive advantage in the market.

3.1.2. Organizational Structure

The operational structure of AdventureWorks' Purchasing Department is built upon the dynamic interaction of three core roles: the *Purchasing Manager*, the *Buyer*, and the *Purchasing Assistant***.** While each of these positions functions within a distinct scope of responsibilities, their collaboration forms the backbone of an

efficient and responsive procurement process that aligns with organizational goals in cost control, supply assurance, and inventory stability.

At the strategic level, the *Purchasing Manager* serves as the architect of the company's procurement framework. This role is not confined to approving large-scale purchasing plans but extends to shaping long-term supplier strategies, assessing market conditions, and ensuring procurement activities align with broader financial and operational targets. The Purchasing Manager must constantly evaluate sourcing risks, supplier reliability, and total cost structures to optimize supply continuity while minimizing exposure to market volatility. As such, this role requires not only a macro-level view of supplier ecosystems and contract lifecycles but also timely insights into how purchasing decisions impact inventory flow, working capital, and production schedules. From a solution standpoint, the Purchasing Manager expects a system that enables centralized oversight of procurement activities, facilitates contract governance, and provides alignment between strategic sourcing decisions and actual purchasing execution.

In contrast, the *Buyer* operates at the tactical and executional level of the procurement function. Their primary responsibility lies in translating purchasing plans or inventory replenishment needs into actionable transactions with suppliers. The role demands a high degree of operational responsiveness, as Buyers are directly involved in managing the purchase order lifecycle—from issuance and confirmation to follow-up and exception handling. This includes resolving discrepancies in delivery times, quantities, or item specifications, often under time-sensitive conditions. Furthermore, since Buyers serve as the primary liaison between the organization and its suppliers on a day-to-day basis, they require complete visibility over supplier performance, delivery histories, and open orders. An effective solution must therefore empower Buyers to act swiftly and decisively by providing real-time order status, streamlined supplier communications, and proactive alerts when deviations occur. The ability to adapt rapidly to unexpected changes is critical to maintaining service levels across the supply chain.

Supporting both strategic and operational functions is the *Purchasing Assistant*, whose role centers on maintaining equilibrium between inventory availability and financial efficiency. The Purchasing Assistant ensures that stock levels are sufficient to meet demand without incurring unnecessary holding costs or risking stockouts. This involves continuous monitoring of inventory movements, trend analysis based on consumption patterns, and forecasting future demand to generate accurate replenishment proposals. Given the fast-moving nature of production and sales cycles, this role requires both precision and foresight. The Purchasing Assistant's input is vital in initiating timely purchasing actions and avoiding disruptions caused by misaligned stock levels. Accordingly, any purchasing solution must integrate tightly with inventory data, enabling the Controller to make data-informed proposals and collaborate seamlessly with Buyers on restocking decisions.

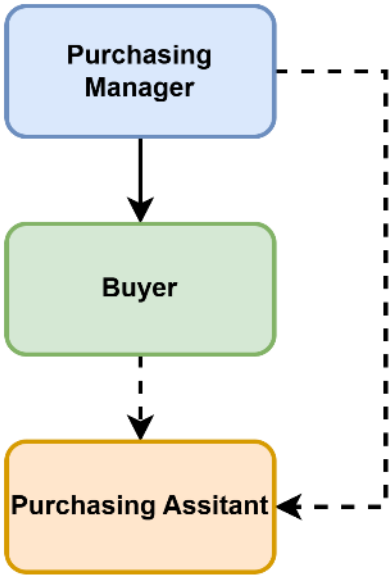


Figure 3.1. Purchasing Department (Source: Authors, 2025)

The interdependence among these three roles illustrates the importance of a cohesive and well-integrated solution that supports both strategic control and operational agility. The Purchasing Manager sets the direction, the Buyer executes in the field, and the Purchasing Assistant ensures the rhythm between consumption and supply is maintained. A robust solution must bridge these perspectives—offering

clarity, consistency, and coordination across all levels of the purchasing hierarchy. In doing so, it not only enhances internal collaboration but also strengthens the organization's ability to respond effectively to external supply chain dynamics.

3.1.3. Purchasing Process

The purchasing process is not merely a material ordering activity but a coordinated system between multiple departments and functions to ensure every purchasing decision stems from actual needs and is controlled for quality, price, timing, and budget.

Step 1: Identify Purchase Need

Purchasing activities originate from actual internal enterprise needs. The Purchasing Assistant directly monitors and analyzes inventory data, periodic material consumption levels, or abnormal signals from the production system. Additionally, specialized departments can submit purchase proposals for specific materials or equipment.

- Input: Actual inventory data, consumption history, production plan, or purchase proposals from relevant departments.
- Output: Preliminary list of purchase needs, serving as the basis for creating a Purchase Requisition.

Step 2: Initiate and Approve Purchase Requisition (PR)

Based on the analysis from the previous step, the Purchasing Assistant or the requester creates a Purchase Requisition. This form must clearly state the purpose, type, quantity of materials, requesting unit, and required delivery time. The request is then sent to the Purchasing Manager for approval according to authorization.

- Input: Clearly defined purchase need, supporting data from the inventory system or production plan.
- Output: Approved Purchase Requisition (validated PR), ready to move to the order creation step.

Step 3: Supplier Selection and Purchase Order (PO) Creation

The Buyer is responsible for receiving the approved PR, then proceeding to search, evaluate, and select suppliers. Selection criteria may include: price, delivery time, payment terms, past reliability, and product technical standards. After identifying a suitable supplier, the Buyer creates a Purchase Order.

- Input: Approved PR, list of potential suppliers, and quotation data.
- Output: Purchase Order (PO) with detailed information: product, quantity, price, delivery deadline, and transaction terms.

Step 4: Purchase Order Approval and Supplier Dispatch

The created order undergoes final approval. The Purchasing Manager reviews the validity of the PO based on budget, price level, transaction volume, and internal regulations. For large orders, multi-level sign-off or leadership approval may be required.

- Input: Purchase Order from Buyer and system approval data.
- Output: Confirmed PO sent to the supplier.

Step 5: Goods Delivery and Receipt

The supplier delivers goods according to the PO. The warehouse department, together with the Buyer, checks and receives the goods. The inspection includes comparing the actual received quantity with the PO, assessing material quality, and recording packaging condition. If the goods meet requirements, a Receiving Report is created.

- Input: Confirmed PO, goods from the supplier, delivery information.
- Output: Goods Receipt Report (Receiving Report), updating the quantity into the inventory system.

Step 6: Invoice Matching and Payment Processing

After delivery is completed, the supplier sends an invoice. The accounting department matches the invoice, PO, and Receiving Report to ensure information consistency (three-way matching process). If there are no discrepancies, payment is processed according to the agreed terms.

- Input: Supplier invoice, confirmed PO, Receiving Report.
- Output: Payment approval, payment order, and recording of supplier liability.

Step 7: Bookkeeping and Data Storage

After payment is completed, the purchasing process continues with post-audit activities to ensure the entire transaction is accurately recorded in the accounting system. Data is also stored for future management, analysis, and internal audit purposes. The accounting department will perform cost recognition entries, update liability accounts, reconcile balances, and allocate purchasing costs to the correct cost objects such as production departments, projects, or overhead costs. Simultaneously, data related to orders, suppliers, invoices, and payment status will be stored in the data management system (data warehouse) to ensure traceability, aggregation, and analysis capabilities.

- Input: Executed payment information, approved invoice, PO data, and Receiving Report.
- Output: Cost and liability recognition in the accounting system, updated purchasing cost reports, stored procurement data for future analysis and auditing.

3.2. Key Performance Indicators (KPIs)

The purchasing department uses the following key performance indicators (KPIs) to monitor and improve procurement efficiency. These indicators offer quantitative insights into the purchasing process, supplier quality, and supply chain responsiveness.

3.2.1. Return Rate

Monitoring Purpose: To evaluate supplier product quality and the effectiveness of quality control during the procurement process.

Definition: Return Rate measures the proportion of products that were ordered but not stocked, due to technical issues, defects, or quality failures.

Formula: Return Rate = (Received Quantity − Stocked Quantity) / Received Quantity

Indicator Interpretation: The lower the Return Rate, the better. A high rate indicates supplier issues or inadequate inspection procedures.

Data Sources:

- Purchasing.PurchaseOrderDetail
- Purchasing.PurchaseOrderHeader

3.2.2. Received Rate

Monitoring Purpose: To assess the reliability of suppliers in delivering the full ordered quantities.

Definition: Received Rate indicates the proportion of products actually received compared to the total quantity ordered.

Formula: Received Rate = Received Quantity / Order Quantity

Indicator Interpretation: A high rate reflects supplier reliability. A low rate may indicate supply issues, partial deliveries, or delivery delays.

Data Sources:

- Purchasing.PurchaseOrderDetail
- Purchasing.PurchaseOrderHeader

3.2.3. Order Quantity

Monitoring Purpose: To track the volume of goods ordered over specific periods, which supports procurement planning and inventory control.

Definition: This KPI sums up the quantity of all products purchased within a defined period (e.g., by day, month, or year).

Formula: OrderQty

Indicator Interpretation: This indicator helps analyze demand trends, forecast future purchasing needs, and avoid overstocking.

Data Sources:

- Purchasing.PurchaseOrderDetail
- Purchasing.PurchaseOrderHeader

3.2.4. Average Lead Time

Monitoring Purpose: To evaluate the responsiveness of the supply chain based on how long it takes to receive goods after ordering.

Definition: Average Lead Time is the average number of days from the order date to the receipt date.

Formula: Average Lead Time

Indicator Interpretation: Shorter lead times suggest a more agile supply chain. Longer durations may lead to stockouts or disrupt production planning.

Data Source: Purchasing.ProductVendor

3.3. Business Intelligence Capabilities for Procurement Decision Support

3.3.1. Strategic Needs for BI in Procurement

The procurement function plays a critical role in ensuring business continuity, especially in manufacturing-oriented enterprises like AdventureWorks. In recent

years, the complexity of sourcing networks, global vendor relationships, and real-time inventory demands have increased pressure on purchasing departments to not only operate efficiently but also make strategic, data-driven decisions. Traditional tools such as spreadsheets or static reports are no longer sufficient to manage the dynamic nature of procurement operations.

Business Intelligence (BI) systems provide a modern solution by offering a centralized and integrated view of procurement data, combining purchase orders, supplier records, lead times, inventory levels, and payment histories. With these capabilities, procurement leaders are empowered to detect inefficiencies, assess supplier performance, monitor compliance, and predict future needs with greater precision. Moreover, the ability to respond quickly to market changes—such as raw material shortages or supplier disruptions—is enhanced by the real-time nature of BI dashboards and alerts. As such, the deployment of BI in procurement is not simply an upgrade in tools; it is a shift toward proactive, insight-led purchasing strategies that align with organizational goals for cost control, agility, and supply chain resilience.

3.3.2. Core Analytical Scenarios

Within the procurement department, BI is applied to a wide range of analytical scenarios that target both operational execution and strategic decision-making. One prominent scenario is the analysis of delivery punctuality. By comparing planned delivery dates with actual receipt timestamps, the system highlights late shipments and quantifies vendor reliability. Such insights are vital in supplier scorecards and contract reviews.

Another scenario involves identifying root causes of purchase order cancellations. While cancellations may stem from many factors—vendor errors, internal misalignment, inventory shifts—the BI system enables categorization of these events and uncovers trends by vendor, product, or seasonality. Procurement teams can thus take corrective action, whether renegotiating contract terms or improving internal coordination.

Additionally, cycle time analysis offers visibility into the full duration of a purchase process—from initial requisition to final payment. BI allows the segmentation of this timeline, helping auditors and managers detect delays in approval chains, invoice processing, or vendor responsiveness. Optimization in these stages can result in both cost savings and enhanced vendor relationships.

Finally, procurement teams can analyze stockouts or understock situations by connecting them to supplier behavior. BI dashboards can attribute late inventory replenishment to specific delayed shipments, allowing planners to adjust reorder strategies or diversify supplier portfolios. When combined with price history analysis across vendors, these scenarios help buyers strike a balance between cost, timeliness, and reliability.

3.3.3. Role-Based Interaction with BI System

The effectiveness of a BI system in procurement depends on its ability to serve the distinct responsibilities of each role within the purchasing department. Since individuals interact with data differently based on their functions and decision-making scope, the system must offer tailored interfaces and tools accordingly.

Purchasing Managers operate at a strategic level. They use BI dashboards to monitor key performance indicators such as supplier reliability, total procurement costs, order cancellation rates, and long-term pricing trends. These insights enable them to evaluate sourcing performance, refine vendor strategies, and make informed decisions aligned with organizational goals.

Buyers, by contrast, are engaged in daily operations. Their use of BI focuses on real-time tasks—tracking open purchase orders, delayed deliveries, cost changes, and low inventory alerts. Features such as reorder recommendations and lead time analysis support their ability to respond quickly and maintain supply chain efficiency.

Purchasing Assistants are tasked with balancing stock levels to avoid both shortages and surpluses. They rely on BI to monitor inventory in real time, analyze historical consumption, and forecast future needs. This data also helps trace inventory

issues back to procurement or supplier performance, allowing for better coordination and corrective actions.

This role-specific design ensures that each team member receives information at the appropriate level of granularity, improving individual decision-making and reinforcing cohesion throughout the procurement workflow.

3.3.4. Types of Intelligence Required

Modern Business Intelligence (BI) systems are no longer limited to traditional reporting functions; they now encompass various forms of analytical reasoning to support complex decision-making. As outlined by TutorialsPoint (2025), BI can be classified according to the analytical depth it offers—primarily across three main categories: *descriptive, predictive,* and *prescriptive intelligence*. In addition, the ability to diagnose causes from patterns in the data is also implicitly embedded within these levels of analysis.

Descriptive intelligence forms the foundation of any BI solution. It is used to represent and summarize historical data, allowing procurement teams to observe what has occurred across their operations. For instance, it enables users to view metrics such as total order volumes, vendor performance over time, and cancellation trends. These insights serve as a factual baseline and are critical for monitoring operational consistency and performance indicators.

Building on this, *predictive intelligence* involves identifying patterns and correlations in historical data to estimate what might happen in the future. In procurement, such analysis supports forecasting demand fluctuations, anticipating delivery delays, and estimating lead time risks. These capabilities allow organizations to make proactive decisions, improve supplier coordination, and better allocate budgets.

Prescriptive intelligence, the most advanced layer, provides actionable recommendations to achieve defined objectives. Unlike predictive analytics—which offers insight into possible outcomes—prescriptive tools suggest optimal decisions

or strategies based on those forecasts. In a procurement setting, this might involve determining the most effective sourcing approach, identifying the best reorder points, or reallocating supply contracts to mitigate risk.

While the source does not explicitly list *diagnostic intelligence* as a standalone category, its essence is reflected in the descriptive layer's capacity to help users "understand why" a particular event occurred. Through drill-downs and cross-dimensional filtering, BI dashboards can facilitate root cause analysis, thereby fulfilling the diagnostic function in practice. (Source: TutorialsPoint)

By integrating these layers of intelligence, BI systems empower procurement professionals not only to monitor past and current operations but also to prepare for future scenarios and take optimized action. The progression from descriptive to prescriptive insight is what makes BI a comprehensive tool for modern procurement strategy.

3.3.5. Detailed Business Requirements

Table 3.1. Business Requirements

Role	Objective	Primary Function	System Requirements	Constraints
Purchasing Manager	Coordinate the entire procurement process to ensure alignment with supply strategy, budget limits, and company	- Approve purchase orders and control costs. - Monitor supplier performance. - Evaluate procurement process compliance.	- System must provide aggregated order reports by allocated budget. - Dashboard for real-time order status tracking. - Budget overrun alerts.	Decisions depend on accuracy and timeliness of inventory data, production forecasts, and allocated budgets.

	business production goals.			
Buyer	Ensure full execution of approved orders, negotiate favorable prices, and track the entire purchase-receipt-payment process with suppliers.	- Create Purchase Orders (PO). - Send PO to the supplier. - Track delivery progress. - Coordinate goods inspection upon receipt.	- System must support direct PO creation from approved requisitions, with editable terms. - Interface for tracking daily delivery status by supplier. - Automatic alerts for overdue POs.	Dependent on supplier response times, market responsiveness, and internal information systems for coordination of inspection, receipt, and payment.
Purchasing Assistant	Ensure optimal inventory levels to prevent production disruptions or excess stock; support purchasing decisions	- Monitor current inventory data. - Compare against alert thresholds to generate purchase proposals. - Coordinate inspection upon goods receipt.	- System must display actual inventory, minimum thresholds, and optimal levels per production plan. - Automatic alerts when stock approaches reorder point. - Allow direct purchase proposal	Accuracy depends on data update frequency; requires close coordination with Buyer to ensure reasonable order lead times.

	based on actual needs.		generation from inventory screen.	

3.4. Data Preparation

3.4.1. Data Sources

In this project, the data source is extracted from the well-known sample database AdventureWorks, provided by Microsoft. AdventureWorks simulates a manufacturing company and offers a rich set of relational data that reflects various business operations such as purchasing, production, human resources, and sales. For the purposes of this project, data related to the Purchasing Department was selected as the foundation for building a specialized data mart. This serves as the basis for constructing a data warehouse, implementing the ETL process, and conducting analytical activities using BI tools.

To build the data mart for the purchasing subsystem, the following tables were selected from the AdventureWorks database due to their relevance and critical roles in the procurement process:

- Purchasing.PurchaseOrderHeader: Records general information about each purchase order.
- Purchasing.PurchaseOrderDetail: Contains line item details for each purchase order.
- Purchasing.Vendor: Contains information about suppliers.
- Purchasing.ProductVendor: Links products to their respective suppliers.
- Purchasing.ShipMethod: Describes shipping methods for the ordered goods.
- Production.Product: Contains data related to products.

- Production.ProductInventory: Tracks inventory quantities of each product.
- Production.ProductCategory: Defines high-level product categories.
- Production.ProductSubCategory: Provides more detailed product classification.
- Production.Location: Identifies warehouse locations related to inventory management.
- HumanResources.Employee: Stores information about employees involved in the purchasing process.
- Person.Person: Provides detailed personal information of employees.

This set of tables supports the construction of the necessary dimension and fact tables for a comprehensive and effective purchasing data mart.

3.4.2. Detailed Description of Data Source

Purchasing.PurchaseOrderHeader: Records general information about each purchase order

Table 3.2. Column of Purchasing.PurchaseOrderHeader

Key	Name	Data type	Null	Description
PK	PurchaseOrderID	int		Primary key. Identity
	RevisionNumber	tinyint		Incremental number to track changes to the purchase order over time. Default: 0
	Status	tinyint		Order current status. 1 = Pending; 2 = Approved; 3 = Rejected; 4 = Complete. Default: 1
	EmployeeID	int		Employee who created the purchase order. Foreign key to Employee.BusinessEntityID. References: HumanResources.Employee
	VendorID	int		Vendor with whom the purchase order is placed. Foreign key to Vendor.BusinessEntityID. References: Purchasing.Vendor
	ShipMethodID	int		Shipping method. Foreign key to ShipMethod.ShipMethodID. References: Purchasing.ShipMethod

	OrderDate	datetime		Purchase order creation date. Default: getdate()
	ShipDate	datetime	x	Estimated shipment date from the vendor.
	SubTotal	money		Purchase order subtotal. Computed as SUM(PurchaseOrderDetail.LineTotal)for the appropriate PurchaseOrderID. Default: 0.00
	TaxAmt	money		Tax amount. Default: 0.00
	Freight	money		Shipping cost. Default: 0.00
	TotalDue	money		Total due to the vendor. Computed as Subtotal + TaxAmt + Freight. Computed: isnull(([SubTotal]+[TaxAmt])+[Freight],(0))
	ModifiedDate	datetime		Date and time the record was last updated. Default: getdate()

Table 3.3. Relation of Purchasing.PurchaseOrderHeader

Foreign table		**Primary table**	**Description**

Purchasing.PurchaseOrderHeader		HumanResources.Employee	**Join:** Purchasing.PurchaseOrderHeader.EmployeeID = HumanResources.Employee.BusinessEntityID **Name:** FK_PurchaseOrderHeader_Employee_EmployeeID Foreign key constraint referencing Employee.EmployeeID
Purchasing.PurchaseOrderHeader		Purchasing.ShipMethod	**Join:** Purchasing.PurchaseOrderHeader.ShipMethodID = Purchasing.ShipMethod.ShipMethodID **Name:** FK_PurchaseOrderHeader_ShipMethod_ShipMethodID Foreign key constraint referencing ShipMethod.ShipMethodID.
Purchasing.PurchaseOrderHeader		Purchasing.Vendor	**Join:** Purchasing.PurchaseOrderHeader.VendorID = Purchasing.Vendor.BusinessEntityID **Name**: FK_PurchaseOrderHeader_Vendor_VendorID Foreign key constraint referencing Vendor.VendorID.
Purchasing.PurchaseOrderDetail		Purchasing.PurchaseOrderHeader	**Join:** Purchasing.PurchaseOrderDetail.PurchaseOrderID = Purchasing.PurchaseOrderHeader.PurchaseOrderID **Name:** FK_PurchaseOrderDetail_PurchaseOrderHeader_PurchaseOrderID

			Foreign key constraint referencing PurchaseOrderHeader.PurchaseOrderID.

Purchasing.PurchaseOrderDetail: Contains line item details for each purchase order

Table 3.4. Column of Purchasing.PurchaseOrderDetail

Key	Name	Data type	Null	Description
PK	PurchaseOrderID	int		Primary key. Foreign key to PurchaseOrderHeader.PurchaseOrderID. References: Purchasing.PurchaseOrderHeader
PK	PurchaseOrderDetailID	int		Primary key. One line number per purchased product. Identity
	DueDate	datetime		Date the product is expected to be received.
	OrderQty	smallint		Quantity ordered.
	ProductID	int		Product identification number. Foreign key to Product.ProductID. References: Production.Product

	UnitPrice	money		Vendor's selling price of a single product.
	LineTotal	money		Per product subtotal. Computed as OrderQty * UnitPrice. Computed: isnull([OrderQty]*[UnitPrice],(0.00))
	ReceivedQty	decimal(8, 2)		Quantity actually received from the vendor.
	RejectedQty	decimal(8, 2)		Quantity rejected during inspection.
	StockedQty	decimal(9, 2)		Quantity accepted into inventory. Computed as ReceivedQty - RejectedQty. Computed: isnull([ReceivedQty]-[RejectedQty],(0.00))
	ModifiedDate	datetime		Date and time the record was last updated. Default: getdate()

Table 3.5 Relation of Purchasing.PurchaseOrderDetail

Foreign table		Primary table	Description

Purchasing.PurchaseOrder Detail	⟝	Production.Product	Purchasing.PurchaseOrderDetail.ProductID = Production.Product.ProductID Name: FK_PurchaseOrderDetail_Product_ProductID Foreign key constraint referencing Product.ProductID.
Purchasing.PurchaseOrder Detai	⟝	Purchasing.Purchase OrderHeader	Purchasing.PurchaseOrderDetail.PurchaseOrderID = Purchasing.PurchaseOrderHeader.PurchaseOrderID Name: FK_PurchaseOrderDetail_PurchaseOrderHeader_PurchaseOrderID Foreign key constraint referencing PurchaseOrderHeader.PurchaseOrderID.

Purchasing.Vendor: Contains information about suppliers

Table 3.6. Column of Purchasing.Vendor

Key	Name	Data type	Null	Description
PK	BusinessEntityID	int		Primary key for Vendor records. Foreign key to

				BusinessEntity.BusinessEntityID. References: Person.BusinessEntity
AK	AccountNumber	nvarchar(15)		Vendor account (identification) number.
	Name	nvarchar(50)		Company name.
	CreditRating	tinyint		1 = Superior, 2 = Excellent, 3 = Above average, 4 = Average, 5 = Below average
	PreferredVendor Status	bit		0 = Do not use if another vendor is available. 1 = Preferred over other vendors supplying the same product. Default: 1
	ActiveFlag	bit		0 = Vendor no longer used. 1 = Vendor is actively used. Default: 1
	PurchasingWeb ServiceURL	nvarchar(1024)	x	Vendor URL.
	ModifiedDate	datetime		Date and time the record was last updated. Default: getdate()

Table 3.7. Relation of Purchasing.Vendor

Foreign table		Primary table	Description
Purchasing.Vendor		Person.BusinessEntity	Purchasing.Vendor.BusinessEntityID = Person.BusinessEntity.BusinessEntityID Name: FK_Vendor_BusinessEntity_BusinessEntityID Foreign key constraint referencing BusinessEntity.BusinessEntityID
Purchasing.ProductVendor		Purchasing.Vendor	Purchasing.ProductVendor.BusinessEntityID = Purchasing.Vendor.BusinessEntityID Name: FK_ProductVendor_Vendor_BusinessEntityID Foreign key constraint referencing Vendor.BusinessEntityID.
Purchasing.PurchaseOrderHeader		Purchasing.Vendor	Purchasing.PurchaseOrderHeader.VendorID = Purchasing.Vendor.BusinessEntityID Name: FK_PurchaseOrderHeader_Vendor_VendorID Foreign key constraint referencing Vendor.VendorID.

Purchasing.ProductVendor: Links products to their respective suppliers

Table 3.8. Column of Purchasing.ProductVendor

Key	Name	Data type	Null	Description
PK	ProductID	int		Primary key. Foreign key to Product.ProductID. References: Production.Product
PK	BusinessEntityID	int		Primary key. Foreign key to Vendor.BusinessEntityID. References: Purchasing.Vendor
	AverageLeadTime	int		The average span of time (in days) between placing an order with the vendor and receiving the purchased product.
	StandardPrice	money		The vendor's usual selling price.
	LastReceiptCost	money	x	The selling price when last purchased.
	LastReceiptDate	datetime	x	Date the product was last received by the vendor.
	MinOrderQty	int		The maximum quantity that should be ordered.

	MaxOrderQty	int		The minimum quantity that should be ordered.
	OnOrderQty	int	x	The quantity currently on order.
	UnitMeasureCode	nchar(3)		The product's unit of measure. References: Production.UnitMeasure
	ModifiedDate	datetime		Date and time the record was last updated. Default: getdate()

Table 3.9. Relation of Purchasing.ProductVendor

Foreign table		Primary table	Description
Purchasing.ProductVendor		Production.Product	Purchasing.ProductVendor.ProductID = Production.Product.ProductID Name: FK_ProductVendor_Product_ProductID Foreign key constraint referencing Product.ProductID.
Purchasing.ProductVendor		Production.UnitMeasure	Purchasing.ProductVendor.UnitMeasureCode = Production.UnitMeasure.UnitMeasureCode

			Name: FK_ProductVendor_UnitMeasure_UnitMeasureCode Foreign key constraint referencing UnitMeasure.UnitMeasureCode.
Purchasing.ProductVendor		Purchasing.Vendor	Purchasing.ProductVendor.BusinessEntityID = Purchasing.Vendor.BusinessEntityID Name: FK_ProductVendor_Vendor_BusinessEntityID Foreign key constraint referencing Vendor.BusinessEntityID.

Purchasing.ShipMethod: Describes shipping methods for the ordered goods

Table 3.10. Column of Purchasing.ShipMethod

Key	Name	Data type	Null	Description
PK	ShipMethodID	int		Primary key for ShipMethod records. Identity
AK	Name	nvarchar(50)		Shipping company name.
	ShipBase	money		Minimum shipping charge. Default: 0.00

	ShipRate	money		Shipping charge per pound. Default: 0.00
AK	rowguid	uniqueidentifier		ROWGUIDCOL number uniquely identifying the record. Used to support a merge replication sample. Default: newid()
	ModifiedDate	datetime		Date and time the record was last updated. Default: getdate()

Table 3.11. Relation of Purchasing.ShipMethod

Foreign table		Primary table	Description
Purchasing.PurchaseOrderHeader	⟝	Purchasing.ShipMethod	Purchasing.PurchaseOrderHeader.ShipMethodID = Purchasing.ShipMethod.ShipMethodID Name: FK_PurchaseOrderHeader_ShipMethod_ShipMethodID Foreign key constraint referencing ShipMethod.ShipMethodID.
Sales.SalesOrderHeader	⟝	Purchasing.ShipMethod	Sales.SalesOrderHeader.ShipMethodID = Purchasing.ShipMethod.ShipMethodID

			Name: FK_SalesOrderHeader_ShipMethod_ShipMethodID Foreign key constraint referencing ShipMethod.ShipMethodID.

Production.Product: Contains data related to products

Table 3.12. Column of Production.Product

Key	Name	Data type	Null	Description
PK	ProductID	int		Primary key for Product records. Identity
AK	Name	nvarchar(50)		Name of the product.
AK	ProductNumber	nvarchar(25)		Unique product identification number.
	MakeFlag	bit		0 = Product is purchased, 1 = Product is manufactured in-house. Default: 1
	FinishedGoodsFlag	bit		0 = Product is not a salable item. 1 = Product is salable. Default: 1
	Color	nvarchar(15)	x	Product color.

	SafetyStockLevel	smallint		Minimum inventory quantity.
	ReorderPoint	smallint		Inventory level that triggers a purchase order or work order.
	StandardCost	money		Standard cost of the product.
	ListPrice	money		Selling price.
	Size	nvarchar(5)	x	Product size.
	SizeUnitMeasureCode	nchar(3)	x	Unit of measure for Size column. References: Production.UnitMeasure
	WeightUnitMeasureCode	nchar(3)	x	Unit of measure for Weight column. References: Production.UnitMeasure
	Weight	decimal(8, 2)	x	Product weight.
	DaysToManufacture	int		Number of days required to manufacture the product.
	ProductLine	nchar(2)	x	R = Road, M = Mountain, T = Touring, S = Standard

	Class	nchar(2)	x	H = High, M = Medium, L = Low
	Style	nchar(2)	x	W = Womens, M = Mens, U = Universal
	ProductSubcategoryID	int	x	Product is a member of this product subcategory. Foreign key to ProductSubCategory.ProductSubCategoryID. References: Production.ProductSubcategory
	ProductModelID	int	x	Product is a member of this product model. Foreign key to ProductModel.ProductModelID. References: Production.ProductModel
	SellStartDate	datetime		Date the product was available for sale.
	SellEndDate	datetime	x	Date the product was no longer available for sale.
	DiscontinuedDate	datetime	x	Date the product was discontinued.
AK	rowguid	uniqueidentifier		ROWGUIDCOL number uniquely identifying the record. Used to support a merge replication sample. Default: newid()

	ModifiedDate	datetime		Date and time the record was last updated. Default: getdate()

Table 3.13. Relation of Production.Product

Foreign table		Primary table	Description
Production.ProductInventory	⟝	Production.Product	Production.ProductInventory.ProductID = Production.Product.ProductID Name: FK_ProductInventory_Product_ProductID Foreign key constraint referencing Product.ProductID.
Purchasing.ProductVendor	⟝	Production.Product	Purchasing.ProductVendor.ProductID = Production.Product.ProductID Name: FK_ProductVendor_Product_ProductID Foreign key constraint referencing Product.ProductID.
Purchasing.PurchaseOrderDetail	⟝	Production.Product	Purchasing.PurchaseOrderDetail.ProductID = Production.Product.ProductID

			Name: FK_PurchaseOrderDetail_Product_ProductID Foreign key constraint referencing Product.ProductID.

Production.ProductInventory: Tracks inventory quantities of each product

Table 3.14. Column of Production.ProductInventory

Key	Name	Data type	Null	Description
PK	ProductID	int		Product identification number. Foreign key to Product.ProductID. References: Production.Product
PK	LocationID	smallint		Inventory location identification number. Foreign key to Location.LocationID. References: Production.Location
	Shelf	nvarchar(10)		Storage compartment within an inventory location.
	Bin	tinyint		Storage container on a shelf in an inventory location.
	Quantity	smallint		Quantity of products in the inventory location. Default: 0

	rowguid	uniqueidentifier		ROWGUIDCOL number uniquely identifying the record. Used to support a merge replication sample. Default: newid()
	ModifiedDate	datetime		Date and time the record was last updated. Default: getdate()

Table 3.15. Relation of Production.ProductInventory

Foreign table		Primary table	Description
Production.ProductInventory		Production.Location	Production.ProductInventory.LocationID = Production.Location.LocationID Name: FK_ProductInventory_Location_LocationID Foreign key constraint referencing Location.LocationID.
Production.ProductInventory		Production.Product	Production.ProductInventory.ProductID = Production.Product.ProductID Name: FK_ProductInventory_Product_ProductID Foreign key constraint referencing Product.ProductID.

Production.ProductCategory: Defines high-level product categories

Table 3.16. Column of Production.ProductCategory

Key	Name	Data type	Null	Description
PK	ProductCategoryID	int		Primary key for ProductCategory records. Identity
AK	Name	nvarchar(50)		Category description. References: Production.ProductModelIllustration
AK	rowguid	uniqueidentifier		ROWGUIDCOL number uniquely identifying the record. Used to support a merge replication sample. Default: newid()
	ModifiedDate	datetime		Date and time the record was last updated. Default: getdate()

Table 3.17. Relation of Production.ProductCategory

Foreign table		Primary table	Description
Production.ProductCat		Production.Product	Production.ProductCategory.Name =

egory		ModelIllustration	Production.ProductModelIllustration.ProductModelID Production.ProductCategory.Name = Production.ProductModelIllustration.IllustrationID Name: fk_ProductModelIllustration_ProductCategory
Production.ProductSubcategory		Production.ProductCategory	Production.ProductSubcategory.ProductCategoryID = Production.ProductCategory.ProductCategoryID Name: FK_ProductSubcategory_ProductCategory_ProductCategoryID Foreign key constraint referencing ProductCategory.ProductCategoryID.

Production.ProductSubCategory: Provides more detailed product classification

Table 3.18. Column of Production.ProductSubCategory

Key	Name	Data type	Null	Description
PK	ProductSubcategoryID	int		Primary key for ProductSubcategory records. Identity

	ProductCategoryID	int		Product category identification number. Foreign key to ProductCategory.ProductCategoryID. References: Production.ProductCategory
AK	Name	nvarchar(50)		Subcategory description.
AK	rowguid	uniqueidentifier		ROWGUIDCOL number uniquely identifying the record. Used to support a merge replication sample. Default: newid()
	ModifiedDate	datetime		Date and time the record was last updated. Default: getdate()

Table 3.19. Relation of Production.ProductSubCategory

Foreign table		Primary table	Description
Production.ProductSubcategory		Production.ProductCategory	Production.ProductSubcategory.ProductCategoryID = Production.ProductCategory.ProductCategoryID Name: FK_ProductSubcategory_ProductCategory_ProductCategoryID

			Foreign key constraint referencing ProductCategory.ProductCategoryID.
Production.Product		Production.ProductSubcategory	Production.Product.ProductSubcategoryID = Production.ProductSubcategory.ProductSubcategoryID Name: FK_Product_ProductSubcategory_ProductSubcategoryID Foreign key constraint referencing ProductSubcategory.ProductSubcategoryID.

Production.Location: Identifies warehouse locations related to inventory management

Table 3.20. Column of Production.Location

Key	Name	Data type	Null	Description
PK	LocationID	smallint		Primary key for Location records. Identity
AK	Name	nvarchar(50)		Location description.
	CostRate	smallmoney		Standard hourly cost of the manufacturing location. Default: 0.00

	Availability	decimal(8, 2)		Work capacity (in hours) of the manufacturing location. Default: 0.00
	ModifiedDate	datetime		Date and time the record was last updated. Default: getdate()

Table 3.21. Relation of Production.Location

Foreign table		Primary table	Description
Production.ProductInventory	>—	Production.Location	Production.ProductInventory.LocationID = Production.Location.LocationID Name: FK_ProductInventory_Location_LocationID Foreign key constraint referencing Location.LocationID.
Production.WorkOrderRouting	>—	Production.Location	Production.WorkOrderRouting.LocationID = Production.Location.LocationID Name: FK_WorkOrderRouting_Location_LocationID Foreign key constraint referencing Location.LocationID.

HumanResources.Employee: Stores information about employees involved in the purchasing process

Table 3.22. Column of HumanResources.Employee

Key	Name	Data type	Null	Description
PK	BusinessEntityID	int		Primary key for Employee records. Foreign key to BusinessEntity.BusinessEntityID. References: Person.Person
AK	NationalIDNumber	nvarchar(15)		Unique national identification number such as a social security number.
AK	LoginID	nvarchar(256)		Network login. Test2
	OrganizationNode	hierarchyid	x	Where the employee is located in corporate hierarchy.
	OrganizationLevel	smallint	x	The depth of the employee in the corporate hierarchy. Computed: [OrganizationNode].[GetLevel]()
	JobTitle	nvarchar(50)		Work title such as Buyer or Sales Representative.
PK (user-	BirthDate	date		Date of birth.

defined)				
	MaritalStatus	nchar(1)		M = Married, S = Single
	Gender	nchar(1)		M = Male, F = Female
	HireDate	date		Employee hired on this date.
	SalariedFlag	bit		Job classification. 0 = Hourly, not exempt from collective bargaining. 1 = Salaried, exempt from collective bargaining. Default: 1
	VacationHours	smallint		Number of available vacation hours. Default: 0
	SickLeaveHours	smallint		Number of available sick leave hours. Default: 0
	CurrentFlag	bit		0 = Inactive, 1 = Active. Default: 1
AK	rowguid	uniqueidentifier		ROWGUIDCOL number uniquely identifying the record. Used to support a merge replication sample. Default: newid()
	ModifiedDate	datetime		Date and time the record was last updated. Default: getdate()

Table 3.23. Relation of HumanResources.Employee

Foreign table		Primary table	Description
HumanResources.Employee	>—	Person.Person	HumanResources.Employee.BusinessEntityID = Person.Person.BusinessEntityID Name: FK_Employee_Person_BusinessEntityID Foreign key constraint referencing Person.BusinessEntityID.
Purchasing.PurchaseOrderHeader	>—	HumanResources.Employee	Purchasing.PurchaseOrderHeader.EmployeeID = HumanResources.Employee.BusinessEntityID Name: FK_PurchaseOrderHeader_Employee_EmployeeID Foreign key constraint referencing Employee.EmployeeID.

Person.Person: Provides detailed personal information of employees

Table 3.24. Column of Person.Person

Key	Name	Data type	Null	Description
PK	BusinessEntityID	int		Primary key for Person records. References: Person.BusinessEntity
	PersonType	nchar(2)		Primary type of person: SC = Store Contact, IN = Individual (retail) customer, SP = Sales person, EM = Employee (non-sales), VC = Vendor contact, GC = General contact
	NameStyle	bit		0 = The data in FirstName and LastName are stored in western style (first name, last name) order. 1 = Eastern style (last name, first name) order. Default: 0
	Title	nvarchar(8)	x	A courtesy title. For example, Mr. or Ms.
	FirstName	nvarchar(50)		First name of the person.
	MiddleName	nvarchar(50)	x	Middle name or middle initial of the person.
	LastName	nvarchar(50)		Last name of the person.
	Suffix	nvarchar(10)	x	Surname suffix. For example, Sr. or Jr.

	EmailPromotion	int		0 = Contact does not wish to receive e-mail promotions, 1 = Contact does wish to receive e-mail promotions from AdventureWorks, 2 = Contact does wish to receive e-mail promotions from AdventureWorks and selected partners. Default: 0
	AdditionalContactInfo	xml	x	Additional contact information about the person stored in xml format.
	Demographics	xml	x	Personal information such as hobbies, and income collected from online shoppers. Used for sales analysis.
AK	rowguid	uniqueidentifier		ROWGUIDCOL number uniquely identifying the record. Used to support a merge replication sample. Default: newid()
	ModifiedDate	datetime		Date and time the record was last updated. Default: getdate()

Table 3.25. Relation of Person.Person

Foreign table		Primary table	Description
Person.Person		Person.BusinessEntity	Person.Person.BusinessEntityID = Person.BusinessEntity.BusinessEntityID Name: FK_Person_BusinessEntity_BusinessEntityID Foreign key constraint referencing BusinessEntity.BusinessEntityID.
Person.BusinessEntityContact		Person.Person	Person.BusinessEntityContact.PersonID = Person.Person.BusinessEntityID Name: FK_BusinessEntityContact_Person_PersonID Foreign key constraint referencing Person.BusinessEntityID.
Sales.Customer		Person.Person	Sales.Customer.PersonID = Person.Person.BusinessEntityID Name: FK_Customer_Person_PersonID Foreign key constraint referencing Person.BusinessEntityID.
Person.EmailAddress		Person.Person	Person.EmailAddress.BusinessEntityID = **Person.Person**.BusinessEntityID **Name**: FK_EmailAddress_Person_BusinessEntityID Foreign key constraint referencing Person.BusinessEntityID.
HumanResources.Employee		Person.Person	HumanResources.Employee.BusinessEntityID =

			Person.Person.BusinessEntityID Name: FK_Employee_Person_BusinessEntityID Foreign key constraint referencing Person.BusinessEntityID.
Person.Password		Person.Person	Person.Password.BusinessEntityID = Person.Person.BusinessEntityID Name: FK_Password_Person_BusinessEntityID Foreign key constraint referencing Person.BusinessEntityID.
Sales.PersonCreditCard		Person.Person	Sales.PersonCreditCard.BusinessEntityID = Person.Person.BusinessEntityID Name: FK_PersonCreditCard_Person_BusinessEntityID Foreign key constraint referencing Person.BusinessEntityID.
Person.PersonPhone		Person.Person	Person.PersonPhone.BusinessEntityID = Person.Person.BusinessEntityID Name: FK_PersonPhone_Person_BusinessEntityID Foreign key constraint referencing Person.BusinessEntityID.

3.4.3. Data Understanding

Microsoft and its community have released various versions of the Adventure Works Cycles (AWC) databases, ranging from 2005 to 2017, for educational and proof-of-concept purposes (Lim and Mafas, 2020). The Adventure Works suite includes two distinct Microsoft SQL Server databases. The first is an Online Transaction Processing (OLTP) database, designed with a comprehensive structure and rich content to efficiently manage daily transactional operations. The second is a data warehouse, tailored for Online Analytical Processing (OLAP) and data mining, enabling complex analytical queries in a robust environment.

To build the Data Mart, we used Python to perform exploratory data analysis on the database tables. Data was extracted from the AdventureWorks2019 database, focusing on three interrelated subsystems: Purchasing, Production, and Human Resources. Each table was then exported as a separate CSV file. The dataset used to construct the Purchasing Data Mart covers the period from 2011 to 2014.

3.4.4. Data Analysis

Data analysis is a crucial process for uncovering meaningful insights that support timely decision-making. In this study, descriptive analytics were employed to conduct a comprehensive examination of the data. Descriptive Analytics, also referred to as Exploratory Data Analysis (EDA), is a fundamental approach in big data analytics, essential for extracting valuable insights and forming well-grounded conclusions. This method allows for an in-depth understanding of the dataset, helping to reveal patterns, trends, and anomalies that inform strategic decisions.

Product Analysis

An overview description of the columns in the product table, including the number of non-null values in each column and the data type of each column.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 504 entries, 0 to 503
Data columns (total 25 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   ProductID             504 non-null    int64
 1   Name                  504 non-null    object
 2   ProductNumber         504 non-null    object
 3   MakeFlag              504 non-null    int64
 4   FinishedGoodsFlag     504 non-null    int64
 5   Color                 256 non-null    object
 6   SafetyStockLevel      504 non-null    int64
 7   ReorderPoint          504 non-null    int64
 8   StandardCost          504 non-null    object
 9   ListPrice             504 non-null    object
 10  Size                  211 non-null    object
 11  SizeUnitMeasureCode   176 non-null    object
 12  WeightUnitMeasureCode 205 non-null    object
 13  Weight                205 non-null    float64
 14  DaysToManufacture     504 non-null    int64
 15  ProductLine           278 non-null    object
 16  Class                 247 non-null    object
 17  Style                 211 non-null    object
 18  ProductSubcategoryID  295 non-null    float64
 19  ProductModelID        295 non-null    float64
 20  SellStartDate         504 non-null    object
 21  SellEndDate           98 non-null     object
 22  DiscontinuedDate      0 non-null      float64
 23  rowguid               504 non-null    object
 24  ModifiedDate          504 non-null    object
```

Figure 3.2. Product table Description (Source: Authors, 2025)

An overview description of the columns in the product table, including the number of non-null values in each column and the data type of each column.

```
       ProductID    MakeFlag  FinishedGoodsFlag  SafetyStockLevel  \
count  504.000000  504.000000         504.000000        504.000000
mean   673.039683    0.474206           0.585317        535.150794
std    229.373142    0.499830           0.493157        374.112954
min      1.000000    0.000000           0.000000          4.000000
25%    447.750000    0.000000           0.000000        100.000000
50%    747.500000    0.000000           1.000000        500.000000
75%    873.250000    1.000000           1.000000       1000.000000
max    999.000000    1.000000           1.000000       1000.000000

       ReorderPoint       Weight  DaysToManufacture  ProductSubcategoryID  \
count    504.000000   205.000000         504.000000            295.000000
mean     401.363095    74.069220           1.103175             12.294915
std      280.584715   182.166588           1.492616              9.860135
min        3.000000     2.120000           0.000000              1.000000
25%       75.000000     2.880000           0.000000              2.000000
50%      375.000000    17.900000           1.000000             12.000000
75%      750.000000    27.350000           1.000000             17.000000
max      750.000000  1050.000000           4.000000             37.000000
```

Figure 3.3. Product Statistical tables (Source: Authors, 2025)

From the above data of the product table, we can visualize it to reveal important information. The image displays a series of visualizations summarizing key product attributes such as manufacturing status, stock levels, costs, prices, and weights. It includes bar charts and a box plot to highlight data distribution, frequency, and outliers. These visuals provide a quick understanding of the dataset's structure and help identify potential issues like data skewness or missing values.

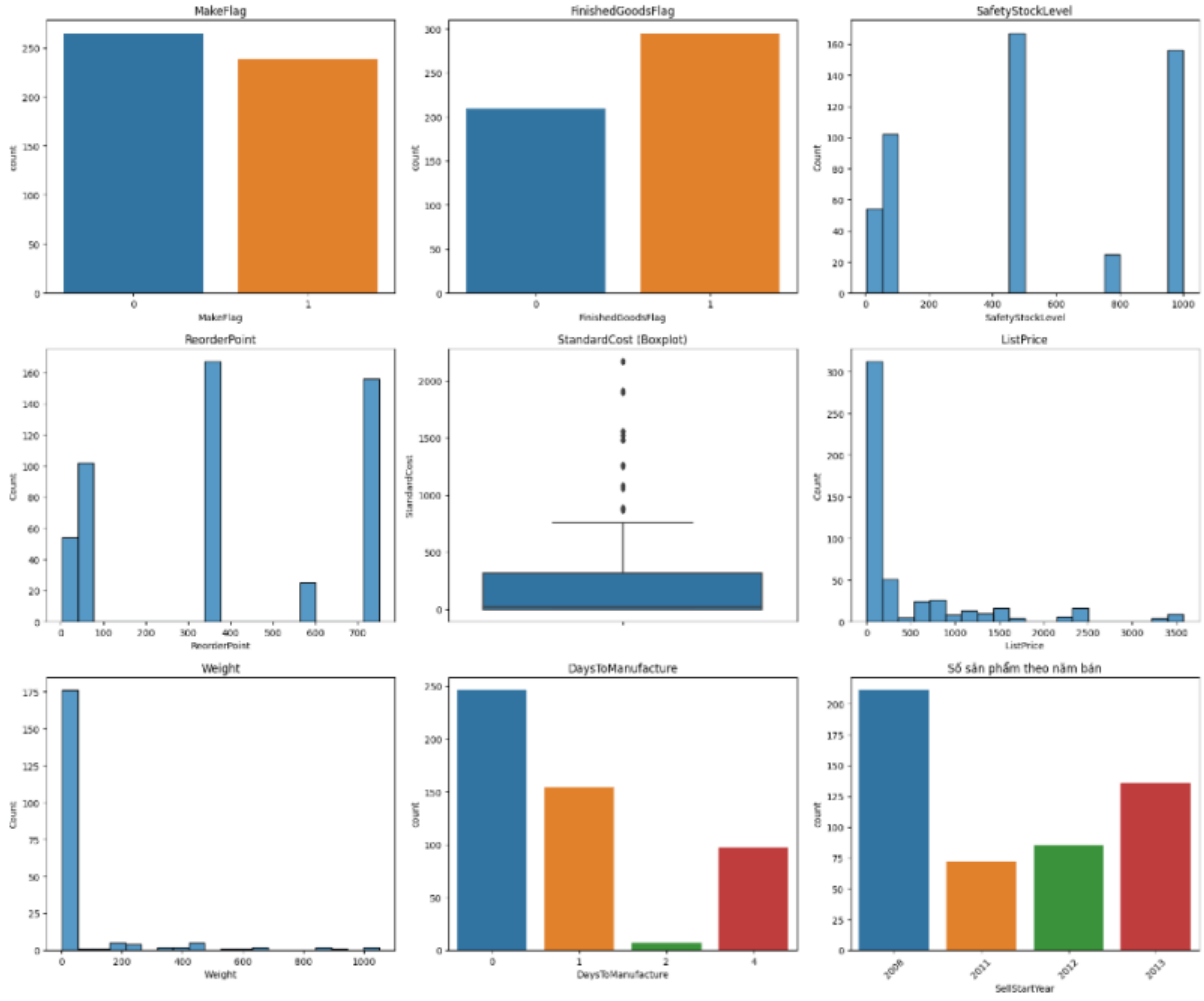


Figure 3.4. Summary charts of product attributes and manufacturing statistics. (Source: Authors, 2025)

Product Inventory Table

```
RangeIndex: 1069 entries, 0 to 1068
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   ProductID     1069 non-null   int64
 1   LocationID    1069 non-null   int64
 2   Shelf         779 non-null    object
 3   Bin           1069 non-null   int64
 4   Quantity      1069 non-null   int64
 5   rowguid       1069 non-null   object
 6   ModifiedDate  1069 non-null   object
dtypes: int64(4), object(3)
```

Figure 3.5. Product Inventory table Description (Source: Authors, 2025)

The Product Inventory table is extracted from the source table Production.ProductInventory, consists of 1069 rows and 7 columns, with specific details outlined as depicted in the figure above.

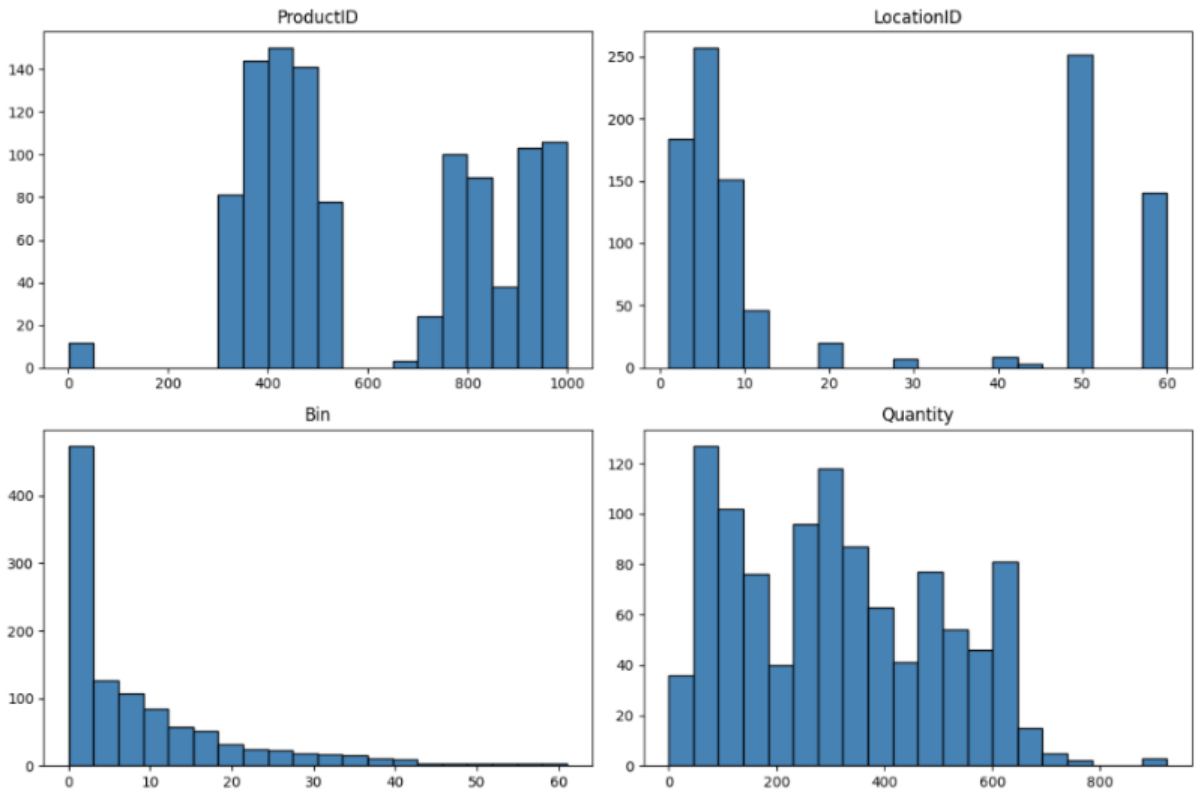


Figure 3.6. The data distribution of the ProductInventory table (Source: Authors, 2025)

Product Subcategory

```
RangeIndex: 37 entries, 0 to 36
Data columns (total 5 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   ProductSubcategoryID  37 non-null     int64
 1   ProductCategoryID     37 non-null     int64
 2   Name                  37 non-null     object
 3   rowguid               37 non-null     object
 4   ModifiedDate          37 non-null     object
dtypes: int64(2), object(3)
```

Figure 3.7. Product Subcategory table Description (Source: Authors, 2025)

The Product Subcategory table, sourced from Production.ProductSubcategory, contains 5 columns, as detailed in figure above.

Product Category

```
RangeIndex: 4 entries, 0 to 3
Data columns (total 4 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   ProductCategoryID  4 non-null     int64
 1   Name               4 non-null     object
 2   rowguid            4 non-null     object
 3   ModifiedDate       4 non-null     object
dtypes: int64(1), object(3)
```

Figure 3.8. Product Category table Description (Source: Authors, 2025)

The Product Category table, sourced from ProductCategory, consists of 4 columns, with specific details shown in figure above.

Product Vendor

```
RangeIndex: 460 entries, 0 to 459
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   ProductID        460 non-null    int64
 1   BusinessEntityID  460 non-null    int64
 2   AverageLeadTime  460 non-null    int64
 3   StandardPrice    460 non-null    object
 4   LastReceiptCost  460 non-null    object
 5   LastReceiptDate  460 non-null    object
 6   MinOrderQty      460 non-null    int64
 7   MaxOrderQty      460 non-null    int64
 8   OnOrderQty       155 non-null    float64
 9   UnitMeasureCode  460 non-null    object
 10  ModifiedDate     460 non-null    object
dtypes: float64(1), int64(5), object(5)
```

Figure 3.9. ProductVendor table Description (Source: Authors, 2025)

The Purchasing.ProductVendor table is a crucial component of the procurement system, extracted from the Purchasing.ProductVendor source table. It provides detailed information on the vendors supplying products to the organization. This table consists of 460 rows and 11 columns, each representing specific data points essential for managing vendor relationships and procurement processes.

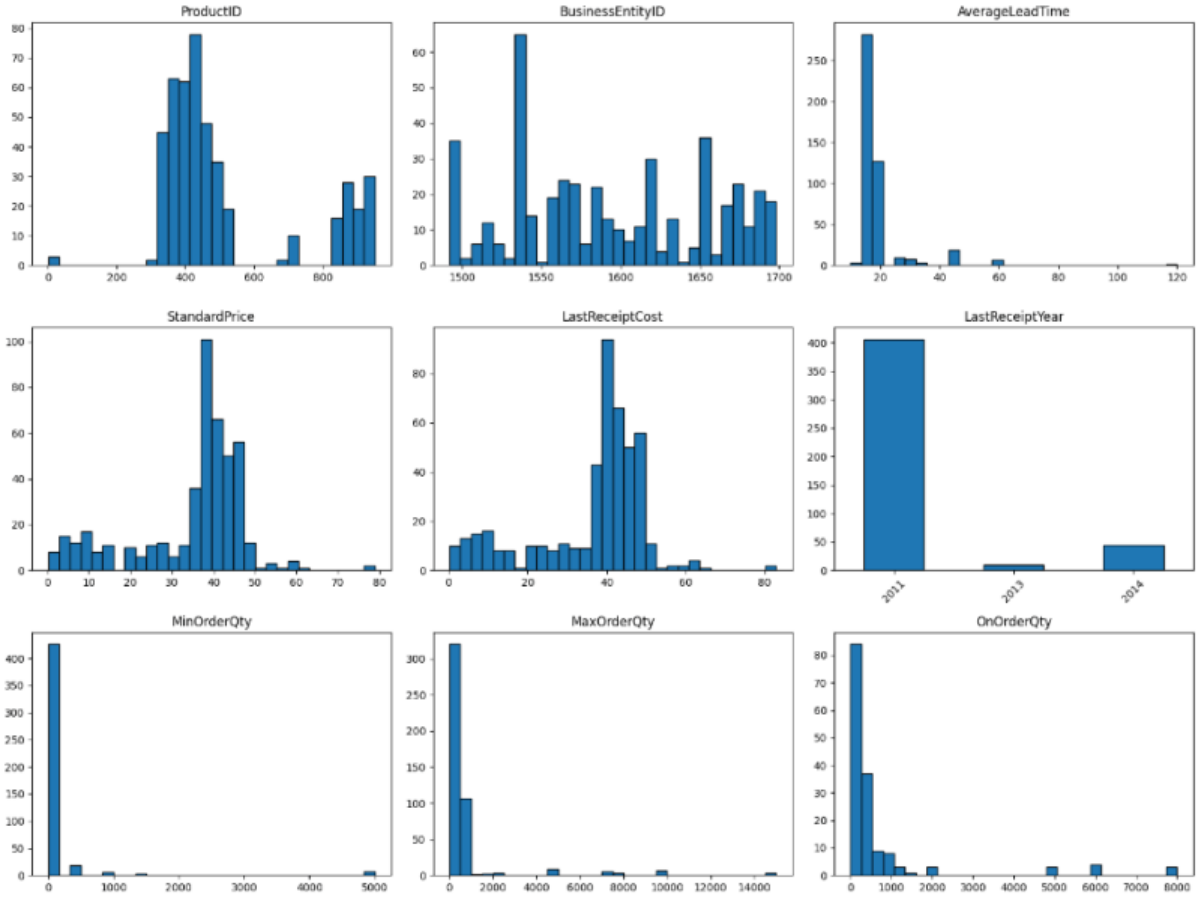


Figure 3.10. The data distribution of the UnitMeasure table (Source: Authors, 2025)

Location table

```
RangeIndex: 14 entries, 0 to 13
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   LocationID    14 non-null     int64
 1   Name          14 non-null     object
 2   CostRate      14 non-null     object
 3   Availability  14 non-null     float64
 4   ModifiedDate  14 non-null     object
dtypes: float64(1), int64(1), object(3)
```

Figure 3.11. Location table Description (Source: Authors, 2025)

The Location table, derived from Production.Location, comprises 14 rows and 5 columns, as illustrated in Figure above.

Purchase Order Header Table

```
RangeIndex: 4012 entries, 0 to 4011
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   PurchaseOrderID  4012 non-null   int64
 1   RevisionNumber   4012 non-null   int64
 2   Status           4012 non-null   int64
 3   EmployeeID       4012 non-null   int64
 4   VendorID         4012 non-null   int64
 5   ShipMethodID     4012 non-null   int64
 6   OrderDate        4012 non-null   object
 7   ShipDate         4012 non-null   object
 8   SubTotal         4012 non-null   object
 9   TaxAmt           4012 non-null   object
 10  Freight          4012 non-null   object
 11  TotalDue         4012 non-null   object
 12  ModifiedDate     4012 non-null   object
dtypes: int64(6), object(7)
```

Figure 3.12. PurchaseOrderHeader table Description (Source: Author 2025)

The Purchase Order Header table, sourced from PurchaseOrderHeader, consists of 4012 rows and 13 columns, with specific details shown in Figure below.

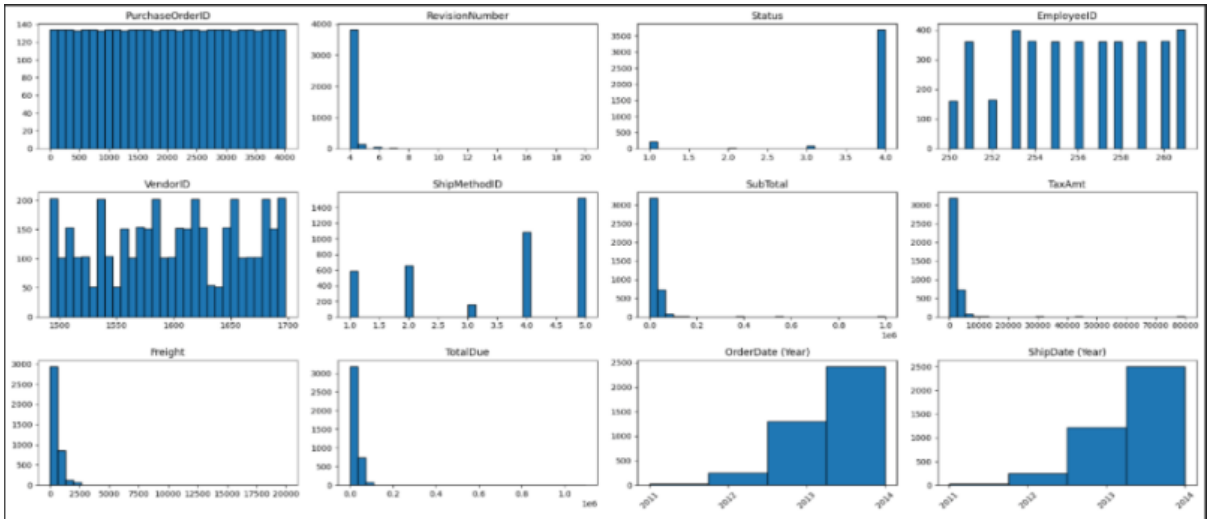


Figure 3.13. The data distribution of the ProductOrderHeader table (Source: Author 2025)

Purchase order detail

```
RangeIndex: 8845 entries, 0 to 8844
Data columns (total 11 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   PurchaseOrderID       8845 non-null   int64
 1   PurchaseOrderDetailID 8845 non-null   int64
 2   DueDate               8845 non-null   object
 3   OrderQty              8845 non-null   int64
 4   ProductID             8845 non-null   int64
 5   UnitPrice             8845 non-null   object
 6   LineTotal             8845 non-null   object
 7   ReceivedQty           8845 non-null   float64
 8   RejectedQty           8845 non-null   float64
 9   StockedQty            8845 non-null   float64
 10  ModifiedDate          8845 non-null   object
dtypes: float64(3), int64(4), object(4)
```

Figure 3.14. PurchaseOderDetail table Description (Source: Author 2025)

The Purchase Order Detail table, sourced from PurchaseOrderDetails, includes 8845 rows and 11 columns. Specific details are illustrated in the figure above.

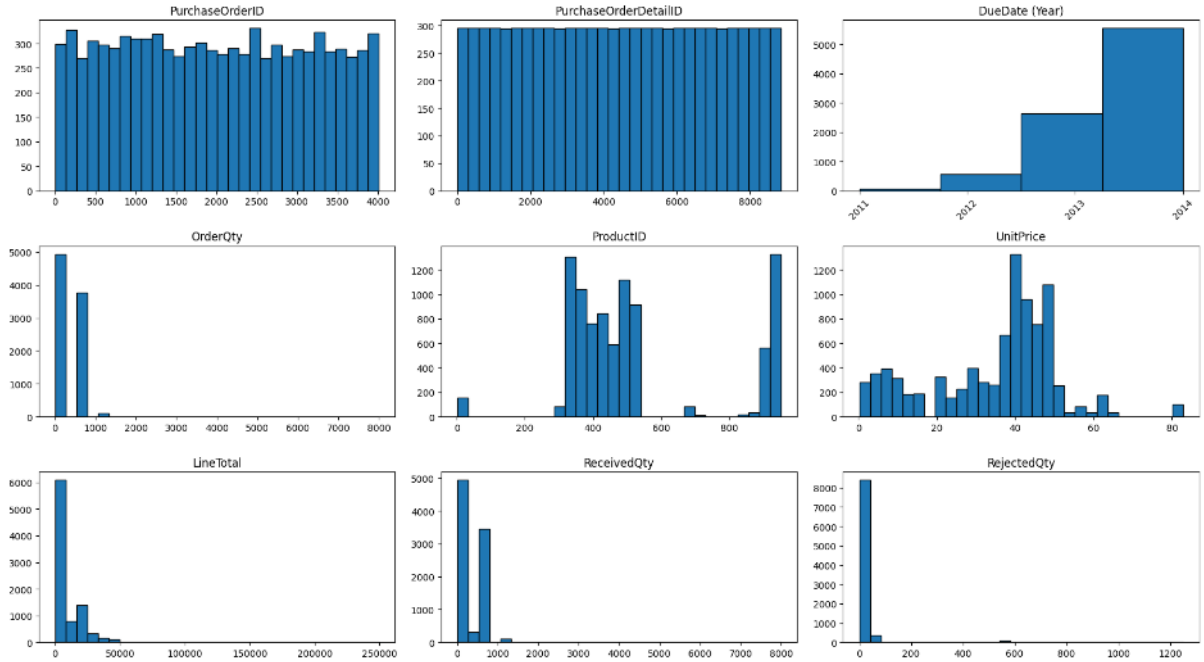
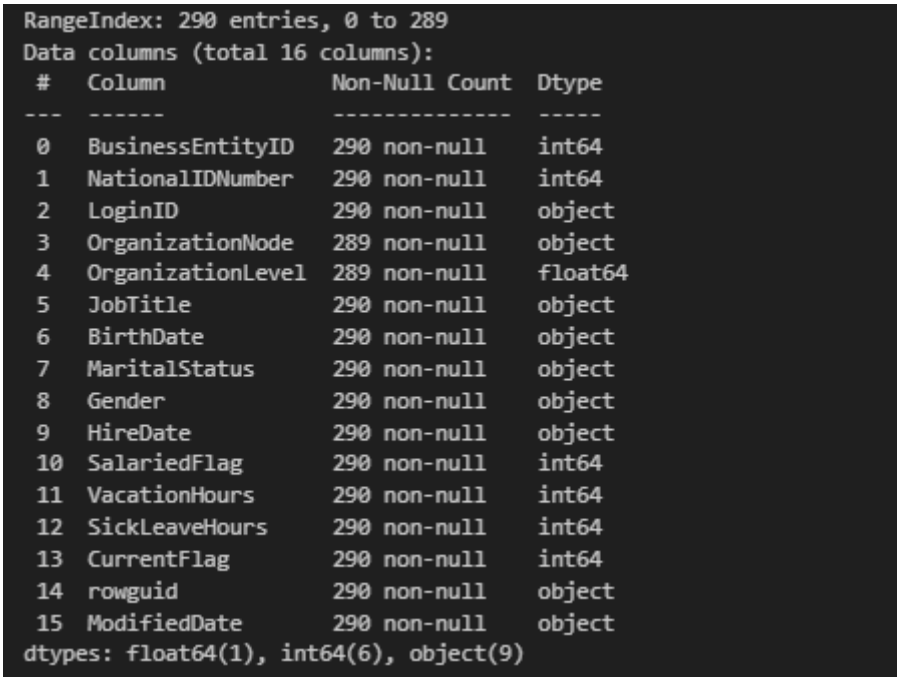


Figure 3.15. The data distribution of the ProductOrderDetail table (Source: Authors, 2025)

Employee table



```
RangeIndex: 290 entries, 0 to 289
Data columns (total 16 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   BusinessEntityID   290 non-null    int64
 1   NationalIDNumber   290 non-null    int64
 2   LoginID            290 non-null    object
 3   OrganizationNode   289 non-null    object
 4   OrganizationLevel  289 non-null    float64
 5   JobTitle           290 non-null    object
 6   BirthDate          290 non-null    object
 7   MaritalStatus      290 non-null    object
 8   Gender             290 non-null    object
 9   HireDate           290 non-null    object
 10  SalariedFlag       290 non-null    int64
 11  VacationHours      290 non-null    int64
 12  SickLeaveHours     290 non-null    int64
 13  CurrentFlag        290 non-null    int64
 14  rowguid            290 non-null    object
 15  ModifiedDate       290 non-null    object
dtypes: float64(1), int64(6), object(9)
```

Figure 3.16. Employee table Description (Source: Authors, 2025)

The Employee table, derived from the Employee source table, comprises 290 rows and 16 columns.

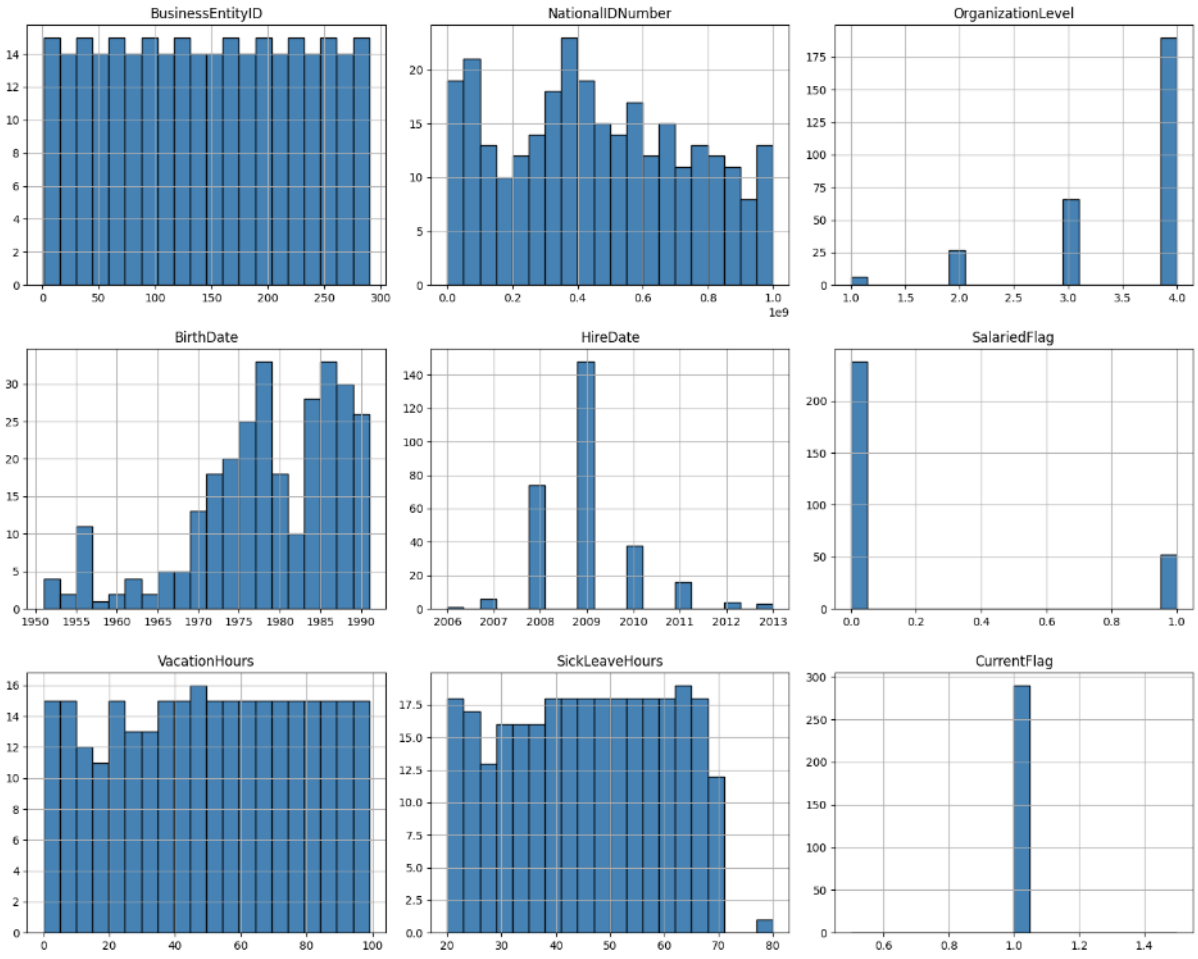


Figure 3.17. The data distribution of the Employee table (Source: Authors, 2025)

CHAPTER 4: BUILDING DATA WAREHOUSE AND INTEGRATING DATA

4.1. Design Data Warehouse

4.1.1. Bus Matrix

Table 4.1. Bus matrix

Business Processes	**Dimensions**				
	Product	Employee	Vendor	Ship Method	Time
Purchase Order Management	X	X	X	X	X
Product Tracking	X				X
Supplier Performance Evaluation	X		X		X
Inventory Monitoring	X		X		X
Shipping Efficiency Analysis				X	X

The bus matrix of the Purchasing Department at AdventureWorks connects key business processes with relevant dimension tables, enabling consistent and efficient data analysis. The use of conformed dimensions within a unified data warehouse architecture

supports accurate reporting and enhances decision-making capabilities. Below is a summary of business processes related to purchasing activities represented in the matrix:

- Purchase Order Management: Supports detailed order tracking. Tracked by: time, vendor, product, employee, shipping method.
- Product Tracking: Ensures accurate delivery and effective inventory control. Tracked by: time, product.
- Supplier Performance Evaluation: Helps assess and maintain supplier quality. Tracked by: time, vendor, product.
- Inventory Monitoring: Ensures optimal stock levels and effective warehouse control. Tracked by: time, vendor, product.
- Shipping Efficiency Analysis: Aims to optimize shipping strategies and minimize delivery costs. Tracked by: time, shipping method.

4.1.2. Master Data and Transaction Data

In modern data warehouse architecture, classifying and thoroughly understanding different types of data is a prerequisite for building an effective analytical system. Two core categories of data in this process are master data and transaction data—each with its distinct roles and structures that complement one another to provide a comprehensive view of organizational operations.

Master data refers to the fundamental information that defines long-term business entities such as products, suppliers, customers, or employees. These entities typically do not change frequently and are reused across various business processes. Therefore, master data forms the foundation of dimension tables in the data warehouse—where attributes are organized to support multidimensional queries and analysis.

Table 4.2. Master data

Master data	Description
Product	Contains detailed information about the company's products, including attributes such as name, category, and cost.
Vendor	Includes essential information about suppliers, such as name, address.
Employee	Information about the staff responsible for placing orders, including employee ID, full name, and position.
Warehouse	Details about the warehouse, such as product quantity, storage compartments, and storage containers.
Ship Method	Contains information on delivery methods, including method name and associated costs.

In contrast, transaction data is dynamic and is generated whenever a business activity occurs, such as placing an order, making a purchase, processing a payment, or executing a shipment. This data captures the specifics of each action: who performed it, what was done, when, where, and at what value. It serves as the primary source for building fact tables, which aggregate quantitative metrics for reporting, KPIs, and trend modeling.

Table 4.3. Transaction data

Description	Description

Date	The date when the purchase order is created or completed.
Purchasing.PurchaseOrderHeader	Provides an overview of each purchase order, including supplier information, order date, status, and related details.
Purchasing.PurchaseOrderDetail	Contains detailed information about individual products associated with specific purchase orders.
Production.ProductInventory	Contains inventory information of products at a specific point in time, including quantity, shelf, bin, and warehouse location.

Master data provides the "context" needed to interpret and analyze transaction data. While transaction data delivers concrete evidence of business events that have taken place, master data ensures consistency and meaning. Both types of data are essential for developing a complete and reliable data warehouse system, capable of effectively supporting Business Intelligence (BI) tools and enabling data-driven decision-making.

4.1.3. Data Model

4.1.3.1 Data Model overview

To meet the analytical requirements of the Purchasing Department at AdventureWorks, this project adopts a Galaxy Schema—also known as a Fact Constellation Schema—as the foundational structure of the data warehouse. This schema is specifically designed to support complex analytical environments where

multiple processes, such as purchasing and inventory management, need to be examined in parallel through shared analytical dimensions.

The Galaxy Schema consists of multiple fact tables—each capturing distinct types of transactional or quantitative data—connected to a set of common dimension tables that provide business context. In our implementation, the model centers around two primary fact tables:

- FactPurchaseOrders, which records transactional data such as order quantities, unit prices, received and rejected quantities, vendor associations, and purchase dates.

- FactInventory, which contains periodic snapshots of inventory levels, including available quantity, storage location, shelf ID, and product references.

These fact tables are linked to shared dimensions such as DimProduct, DimVendor, DimEmployee, DimTime, and DimShipMethod. This structure enables analysts to conduct integrated queries across business domains. For example, users can examine vendor performance by comparing purchase volume and rejection rates, while also evaluating inventory availability for the same set of products and time periods.

The Galaxy Schema offers several key advantages in this context:

- Integrated Multi-Process Analysis: By modeling different operational areas through dedicated fact tables, the schema allows for both isolated and cross-functional analysis. Analysts can study purchasing trends independently, or relate them to inventory behavior across time and categories.
- Data Consistency Across Facts: Shared dimension tables ensure consistent definitions and classifications—such as product categories or time hierarchies—across all analytical scenarios, avoiding redundancy and promoting a single version of truth.

- Historical Context and Data Evolution: Dimensions such as DimVendor and DimProduct are implemented with Slowly Changing Dimension (SCD) techniques to preserve historical changes over time. This enables accurate reporting and trend analysis, even as business entities evolve.
- Scalability and Maintainability: The schema's modular structure supports easy addition of new fact tables or dimensions. For example, incorporating a sales fact table in the future would only require linking to existing dimensions, minimizing redesign efforts.
- Compatibility with Modern BI Tools: Platforms like Power BI and SQL Server Analysis Services (SSAS) are highly compatible with Galaxy Schema architectures. The shared-dimension approach simplifies OLAP cube development, enhances reusability of measures and hierarchies, and supports efficient drill-down and roll-up analysis.

In practice, this data model empowers users to explore complex business questions with flexibility and speed. Whether comparing vendor delivery performance over time, tracking inventory stockouts by product category, or correlating purchase orders with warehouse availability, the Galaxy Schema provides the structure and depth needed for advanced business intelligence in the procurement domain.

Table 4.4. Description of Galaxy data warehouse model

No.	Relationship	Type	Description
1	DimVendor – FactPurchaseOrders	1-n	One vendor can have one or more records in FactPurchaseOrders; each record in FactPurchaseOrders references only one vendor.
2	DimEmployee - FactPurchaseOrders	1-n	One employee can have one or more records in FactPurchaseOrders; each record references only one employee.

3	DimProduct - FactPurchaseOrders	1-n	One product can have one or more records in FactPurchaseOrders; each record references only one product.
4	DimTime - FactPurchaseOrders	1-n	One datetime can have one or more records in FactPurchaseOrders; each record references only one datetime.
5	DimShipMethod - FactPurchaseOrders	1-n	One shipping method can have one or more records in FactPurchaseOrders; each record references only one shipping method.
7	DimProduct – FactInventory	1-n	One product may be present in multiple inventory snapshots; each record in FactInventory references one product.

4.1.3.2. Fact and Dimension tables

Table 4.5. Data Warehouse Table Descriptions

No.	Table Name	Description
1	DimVendor	Information about vendors collaborating with the company.
2	DimEmployee	Employee information.
3	DimProduct	Product information, including inventory quantity and purchasing status.
4	DimTime	Time information.
5	DimShipMethod	Shipping method information.
6	FactPurchaseOrders	Aggregated data about vendors, products, employees, and purchasing activities over time.
7	FactInventory	Stores periodic inventory data, including stock levels and associated product IDs.

Table Structure Details

DimVendor

Data Source: Purchasing.Vendor, Purchasing.ProductVendor

Table 4.6. DimVendor table

Column Name	Data Type	SCD Type	Source	Description
VendorKey (PK)	int	0	-	Primary key.
BusinessEntityID (NK)	int	0	Purchasing.Vendor	Business entity identifier.
VendorName	nvarchar(50)	1	Purchasing.Vendor.Name	Vendor name.
CreditRating	tinyint	2	Purchasing.Vendor	Credit rating.
PreferredVendorStatus	bit	2	Purchasing.Vendor	Preferred vendor status.
ActiveFlag	bit	0	Purchasing.Vendor	Active flag.
AverageLeadTime	int	2	Purchasing.ProductVendor	Average delivery time.
StandardPrice	money	2	Purchasing.ProductVendor	Standard price.
LastReceiptCost	money	2	Purchasing.ProductVendor	Last receipt cost.
StartDate	datetime	0	-	Start date.
EndDate	datetime	0	-	End date.

DimProduct

DataSource: Production.Product, Production.ProductCategory, Production.ProductSubCategory

Table 4.7. DimProduct table

Column Name	Data Type	SCD Type	Source	Description
ProductKey (PK)	int	0	-	Primary key.
ProductID (NK)	int	0	Production.Product	Product ID.
ProductName	nvarchar(50)	1	Production.Product.Name	Product name.
ProductNumber	nvarchar(25)	1	Production.Product	Product number.
MakeFlag	bit	1	Production.Product	Manufacturing flag.
FinishedGoodFlag	bit	1	Production.Product	Finished goods flag.
SafetyStockLevel	smallint	2	Production.Product	Safety stock level.
ReorderPoint	smallint	2	Production.Product	Reorder point.
ProductCategoryName	nvarchar(50)	1	Production.ProductCategory.Name	Product category name.

ProductSubCategoryName	nvarchar(50)	1	Production.ProductSubCategory.Name	Product subcategory name.
StartDate	datetime	0	-	Start date.
EndDate	datetime	0	-	End date.

DimEmployee

Data Source: HumanResources.Employee, Person.Person

Table 4.8. DimEmployee table

Column Name	Data Type	SCD Type	Source	Description
EmployeeKey (PK)	int	0	-	Primary key.
BusinessEntityID (NK)	int	0	HumanResources.Employee	Business entity identifier.
Gender	nchar(1)	1	HumanResources.Employee	Gender.
BirthDate	date	1	HumanResources.Employee	Date of birth.
HireDate	date	1	HumanResources.Employee	Hire date.

JobTitle	nvarchar(50)	2	HumanResources.Employee	Job title.
SalariedFlag	bit	1	HumanResources.Employee	Salaried flag.
ActiveFlag	bit	0	HumanResources.Employee.CurrentFlag	Active flag.
FirstName	nvarchar(50)	1	Person.Person	First name.
MiddleName	nvarchar(50)	1	Person.Person	Middle name.
LastName	nvarchar(50)	1	Person.Person	Last name.
StartDate	datetime	0	-	Start date.
EndDate	datetime	0	-	End date.

DimTime

Data Source: Generated dimension

Table 4.9. DimTime table

Column Name	Data Type	SCD Type	Description
DateKey (PK)	datetime	0	Primary key.

DateFull	date	0	Full date.
TheDay	int	0	Day of the month.
TheDayName	nvarchar(30)	0	Day name (e.g., Monday).
TheWeek	int	0	Week of the year.
TheISOWeek	int	0	ISO week.
TheDayOfWeek	int	0	Day of week (numeric).
TheMonth	int	0	Month.
TheMonthName	nvarchar(30)	0	Month name.
TheQuarter	int	0	Quarter.
TheYear	int	0	Year.
TheFirstOfMonth	date	0	First day of the month.
TheLastOfYear	date	0	Last day of the year.

DimShipMethod

Data Source: Purchasing.ShipMethod

Table 4.10. DimShipMethod table

Column Name	Data Type	SCD Type	Source	Description
ShipMethodKey (PK)	int	0	-	Primary key.
ShipMethodID (NK)	int	0	Purchasing.ShipMethod	Shipping method ID.
Name	nvarchar(50)	1	Purchasing.ShipMethod	Method name.
ShipBase	money	0	Purchasing.ShipMethod	Base shipping cost.
ShipRate	money	0	Purchasing.ShipMethod	Shipping rate.

FactPurchaseOrders

Data Source: Purchasing.PurchaseOrderHeader, Purchasing.PurchaseOrderDetail

Table 4.11. FactPurchaseOrders table

Column	Data Type	SCD Type	Source	Description
PurchaseOrderID	int	0	-	
PurchaseOrderDetailID	int	0	-	

RevisionNumber	tinyint	0	Purchasing.PurchaseOrderHeader	Revision number.
Status	tinyint	0	Purchasing.PurchaseOrderHeader	Status.
OrderDate	datetime	0	Purchasing.PurchaseOrderHeader	Order date.
ShipDate	datetime	0	Purchasing.PurchaseOrderHeader	Ship date.
Freight	money	0	Purchasing.PurchaseOrderHeader	Freight
ProductKey (FK)	int	0	-	Foreign key to DimProduct.
EmployeeKey (FK)	int	0	-	Foreign key to DimEmployee.
VendorKey (FK)	int	0	-	Foreign key to DimVendor.
ShipMethodKey (FK)	int	0	-	Foreign key to DimShipMethod.

OrderQty	smallint	0	Purchasing.PurchaseOrderDetail	Order quantity.
UnitPrice	money	0	Purchasing.PurchaseOrderDetail	Unit price.
LineTotal	money	0	Purchasing.PurchaseOrderDetail	Line total amount.
ReceivedQty	decimal(8,2)	0	Purchasing.PurchaseOrderDetail	Received quantity.
RejectedQty	c(8,2)	0	Purchasing.PurchaseOrderDetail	Rejected quantity.
StockedQty	decimal(9,2)	0	Purchasing.PurchaseOrderDetail	Stocked quantity.

FactInventory

Data Source: Production.ProductInventory, Production.Location, DimTime

Table 4.12. FactInventory table

Column	**Data Type**	**SCD Type**	**Source**	**Description**
ProductID (FK)	int	0	Production.ProductInventory	Product ID.

LocationID (FK)	smallint	0	Production.ProductInventory	Location ID.
ProductKey	int	0	-	Foreign key to DimProduct.
Shelf	nvarchar(10)	0	Production.ProductInventory	Shelf location.
Bin	tinyint	0	Production.ProductInventory	Bin number.
Quantity	tinyint	0	Production.ProductInventory	Quantity.
Availability	decimal(8,2)	0	Production.Location	Availability status.
InventoryDate	datetime	0	DimTime	
ProductID (FK)	int	0	Production.ProductInventory	Product ID.

4.1.4. Data Warehouse building

After selecting the tables to use in the dataset and filtering the necessary columns, the team decided to build a data warehouse as shown below.

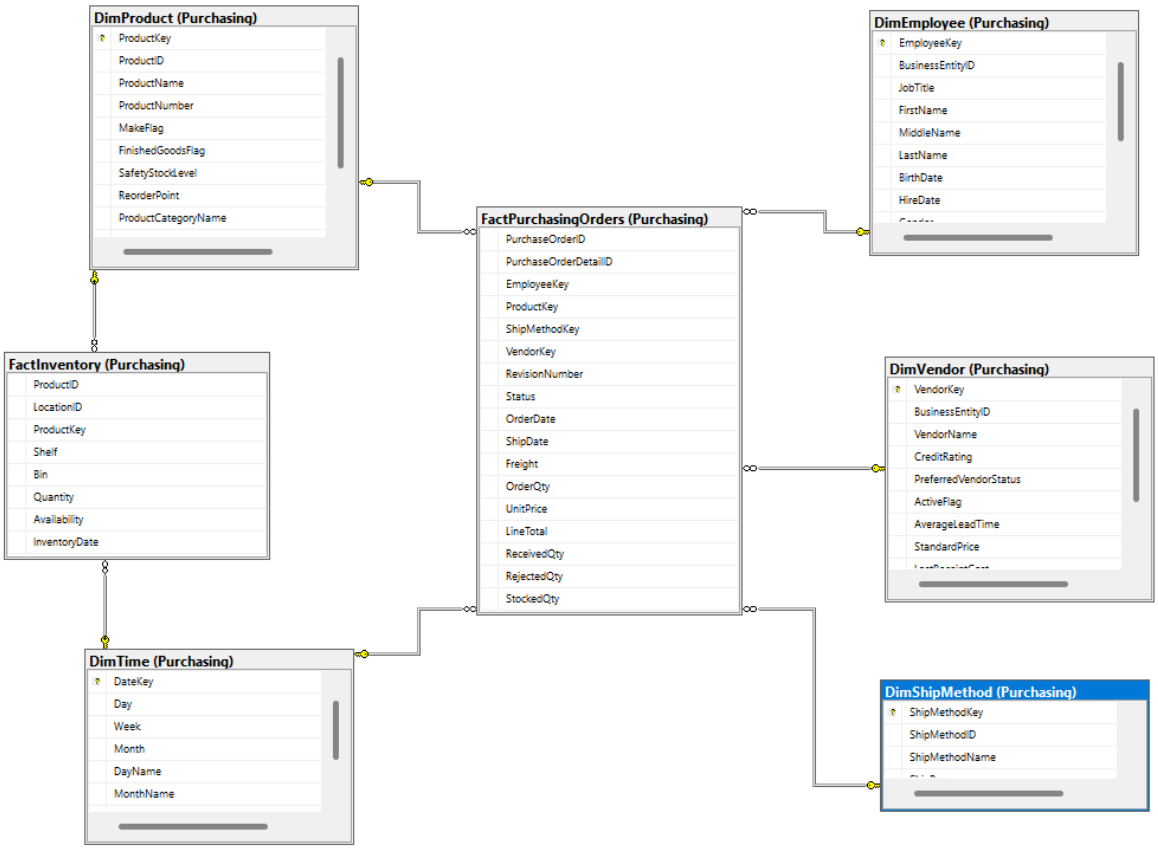


Figure 4.1. Purchasing data warehouse model

4.2. ETL Process

4.2.1. Rationale for Selecting the ETL Strategy and Data Integration Framework

During the development of the Data Mart for the Purchasing subsystem, the project team selected the ETL (Extract, Transform, Load) approach as the core data integration strategy. This decision was driven by the need to process dispersed data originating from multiple systems and tables, requiring a robust yet flexible mechanism to consolidate and standardize the data architecture.

ETL proved to be a suitable choice in this context due to its ability to manage and streamline the flow of data from source systems to the data warehouse, ensuring that the data is accurate, reliable, and ready for querying, analysis, and long-term monitoring.

The key reasons for adopting the ETL strategy are as follows:

- *Integration from heterogeneous systems:* The Purchasing subsystem receives data from various independent sources, each with distinct formats and structures. ETL facilitates the consolidation of this information by standardizing and unifying it into a common schema, thus simplifying downstream analytics and reporting.

- *Efficient handling of complex and high-volume data:* Purchasing data typically involves large datasets and intricate relationships among entities such as orders, products, and suppliers. ETL offers a structured and sequential process that ensures smooth data processing and management in such complex environments.

- *Support for historical tracking and versioning:* As purchasing data evolves—such as price adjustments, product updates, or changes in supplier information—it is essential to retain historical versions of the data. ETL enables the design of processing flows that preserve and track historical records, which is critical for time-series analysis and traceability.

- *Automation and scalability:* One of ETL's main advantages lies in its ability to automate data processing tasks, reducing reliance on manual operations and minimizing human error. Furthermore, ETL pipelines can be adapted and scaled with relative ease when there is a need to adjust data structures, incorporate new business logic, or expand the system.

Data Integration Framework Using ETL

Extract

In the initial phase, data is extracted from key tables within the Purchasing subsystem, including:

- HumanResources.Employee

- Purchasing.Vendor

- Person.Person
- Purchasing.ProductVendor
- Production.Product
- Production.ProductInventory
- Production.ProductCategory
- Production.ProductSubCategory
- Production.Location
- Purchasing.ShipMethod
- PurchaseOrderDetails
- PurchaseOrderHeader

The extraction focuses on collecting essential information such as purchase order details, product data, supplier records, and ordered items. Once retrieved, the data is reorganized into structured tabular formats suitable for further SQL-based transformation steps within the ETL pipeline.

Transform

After extraction, the next step involves refining and standardizing the data to ensure consistency and usability. This phase includes tasks such as reformatting values, cleansing inconsistencies, filtering out invalid entries, and integrating records from multiple sources. Special attention is given to time-sensitive data, such as historical orders or price fluctuations, which are processed in a way that retains previous states while accurately representing the current status. This approach supports not only real-time reporting but also long-term trend analysis and forecasting.

Load

The final phase involves loading the processed data into the data warehouse. This step incorporates validation procedures to ensure completeness, uniqueness, and

accuracy. Tools such as SSIS are often used to automate this loading process while simultaneously tracking performance metrics, logging operations, and flagging anomalies. As a result, the warehouse remains consistently updated and ready to support analytical queries and business intelligence tasks.

4.2.2 Detailed Operational Process of the System

The end-to-end integration model is a structured approach for integrating data in Business Intelligence (BI) projects, specifically within the ETL (Extract, Transform, Load) process.

This technique ensures more effective integration and maintenance of data in BI systems. It clarifies the detailed ETL process, explains how changing data is handled, ensures data integrity and accuracy, illustrates the complexity of the system, and provides a comprehensive view of the BI system's backend.

The operation of the end-to-end integration model is as follows:

- First, data is extracted from the OLTP database by focusing on specified data fields.
- After extraction, the data is searched in the data warehouse to determine appropriate storage and transformation methods.
- If the data is not found in the warehouse due to ID conflicts, it will be inserted directly into the warehouse.
- When duplicate data is encountered, it is classified according to the Slowly Changing Dimension (SCD) type.
- Type 1 SCD data will be updated, meaning the existing record is overwritten with new data without preserving historical changes.
- Type 2 SCD data will be inserted, typically by creating a new record for the changed data and updating the EndDate attribute of the old record to reflect its historical status.

Modified data is compared with the current data in the warehouse and updated if necessary.

- When new information is added to a table, the EndDate attribute in the warehouse is also updated, and new records are added to the database.

→ This entire technique - including extraction, transformation, and loading, with specific handling for new, updated, and historical data - aims to ensure more effective data integration and maintenance in the BI system

The research team applied the SCD method to handle changing data across all dimension tables. The specific SCD type applied depends on the inherent characteristics of each attribute in the table.

The use of different SCD types (0, 1, and 2) for individual attributes within the same dimension table demonstrates a deep understanding of historical data management. For example, JobTitle is a Type 2 SCD, while FirstName is a Type 1 SCD in the DimEmployee table. This detailed approach ensures that significant historical changes that affect business analysis are retained for auditing and trend analysis. Meanwhile, less important or descriptive changes only update the current record to optimize storage and performance. This careful balance optimizes storage and query performance while preserving necessary historical accuracy for robust BI reporting.

4.2.3. Practical execution of SSIS process

Logging in the ETL Process

During the development of ETL packages using SSIS, the team incorporated both system-defined and user-defined variables to support monitoring, logging, and error-handling throughout the data loading process. This approach is considered a best practice in enterprise environments, enabling better traceability, error tracking, and diagnostics.

The figure below presents a list of variables configured within the package responsible for processing the DimEmployee table:

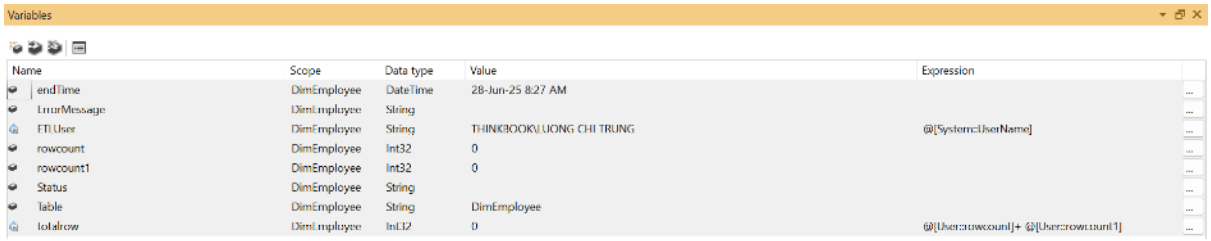


Figure 4.2. System and Custom Variables in the ETL Package – DimEmployee

Table 4.13. Description of Variables Used in the ETL Package

Variable	Data Type	Description
endTime	DateTime	Marks the end time of the package execution; used for logging or performance comparison.
ErrorMessage	String	Captures error messages if any issues occur during execution (for enhanced error handling).
ETLUser	String	Automatically retrieves the user executing the package via the expression @[System::UserName].
rowcount	Int32	Stores the number of rows successfully loaded in the NewOutput and Historical Attribute Inserts Output branches.
rowcount1	Int32	Stores the number of rows successfully loaded in the Changing Attribute Updates Output branch
totalrow	Int32	Total number of rows loaded into the dimension
Status	String	Indicates execution status (e.g., Success, Failed); helps with overall job verification.

Table	String	Records the target table name—in this case, DimEmployee.

Utilizing these variables allows the ETL process to be more transparent, easier to troubleshoot, and more scalable. It is particularly useful when implementing centralized logging mechanisms such as writing execution metadata to a table like ETL_Load_Log.

Standard Control Flow for Dimension Table ETL

The ETL process for each dimension table in the system follows a standardized control flow pattern. This design ensures better monitoring and logging after each execution, making the overall process more reliable and maintainable.

The following figure illustrates the typical control flow structure used for dimension tables such as DimEmployee, DimVendor, DimProduct, and others:

Figure 4.3. Standard Control Flow for ETL of Dimension Tables

The control flow consists of four main steps:

- *Set Status:* Before the main data flow is executed, the variable Status is initialized—commonly with a value like "Running"—to mark the beginning of the process. This setup helps differentiate the state of execution and supports log tracking or error handling in case of failures during the ETL process.

- *Execute Data Flow (e.g., DimEmployee):* This is the core of the package, responsible for extracting data from OLTP sources, applying necessary transformations (e.g., Derived Column, Lookup, Data Conversion), and loading the processed data into the corresponding dimension table in the Data Warehouse.

- *Set End Time:* Upon completion, the system assigns the current timestamp (GETDATE()) to the variable endTime. This allows accurate measurement of execution time for each individual table—especially useful for performance analysis or historical monitoring.

- *Write Log to ETL_Load_Log Table:* In the final step, a logging task is triggered to store metadata such as the table name, executing user (ETLUser), status (Status), number of rows processed (rowcount), end time (endTime), and any error messages if applicable. This information is saved to a centralized log table (ETL_Load_Log) used for auditing ETL processes across the entire project.

By adopting this structured control flow, the ETL becomes clear, sequential, and easy to test or rerun when needed. Moreover, it supports future automation via scheduling tools like SQL Server Agent.

Our team found that having predefined variables for logging—such as rowcount, ETLUser, and Status—combined with centralized logging significantly reduced testing time, especially when handling multiple dimension tables concurrently.

Error Logging and Failure Status Recording in Dimension Table ETL

In addition to logging successful executions, the ETL system is also designed to capture detailed information when a failure occurs during the data loading process. This enables the operations team to quickly pinpoint the root cause and location of the error, enhancing overall monitoring and recovery capabilities.

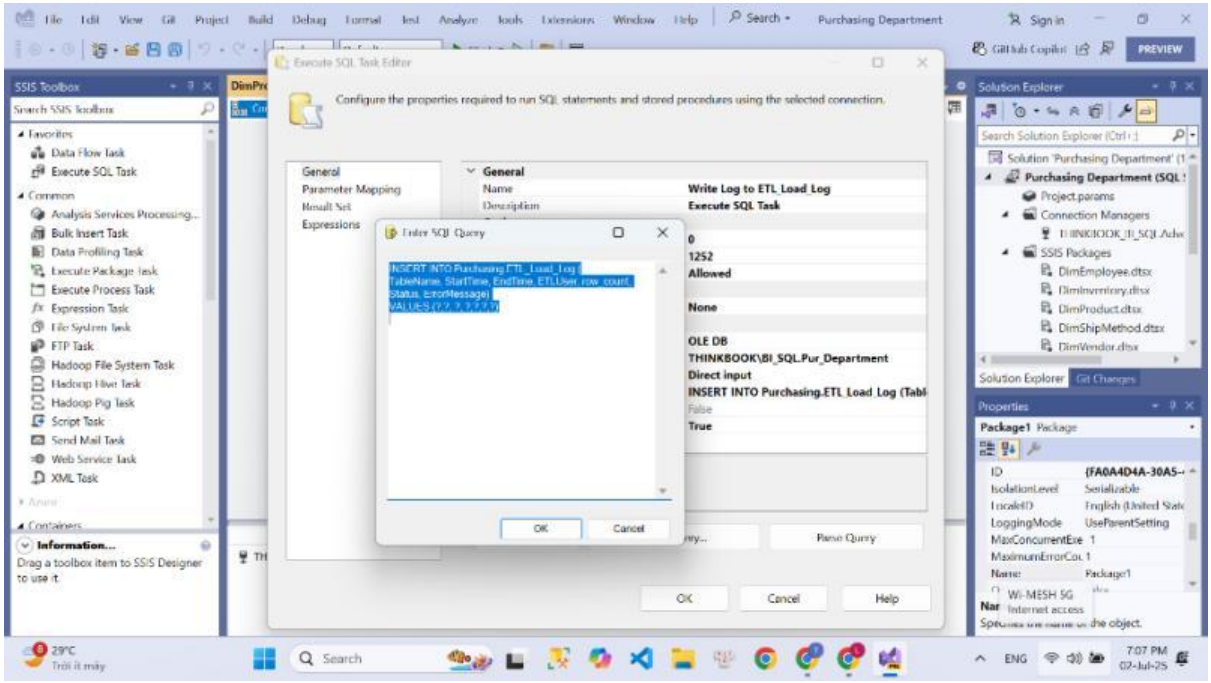


Figure 4.4. Task for Logging ETL Status

The Execute SQL Task window contains an INSERT INTO Purchasing.ETL_Load_Log statement, which records the following:

- The name of the dimension table being processed
- The user executing the ETL process (retrieved from the ETLUser variable)
- The number of rows processed (row_count)
- The completion time (EndTime)
- The execution status (e.g., "Success")
- Any error message, if applicable (ErrorMessage — typically left blank on success)

This SQL command uses dynamic SSIS variables, making it reusable across multiple dimension tables with minimal configuration changes.

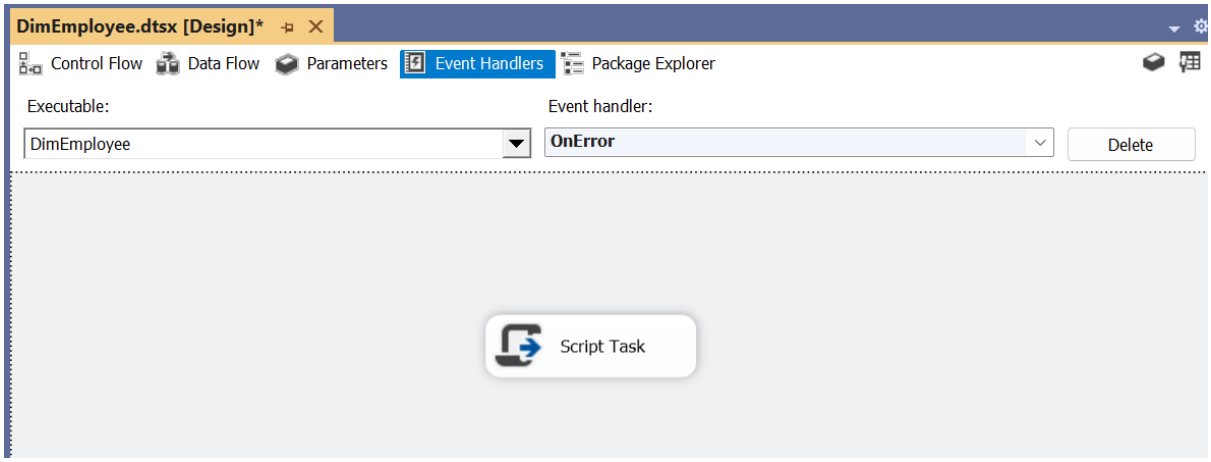


Figure 4.5. OnError Event – DimEmployee

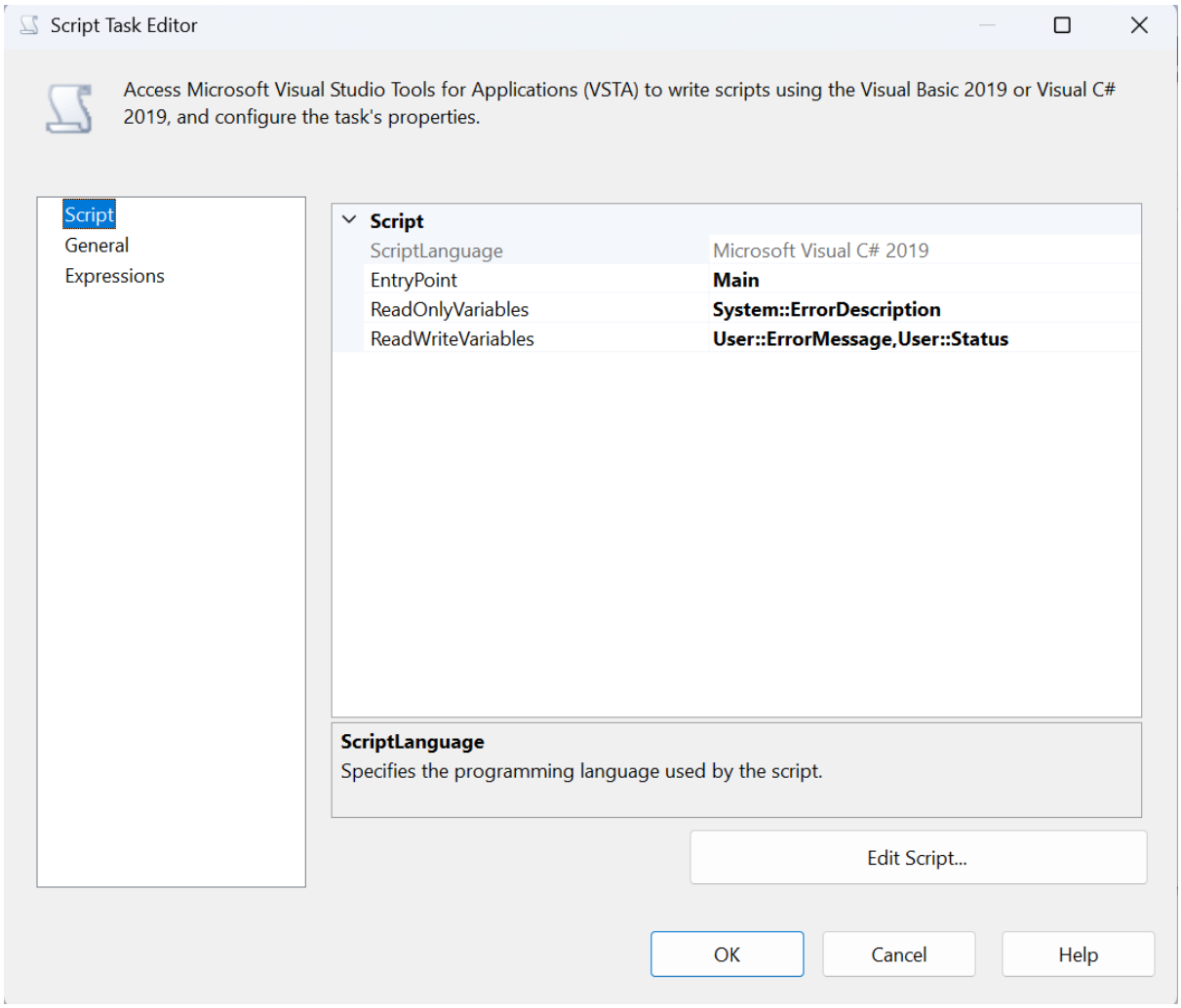


Figure 4.6. Script Task – Log Error & Failure Status

Additionally, an Event Handler with a Script Task is configured for the OnError event of the package. If any task fails during execution, the handler will:

- Retrieve the system-generated error description via the System::ErrorDescription variable
- Assign this error message to the user-defined variable User::ErrorMessage
- Set the Status variable to "Failed"

The script inside this task is written in Microsoft Visual C# 2019, allowing for flexible and clear error-handling logic within the control flow.

Data Flow Task – ETL for DimEmployee Table

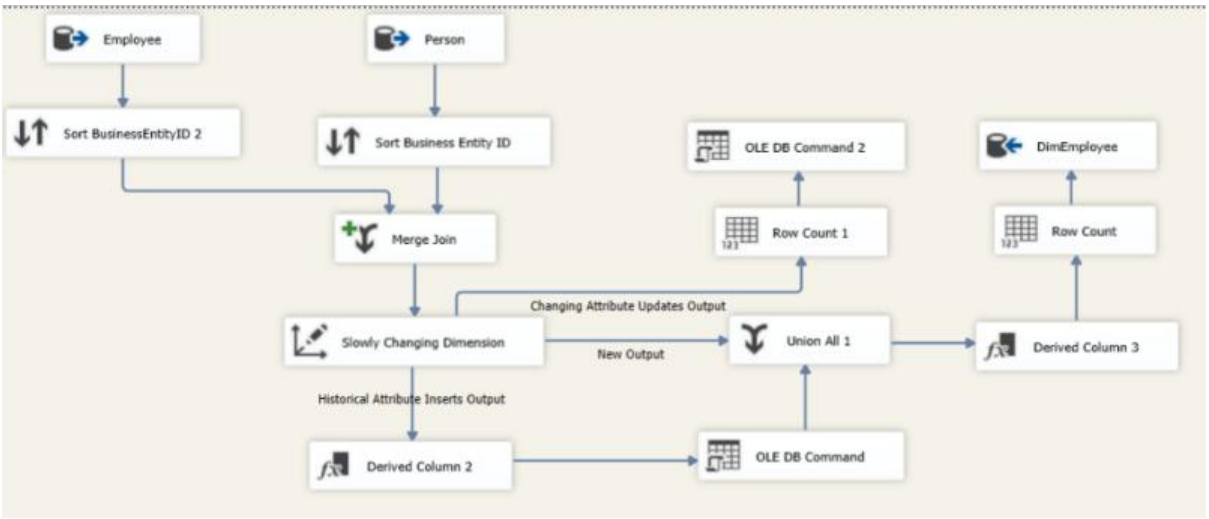


Figure 4.7. Data Flow for ETL of DimEmployee using SCD Techniques

The ETL process for the DimEmployee table begins by sourcing data from two operational tables: Employee and Person. After sorting each dataset by BusinessEntityID, the two streams are merged using a Merge Join, producing a comprehensive record that includes both personal and employment-related details.

Once joined, the data flows through the Slowly Changing Dimension (SCD) transformation. This component determines the appropriate handling strategy based on how attribute values have changed.

Based on the configuration used in this project, the attributes in the DimEmployee table are categorized as follows:

- SCD Type 2: Applied to the JobTitle attribute. When the job title changes, a new record is inserted, while the previous record is updated with an EndDate. This approach preserves the full history of role changes over time.
- SCD Type 1: Used for fields like FirstName, MiddleName, LastName, Gender, and SalariedFlag. These attributes are overwritten upon change—no historical values are retained.
- SCD Type 0: Applied to stable attributes such as BusinessEntityID, StartDate, EndDate, and CurrentFlag. These fields remain constant and are never updated once inserted.

Based on the SCD logic:

- Records identified as new or Type 2 changes are routed through the New Output or Historical Attribute Inserts Output branches. Additional system fields are appended via a Derived Column transformation before being inserted into the target table.
- Records requiring Type 1 updates are processed using an OLE DB Command to apply in-place updates.

All resulting rows—whether inserted or updated—are consolidated using a Union All transformation. System metadata such as LoadDate is added, the number of rows processed is counted via Row Count, and the final dataset is loaded into the DimEmployee table.

This multi-type SCD setup enables the system to not only reflect the most current employee data but also maintain a traceable history of job title changes. This design supports both point-in-time reporting and longitudinal workforce analysis.

Data Flow Task – ETL for the DimProduct Table

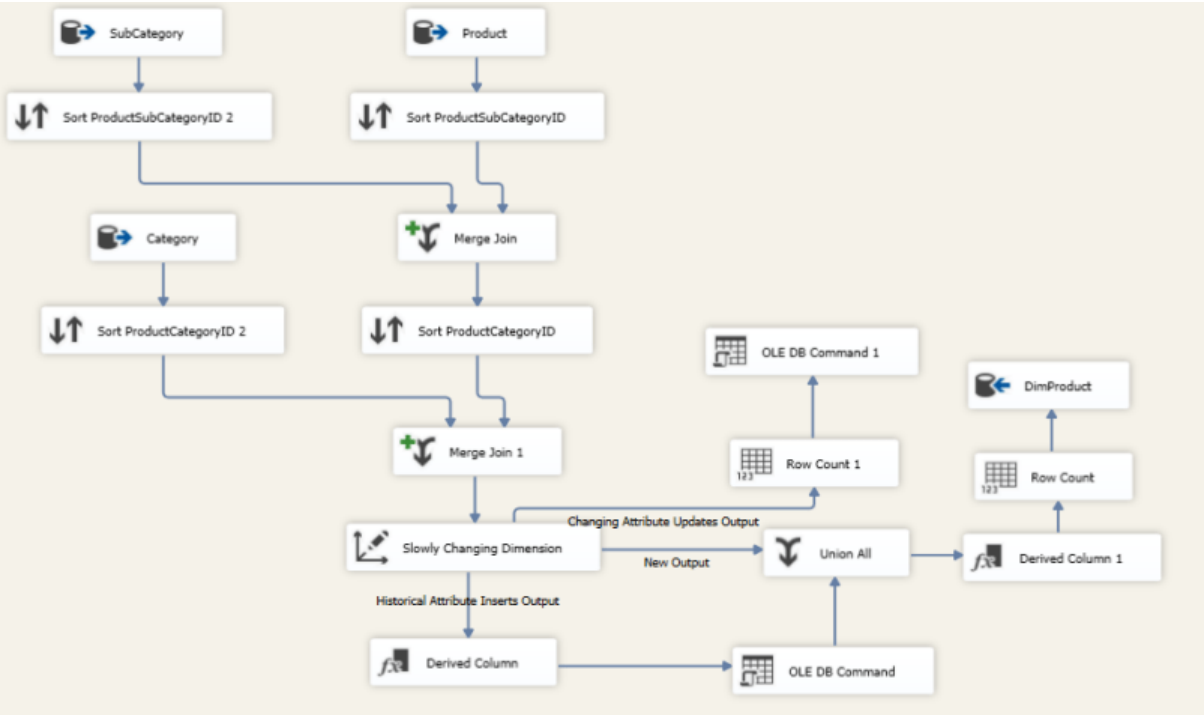


Figure 4.8. Detailed Data Flow for ETL of DimProduct Using SCD

The ETL process for the DimProduct table begins by integrating data from three primary source tables: Product, ProductSubcategory, and ProductCategory. Each table is first sorted by its corresponding foreign key (ProductSubcategoryID, ProductCategoryID) before being merged through two sequential Merge Join steps. This produces a complete product record that includes classification, category, and descriptive information.

Once combined, the data passes through a Slowly Changing Dimension (SCD) transformation, which categorizes each record based on the type and scope of attribute changes to determine the appropriate update strategy.

The attributes in DimProduct are categorized as follows:

- SCD Type 2: Applied to the StandardCost attribute, which tracks changes in product cost over time. When this value changes, a new record is inserted, and the previous one is updated with an EndDate to maintain historical accuracy.

- SCD Type 1: Used for attributes such as ProductName, ProductNumber, Color, Size, and Weight. These fields are updated in place whenever changes occur, and prior values are not retained.

- SCD Type 0: Assigned to static or identifying fields including ProductKey, ProductID, StartDate, EndDate, ProductCategory, and ProductSubcategory. These fields remain unchanged throughout the record's lifetime.

Based on this classification:

- New records and those with Type 2 changes are routed through the New Output or Historical Attribute Inserts Output, followed by a Derived Column step to add system fields before being inserted into the destination table.

- Records with Type 1 changes are handled via an OLE DB Command, which performs in-place updates to the existing records.

All processed rows—whether inserted or updated—are merged using a Union All transformation. System metadata is appended through a Derived Column, and the number of processed rows is counted using a Row Count transformation before loading the final dataset into the DimProduct table.

This approach ensures the system maintains up-to-date product data while also preserving historical changes in critical fields such as cost—supporting more robust business intelligence and analytical capabilities.

Data Flow Task – ETL for the DimShipMethod Table

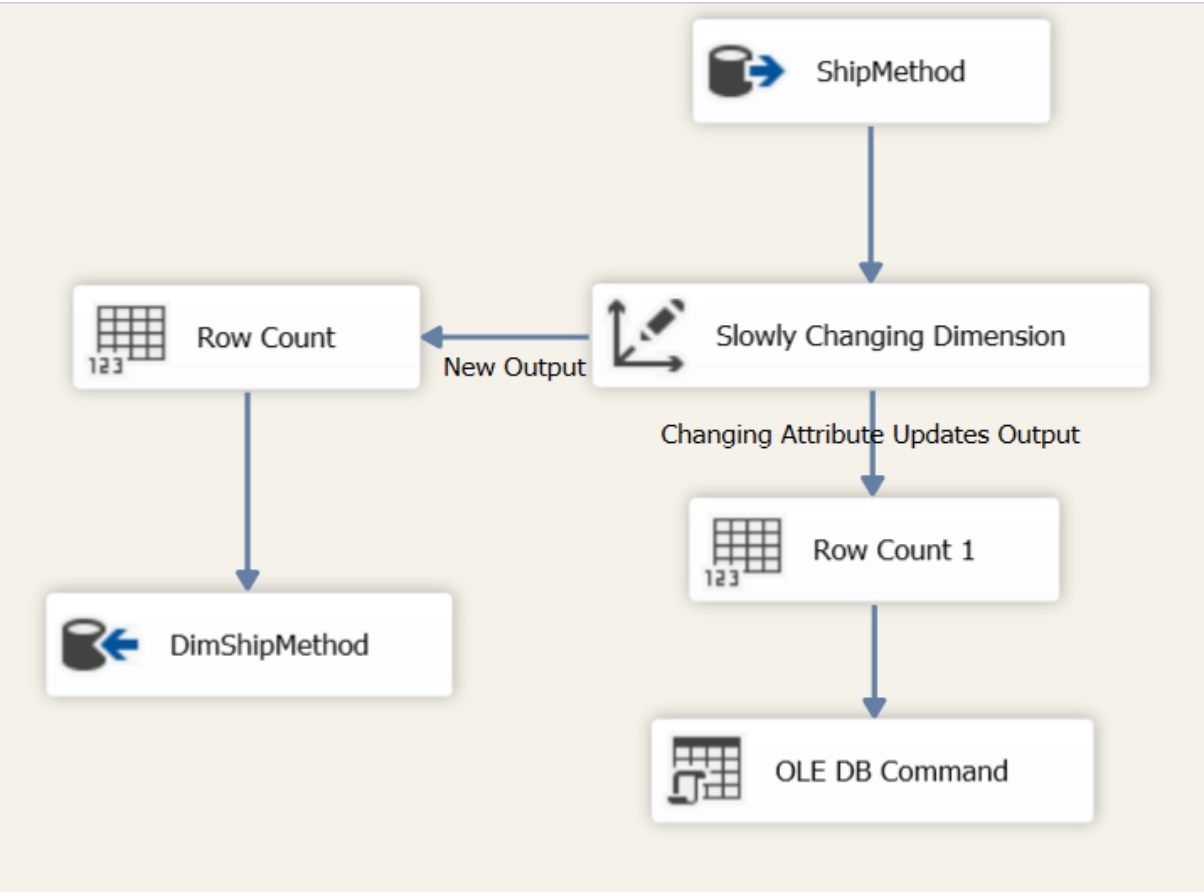


Figure 4.9. Detailed Data Flow for ETL of DimShipMethod Using SCD

The data for the DimShipMethod table is sourced directly from the ShipMethod table. The ETL process is managed through a Slowly Changing Dimension (SCD) component to ensure that the warehouse reflects current shipping information and handles changes appropriately.

In this flow, the attributes from ShipMethod are managed using two main SCD types:

- SCD Type 1: Applied to attributes that may change over time but do not require historical tracking—such as Name, ShipBase, and ShipRate. Updates to these fields are applied directly to the existing records via an OLE DB Command.
- SCD Type 0 (default): Used for identifier columns such as ShipMethodID, which remain constant and are not modified during the ETL process.

New records—those not previously found in the destination table—are routed through the New Output branch. These records are counted using the Row Count transformation and then inserted into the DimShipMethod table.

Overall, while this ETL flow is simpler compared to those of other dimension tables, it still ensures data accuracy, consistency, and efficient updates for shipping method information.

Data Flow Task – ETL for the DimVendor Table

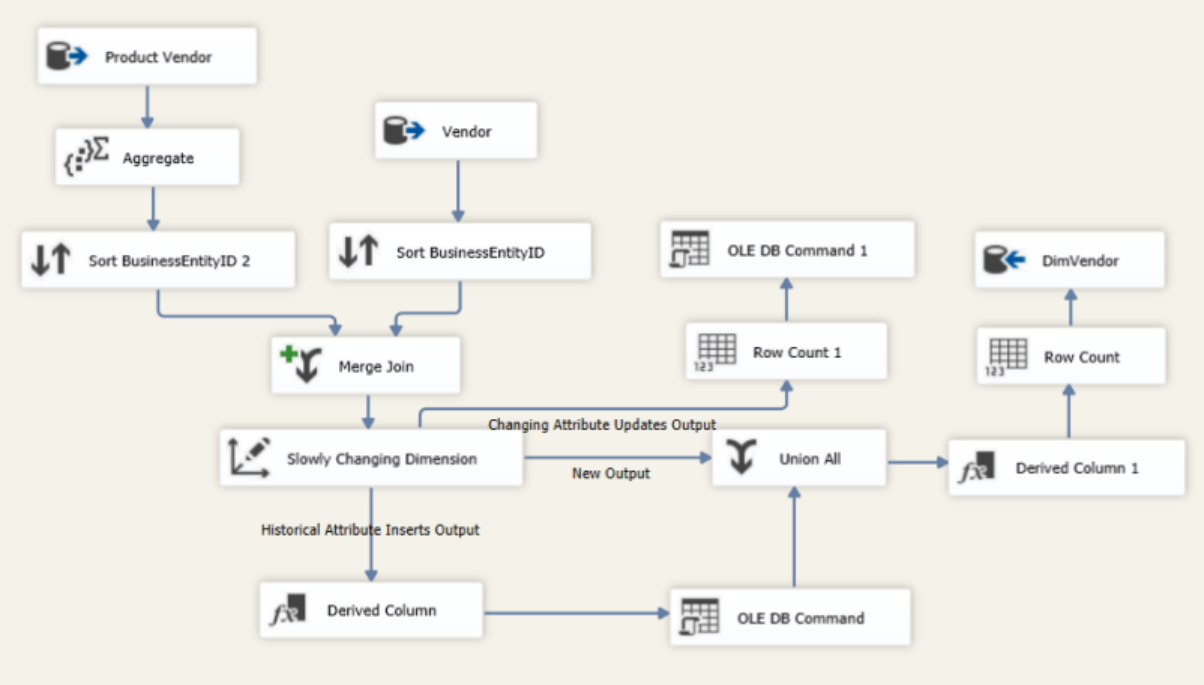


Figure 4.10. Detailed Data Flow for ETL of DimVendor Using SCD

The DimVendor table is constructed from two primary data sources: the Vendor and ProductVendor tables. The ProductVendor table is first processed through an Aggregate transformation to normalize and summarize the data. It is then merged with the Vendor table using a Merge Join on the BusinessEntityID key. This join enables the consolidation of detailed information on each vendor and the products they supply.

Following the join, the dataset flows into a Slowly Changing Dimension (SCD) component, which analyzes attribute changes and determines the appropriate handling for each record.

The SCD configuration for DimVendor includes:

- SCD Type 2: Applied to the CreditRating attribute to preserve the history of vendor credit changes. When this value changes, a new record is created while the old one is closed with an updated EndDate.
- SCD Type 1: Used for fields such as Name, ActiveFlag, and StandardPrice, where updates are made directly to the existing records without retaining previous values.
- SCD Type 0: Assigned to fixed or identifier fields such as VendorKey, VendorID, StartDate, and EndDate, which remain unchanged throughout the ETL process.

The processing logic is as follows:

- New records and Type 2 changes are routed through the New Output or Historical Attribute Inserts Output branches. After a Derived Column step adds necessary system fields, the data is inserted into the destination table.
- Type 1 updates are handled using an OLE DB Command, which directly updates the existing records.

Finally, all rows—whether inserted or updated—are consolidated using a Union All transformation, followed by Row Count and a Derived Column to append additional metadata, before being loaded into the DimVendor table.

This SCD-based implementation ensures that the system reflects the current vendor status while retaining the historical changes in credit rating and supply activity—supporting both operational accuracy and time-based vendor analysis.

Control Flow – Loading Data into FactInventory

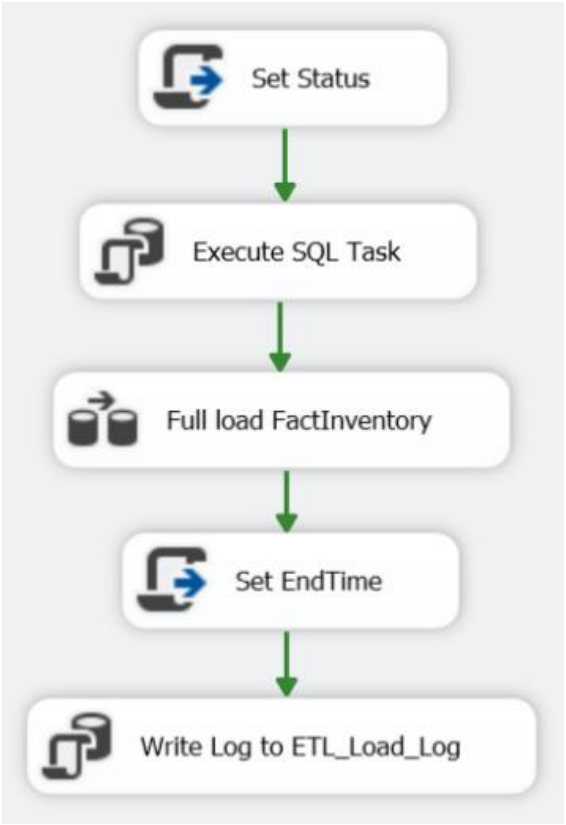


Figure 4.11. Control Flow for Loading Data into FactInventory

The figure above illustrates the main control flow used to fully reload data into the FactInventory table. The process begins with a Script Task to update the initial load status. This is followed by a SQL Task to perform preparatory operations, such as truncating or staging necessary data.

Next, the data is fully loaded into the FactInventory table through a dedicated Data Flow task. Once the data load is complete, the system records the current timestamp using a Set EndTime task. Finally, the process concludes by writing execution metadata to the ETL_Load_Log table, ensuring traceability and enabling comprehensive monitoring of the ETL workflow.

Data Flow – Loading Data into FactInventory

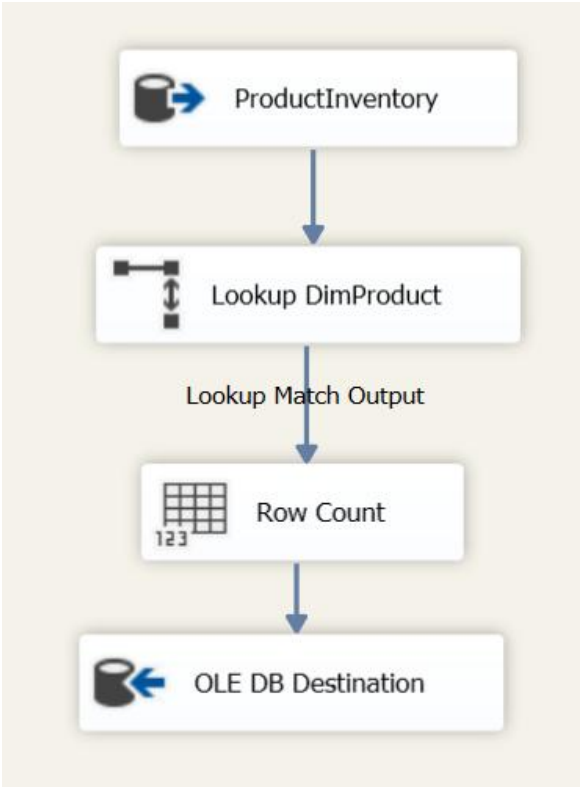


Figure 4.12. Data Flow for Populating FactInventory

The FactInventory table is loaded using a full load approach, where all existing records are refreshed during each ETL run. The source data originates from the ProductInventory table in the transactional system and is processed through a series of transformations to ensure data integrity and consistency.

A key step involves the Lookup DimProduct transformation, which matches each product in the source data with its corresponding surrogate key in the dimension table. Only records with a successful match (Lookup Match Output) are passed forward, ensuring that referential integrity is preserved.

Subsequently, a Row Count transformation captures the number of records being inserted, enabling monitoring and audit logging. Finally, the cleansed and validated dataset is loaded into the FactInventory table via an OLE DB Destination.

This structured data flow ensures accurate foreign key resolution, consistent loading behavior, and reliable performance for downstream analytics.

Control Flow – Loading Data into FactPurchaseOrders

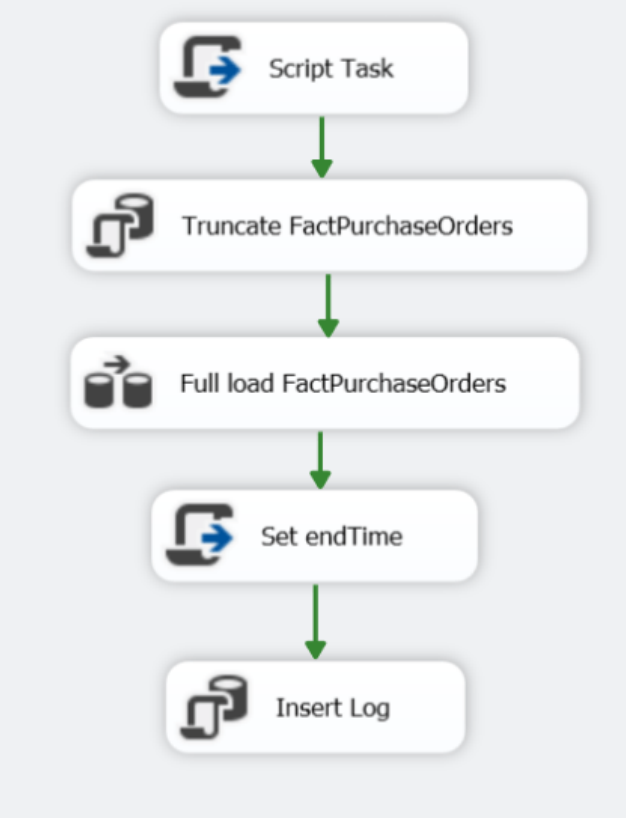


Figure 4.13. Control Flow for Loading Data into FactPurchaseOrders

The figure above illustrates the main control flow used to fully reload data into the FactPurchaseOrders table. The process begins with a Script Task to initialize system variables. Next, the fact table is cleared using a Truncate operation, followed by the execution of the detailed data loading workflow through a dedicated Data Flow task.

Once the loading process is completed, the system captures the current timestamp via a Set endTime task, and processing metadata is written to the log table through an Insert Log step. This control flow ensures clear traceability, structured execution, and ease of monitoring throughout the ETL process.

Data Flow – Loading Data into FactPurchaseOrders

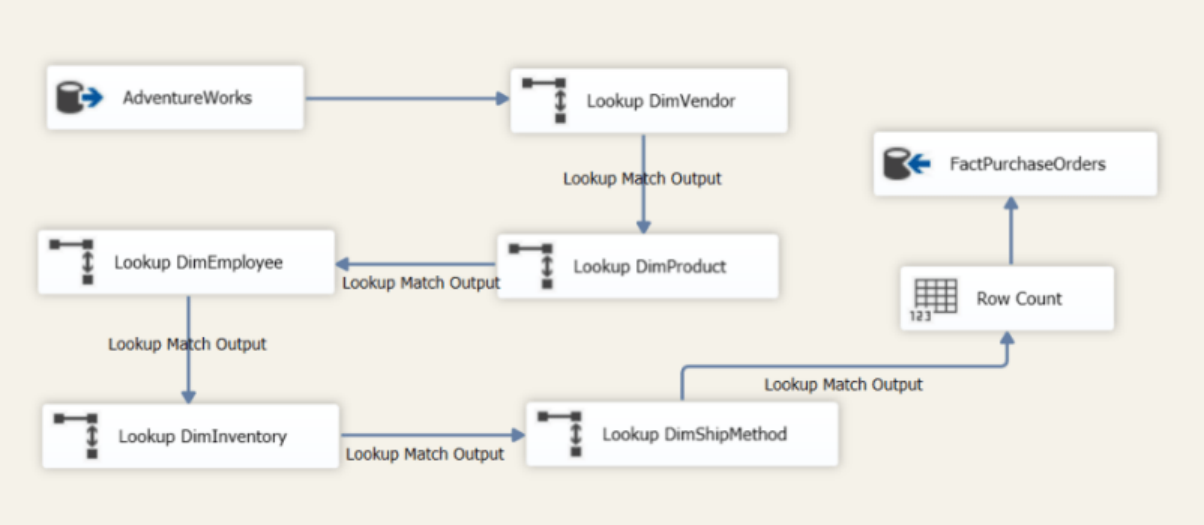


Figure 4.14. Data Flow for Populating FactPurchaseOrders

The FactPurchaseOrders table is loaded using a full load approach, meaning the table is completely refreshed with each ETL execution. Data is sourced from the transactional system (*AdventureWorks*) and passed through a series of Lookup transformations to resolve foreign keys with the corresponding dimension tables.

Specifically, the data is matched against the following dimensions:

- Lookup DimVendor: retrieves the supplier key
- Lookup DimProduct: maps the product identifier
- Lookup DimEmployee: resolves the employee handling the order
- Lookup DimShipMethod: assigns the shipping method key

Each lookup uses the Match Output path to ensure that only records with valid foreign key relationships are passed forward, maintaining referential integrity across the data warehouse.

After all lookups are completed, a Row Count transformation tallies the number of records loaded. The final dataset is then written to the FactPurchaseOrders table.

By resolving all dimension keys prior to insertion, this design enhances performance, ensures data consistency, and enables robust analytical querying downstream.

Once the data has been processed and loaded into the FactPurchaseOrders table, the overall ETL workflow continues with the execution of the master pipeline. This main pipeline is responsible for orchestrating the sequence of child packages, managing the flow of the entire ETL process. Such an architecture ensures that all tables are updated completely and in the correct logical order, resulting in a fully automated and end-to-end integrated data processing system.

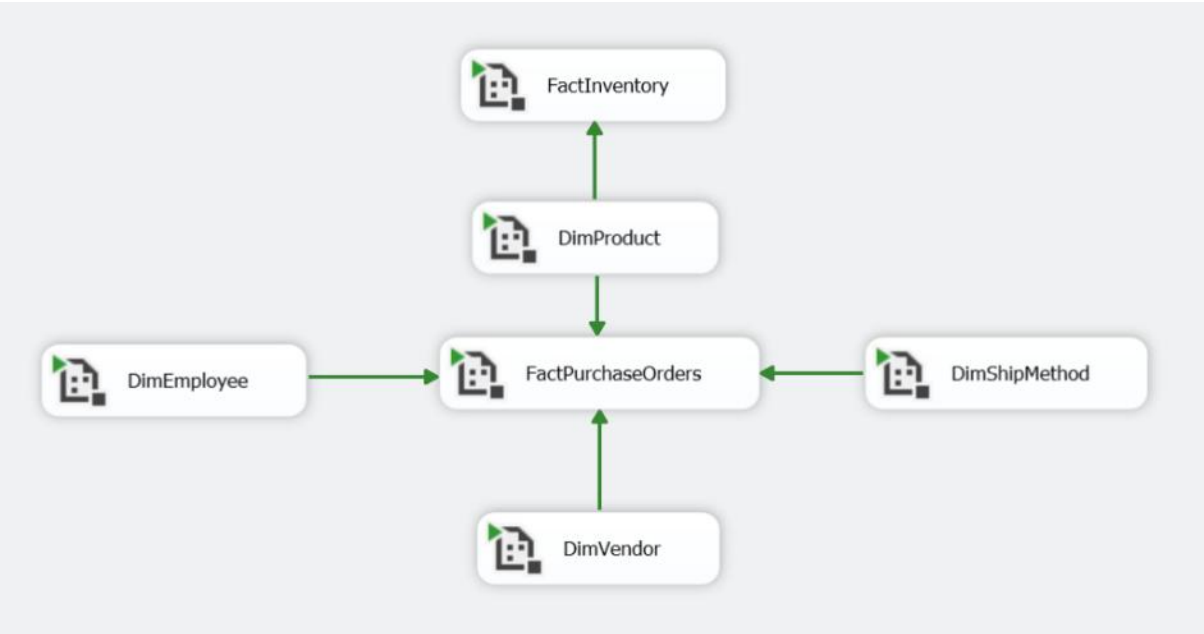
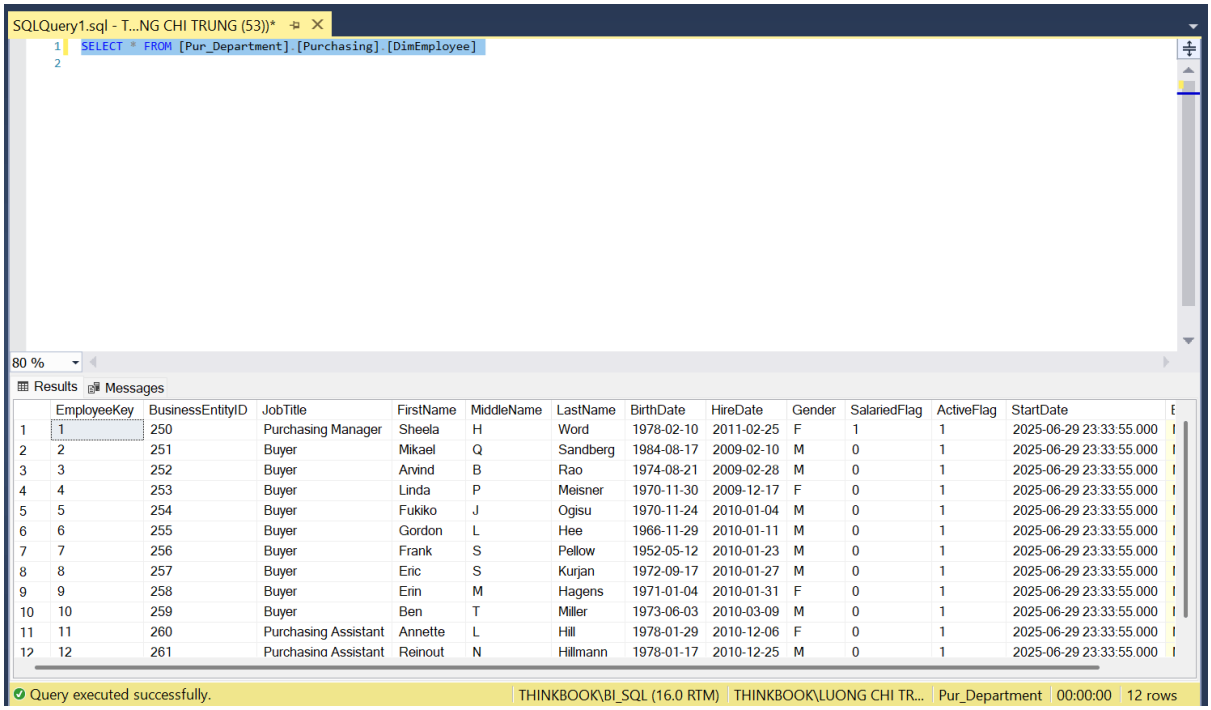


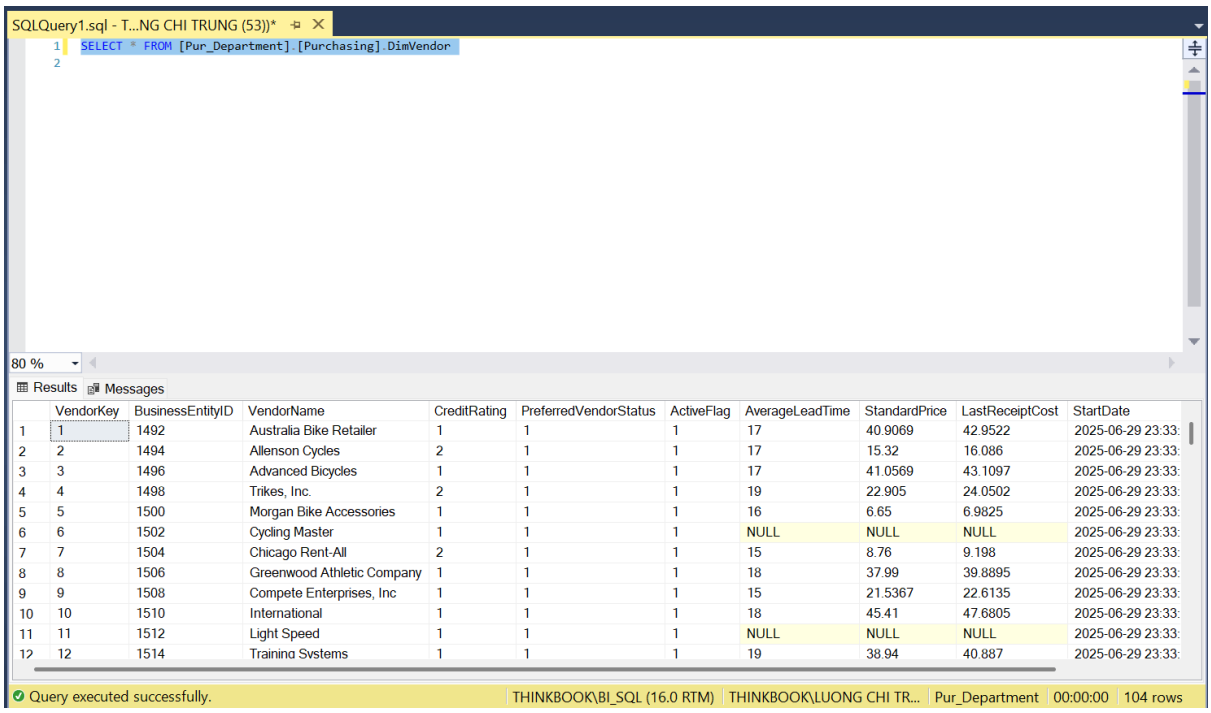
Figure 4.15. ETL Completion

After the full ETL process has been completed, data has been successfully loaded into all relevant tables within the data warehouse, including both dimension and fact tables. The following section presents query results retrieved directly from these tables, reflecting the final output after successful deployment.



	EmployeeKey	BusinessEntityID	JobTitle	FirstName	MiddleName	LastName	BirthDate	HireDate	Gender	SalariedFlag	ActiveFlag	StartDate	E
1	1	250	Purchasing Manager	Sheela	H	Word	1978-02-10	2011-02-25	F	1	1	2025-06-29 23:33:55.000	I
2	2	251	Buyer	Mikael	Q	Sandberg	1984-08-17	2009-02-10	M	0	1	2025-06-29 23:33:55.000	I
3	3	252	Buyer	Arvind	B	Rao	1974-08-21	2009-02-28	M	0	1	2025-06-29 23:33:55.000	I
4	4	253	Buyer	Linda	P	Meisner	1970-11-30	2009-12-17	F	0	1	2025-06-29 23:33:55.000	I
5	5	254	Buyer	Fukiko	J	Ogisu	1970-11-24	2010-01-04	M	0	1	2025-06-29 23:33:55.000	I
6	6	255	Buyer	Gordon	L	Hee	1966-11-29	2010-01-11	M	0	1	2025-06-29 23:33:55.000	I
7	7	256	Buyer	Frank	S	Pellow	1952-05-12	2010-01-23	M	0	1	2025-06-29 23:33:55.000	I
8	8	257	Buyer	Eric	S	Kurjan	1972-09-17	2010-01-27	M	0	1	2025-06-29 23:33:55.000	I
9	9	258	Buyer	Erin	M	Hagens	1971-01-04	2010-01-31	F	0	1	2025-06-29 23:33:55.000	I
10	10	259	Buyer	Ben	T	Miller	1973-06-03	2010-03-09	M	0	1	2025-06-29 23:33:55.000	I
11	11	260	Purchasing Assistant	Annette	L	Hill	1978-01-29	2010-12-06	F	0	1	2025-06-29 23:33:55.000	I
12	12	261	Purchasing Assistant	Reinout	N	Hillmann	1978-01-17	2010-12-25	M	0	1	2025-06-29 23:33:55.000	I

Figure 4.16. DimEmployee Result



	VendorKey	BusinessEntityID	VendorName	CreditRating	PreferredVendorStatus	ActiveFlag	AverageLeadTime	StandardPrice	LastReceiptCost	StartDate
1	1	1492	Australia Bike Retailer	1	1	1	17	40.9069	42.9522	2025-06-29 23:33:
2	2	1494	Allenson Cycles	2	1	1	17	15.32	16.086	2025-06-29 23:33:
3	3	1496	Advanced Bicycles	1	1	1	17	41.0569	43.1097	2025-06-29 23:33:
4	4	1498	Trikes, Inc.	2	1	1	19	22.905	24.0502	2025-06-29 23:33:
5	5	1500	Morgan Bike Accessories	1	1	1	16	6.65	6.9825	2025-06-29 23:33:
6	6	1502	Cycling Master	1	1	1	NULL	NULL	NULL	2025-06-29 23:33:
7	7	1504	Chicago Rent-All	2	1	1	15	8.76	9.198	2025-06-29 23:33:
8	8	1506	Greenwood Athletic Company	1	1	1	18	37.99	39.8895	2025-06-29 23:33:
9	9	1508	Compete Enterprises, Inc	1	1	1	15	21.5367	22.6135	2025-06-29 23:33:
10	10	1510	International	1	1	1	18	45.41	47.6805	2025-06-29 23:33:
11	11	1512	Light Speed	1	1	1	NULL	NULL	NULL	2025-06-29 23:33:
12	12	1514	Training Systems	1	1	1	19	38.94	40.887	2025-06-29 23:33:

Figure 4.17. DimVendor result

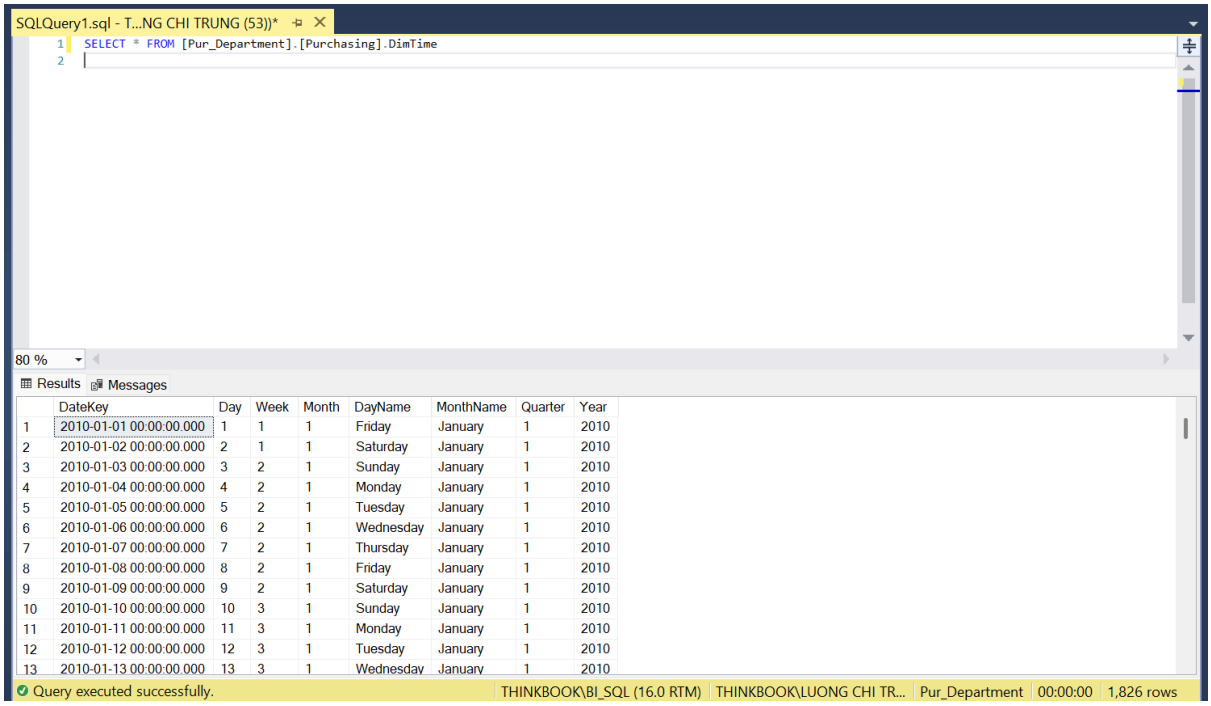


Figure 4.18. DimTime result

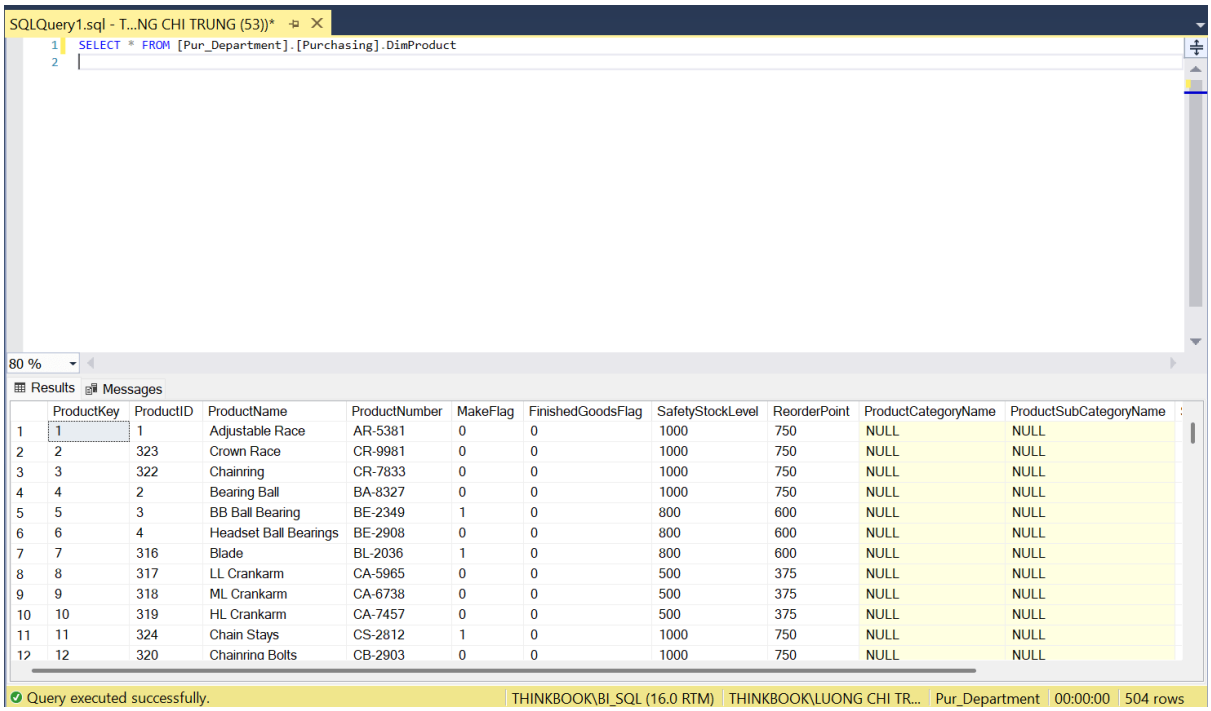


Figure 4.19. DimProduct result

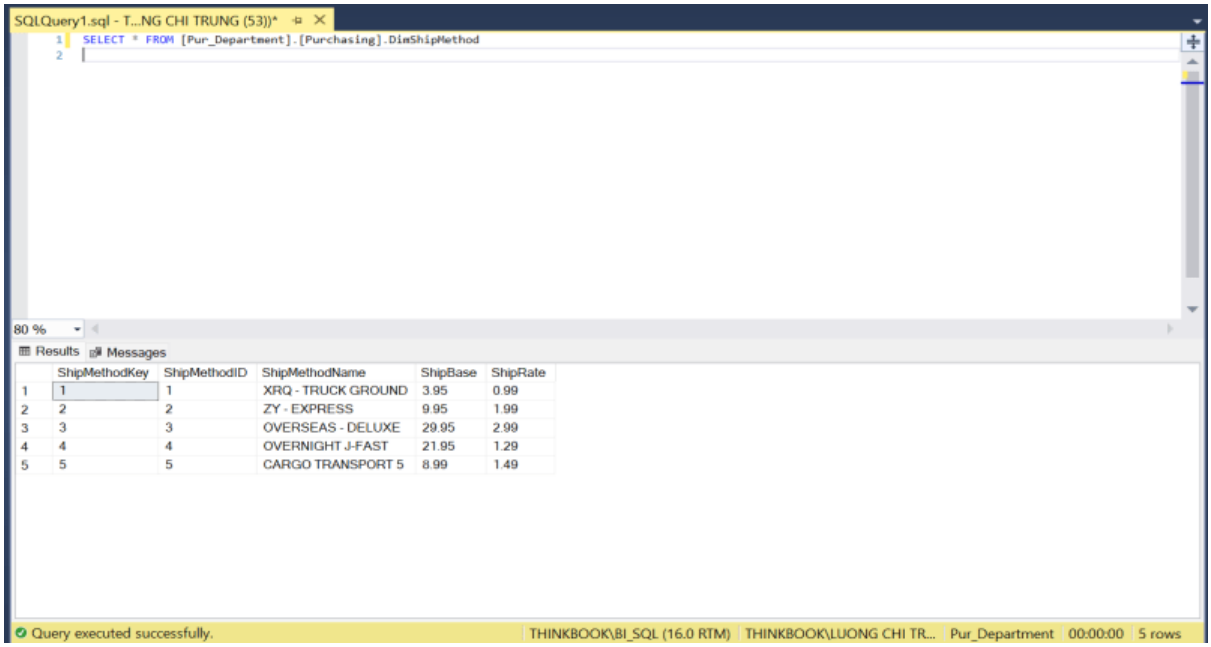


Figure 4.20. DimShipMethod result

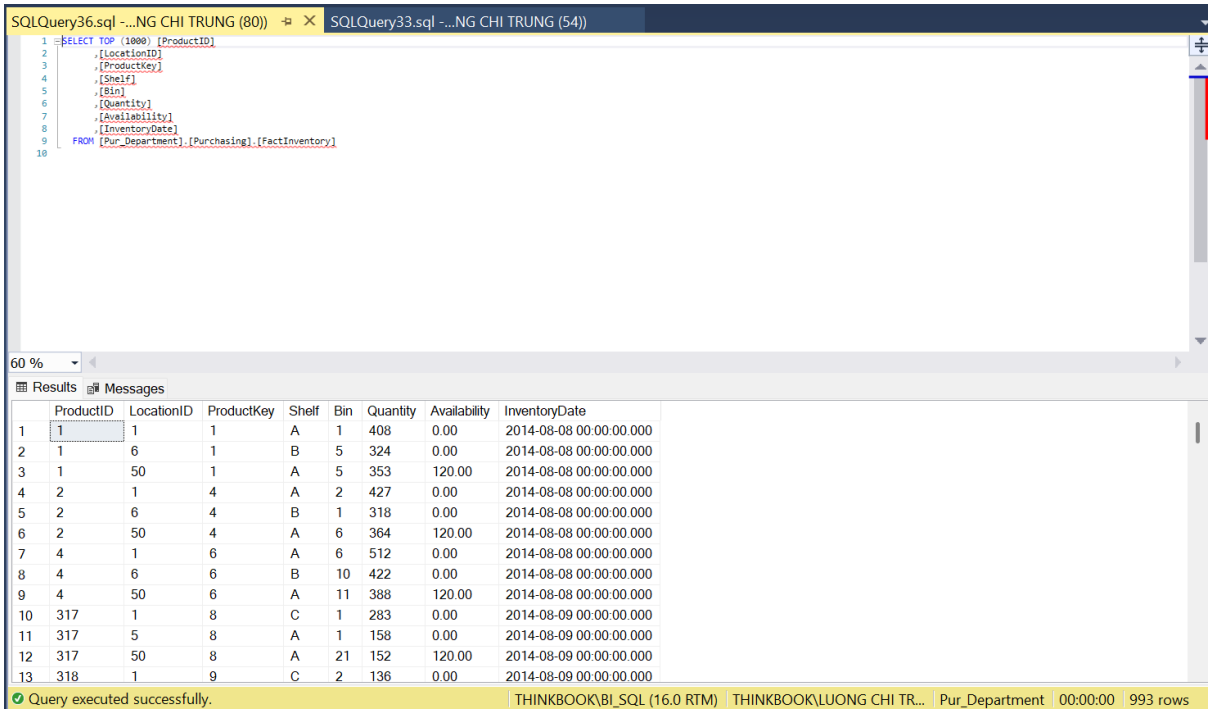


Figure 4.21 FactInventory result

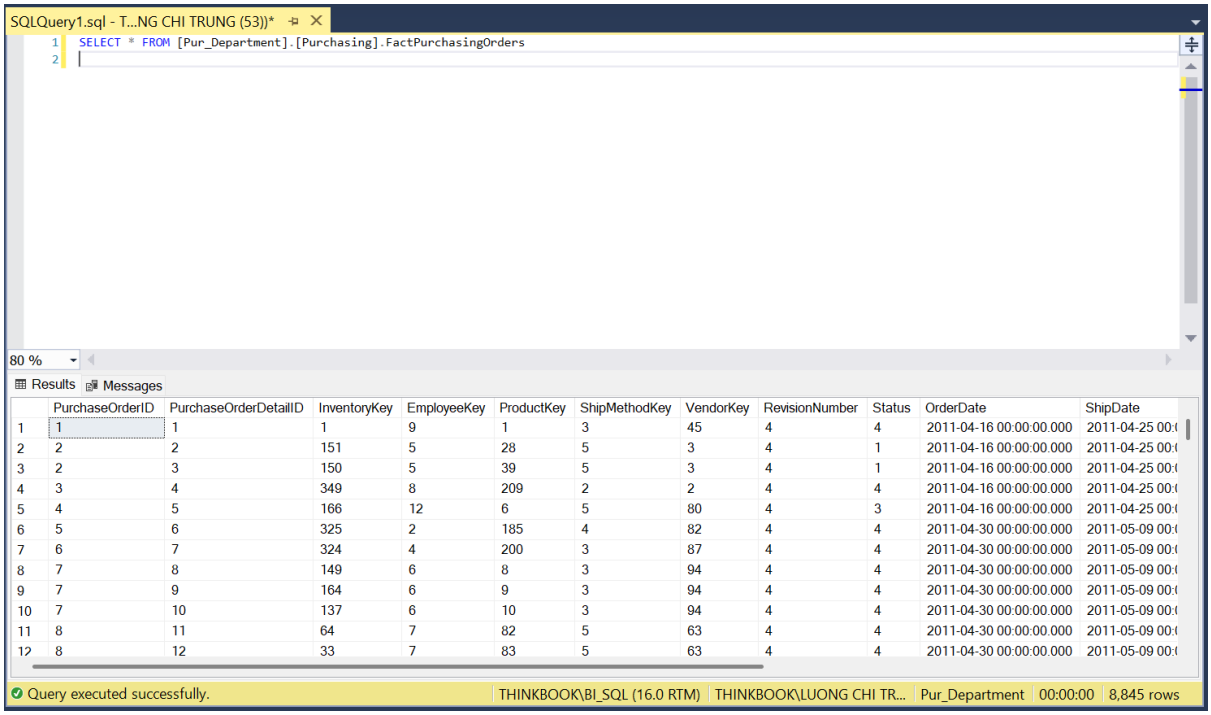


Figure 4.22. FactPurchaseOrders result

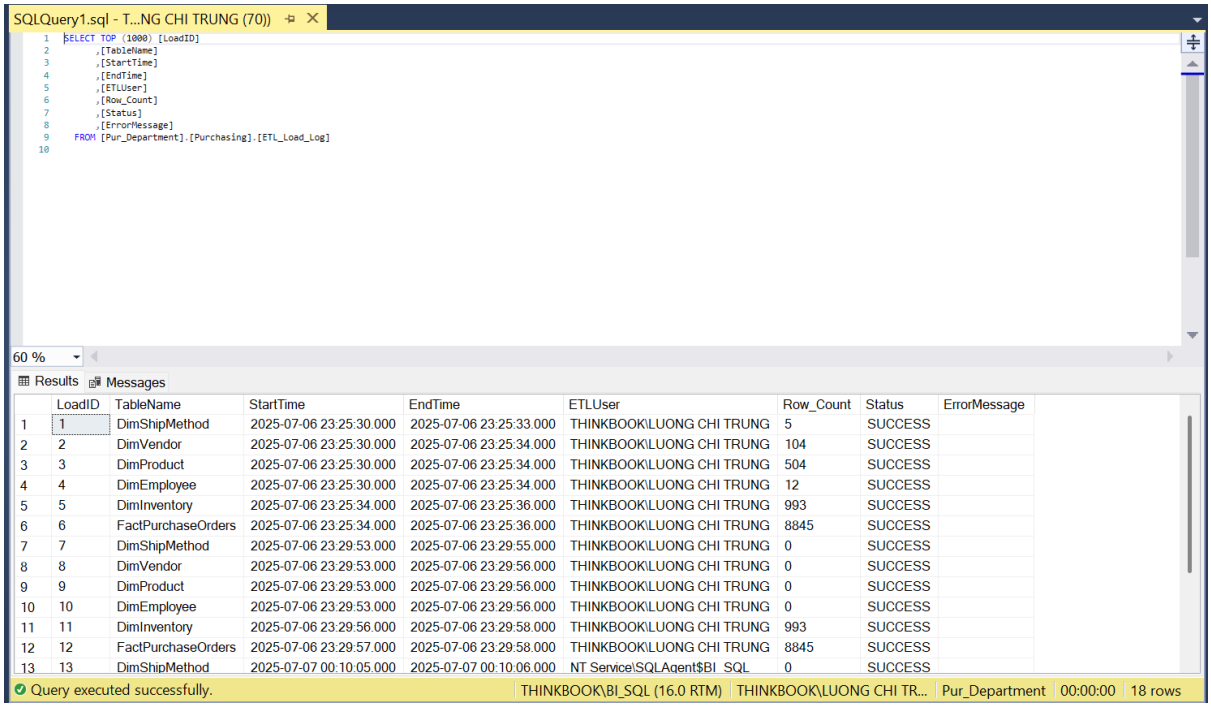


Figure 4.23. ETL Load Log

Deployment to Integration Services Catalogs

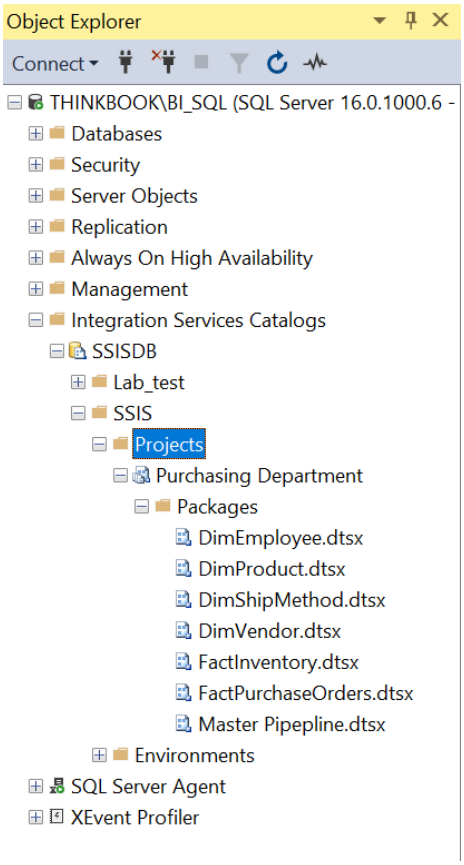


Figure 4.24. Deployed ETL Packages in SSIS Catalog

After the ETL packages were fully designed and tested, all .dtsx packages corresponding to individual data tables were deployed to the Integration Services Catalogs within SQL Server. Specifically, the packages were deployed under the path: SSISDB > SSIS > Projects > Purchasing Department, including all relevant dimension tables (DimEmployee, DimProduct, DimShipMethod, DimVendor) as well as the fact table (FactPurchaseOrders). In addition, a centralized package named Master Pipeline.dtsx was deployed to coordinate the sequential execution of all child packages.

Deploying through Integration Services Catalogs provides a robust foundation for managing, monitoring, and automating the ETL process in a production environment. It also enables seamless integration with SQL Server Agent for scheduling and running ETL jobs as needed.

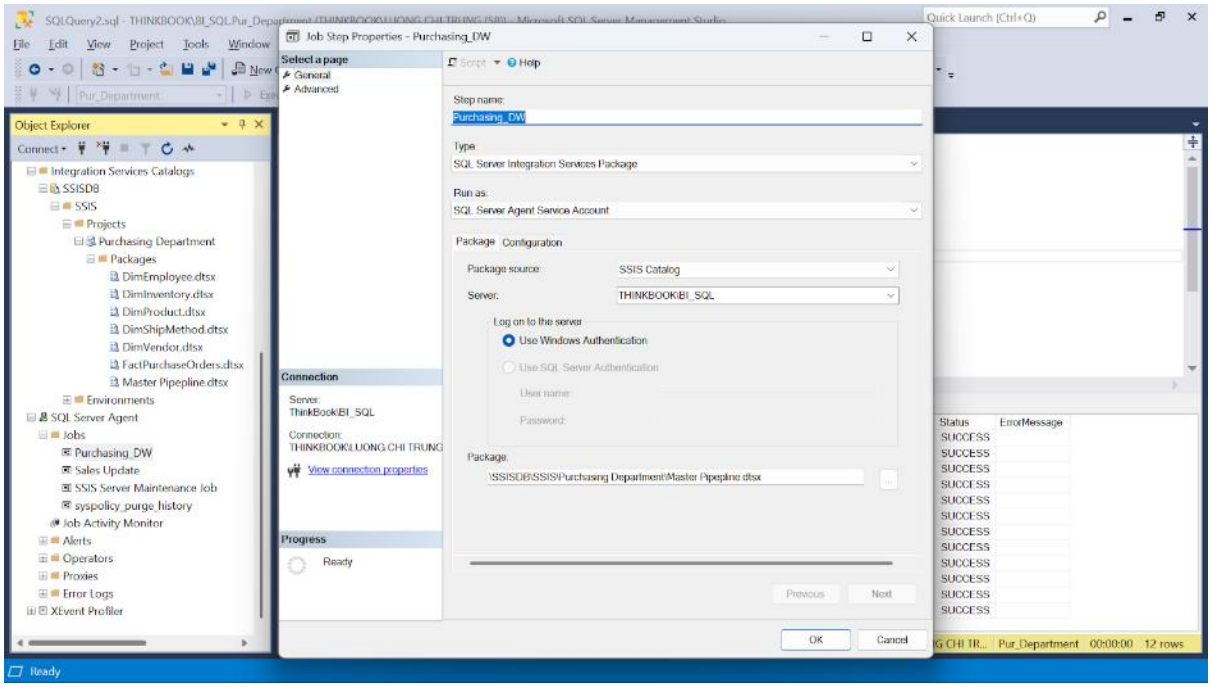


Figure 4.25. Purchasing_DW Job Configuration in SSMS

To streamline and automate the ETL process, the team configured a job using SQL Server Agent. A job named Purchasing_DW was created to trigger the Master Pipeline.dtsx package previously deployed to the Integration Services Catalogs. This master package is responsible for orchestrating the execution of all individual ETL packages corresponding to each table.

Within the job's step configuration, the task type was set to SQL Server Integration Services Package, and the job was run under the SQL Server Agent Service Account. The package source was defined as the SSIS Catalog, accessed via Windows Authentication. Once scheduled, this job enables the ETL pipeline to run automatically at predefined intervals, ensuring the data warehouse remains current and consistent.

The screenshot below confirms that the job executed successfully. All child packages ran with a SUCCESS status, and detailed metadata—such as start time, end time, row counts, execution user, and status—was recorded in the centralized ETL log table for monitoring and audit purposes.

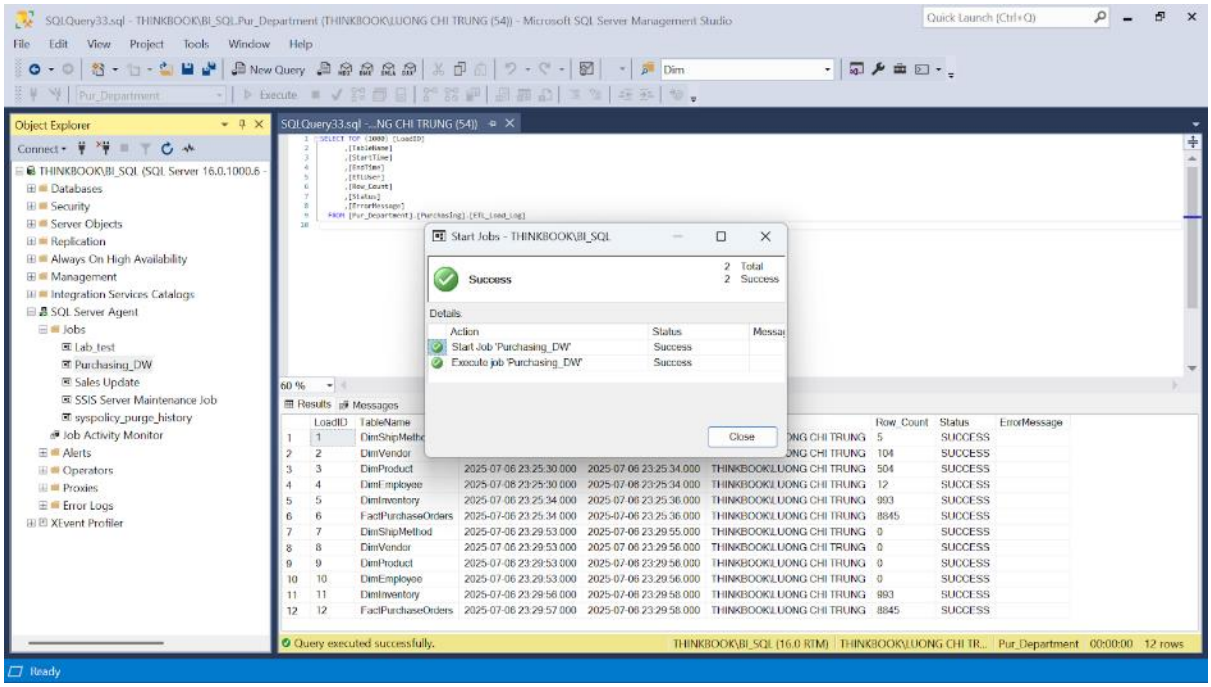


Figure 4.26. Purchasing_DW Job Execution Success Log

Results Messages

	LoadID	TableName	StartTime	EndTime	ETLUser	Row_Count	Status	ErrorMessage
7	7	DimShipMethod	2025-07-06 23:29:53.000	2025-07-06 23:29:55.000	THINKBOOK\LUONG CHI TRUNG	0	SUCCESS	
8	8	DimVendor	2025-07-06 23:29:53.000	2025-07-06 23:29:56.000	THINKBOOK\LUONG CHI TRUNG	0	SUCCESS	
9	9	DimProduct	2025-07-06 23:29:53.000	2025-07-06 23:29:56.000	THINKBOOK\LUONG CHI TRUNG	0	SUCCESS	
10	10	DimEmployee	2025-07-06 23:29:53.000	2025-07-06 23:29:56.000	THINKBOOK\LUONG CHI TRUNG	0	SUCCESS	
11	11	DimInventory	2025-07-06 23:29:56.000	2025-07-06 23:29:58.000	THINKBOOK\LUONG CHI TRUNG	993	SUCCESS	
12	12	FactPurchaseOrders	2025-07-06 23:29:57.000	2025-07-06 23:29:58.000	THINKBOOK\LUONG CHI TRUNG	8845	SUCCESS	
13	13	DimShipMethod	2025-07-07 00:10:05.000	2025-07-07 00:10:06.000	NT Service\SQLAgent$BI_SQL	0	SUCCESS	
14	14	DimEmployee	2025-07-07 00:10:13.000	2025-07-07 00:10:15.000	NT Service\SQLAgent$BI_SQL	0	SUCCESS	
15	15	DimProduct	2025-07-07 00:10:19.000	2025-07-07 00:10:21.000	NT Service\SQLAgent$BI_SQL	0	SUCCESS	
16	16	DimVendor	2025-07-07 00:10:24.000	2025-07-07 00:10:25.000	NT Service\SQLAgent$BI_SQL	0	SUCCESS	
17	17	DimInventory	2025-07-07 00:10:29.000	2025-07-07 00:10:30.000	NT Service\SQLAgent$BI_SQL	993	SUCCESS	
18	18	FactPurchaseOrders	2025-07-07 00:10:34.000	2025-07-07 00:10:36.000	NT Service\SQLAgent$BI_SQL	8845	SUCCESS	

Query executed successfully. | THINKBOOK\BI_SQL (16.0 RTM) | THINKBOOK\LUONG CHI TR... | Pur_Department | 00:00:00 | 18 rows

Figure 4.27. ETL Result Log

CHAPTER 5. DATA ANALYSIS AND VISUALIZATION

5.1. Data analytics with SSAS technology

5.1.1. Building the cube

To install and analyze data using SQL Server Analysis Services (SSAS), we will set up a project called Final in SSAS. This will involve building a cube and conducting data analysis within the project. The process will include the following steps:

Step 1: Set up a new user account and assign administrative privileges on SQL Server

Step 2: Launch Visual Studio and start a new SSAS project titled FinalProject. Connect this project to the data warehouse, which includes dimension and fact tables, using the newly created login credentials.

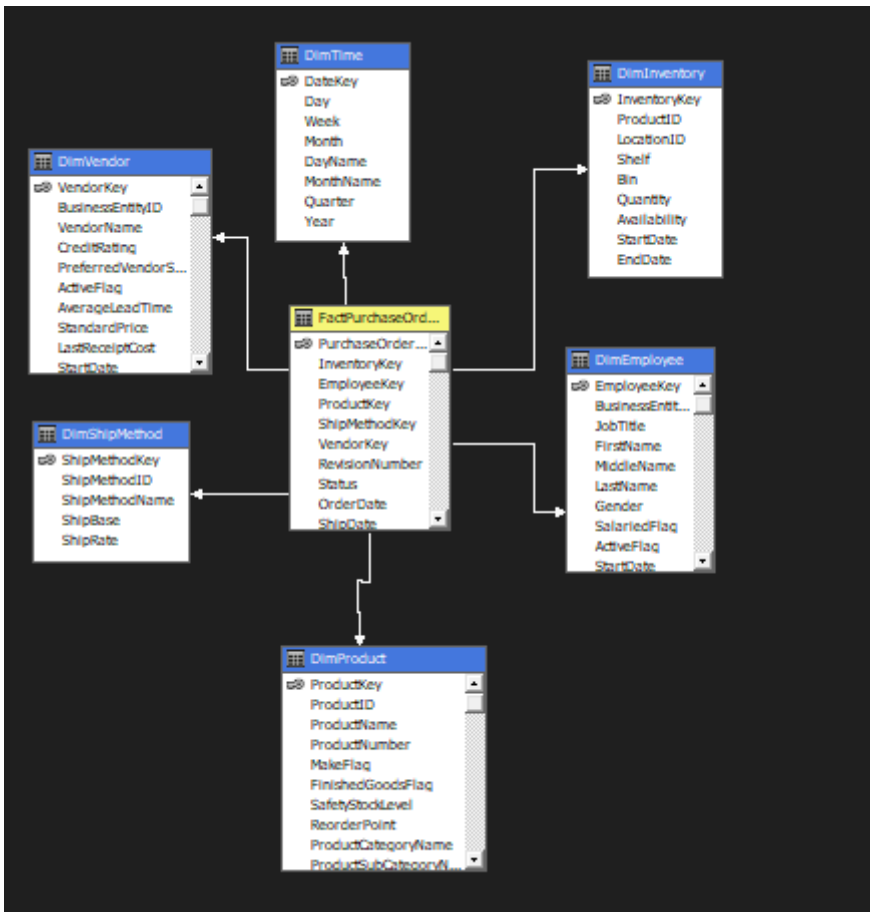


Figure 5.1. SSAS cube

5.1.2. Analysis with SSAS

	Order Qty	Growth Rate
All	2348637	(null)
2011	15977	(null)
2012	140890	8.82
2013	763758	5.42
2014	1428012	1.87

Figure 5.2. GrowthRate of Order Quantity 2011- 2014

The table presents the Growth Rate of Order Quantity from 2011 to 2014. In 2011, the recorded order quantity was 15,977, and no growth rate was calculated as it serves as the base year. In 2012, order quantity sharply increased to 140,890, showing a remarkable growth rate of 8.82, indicating a strong upward trend. This growth continued in 2013, with the order quantity reaching 763,758, though the growth rate slightly declined to 5.42, suggesting a deceleration in growth momentum. By 2014, the order quantity peaked at 1,428,012; however, the growth rate further decreased to 1.87, implying that although orders continued to increase, the pace of growth had significantly slowed. Overall, the period reflects a rapid initial expansion followed by a gradual stabilization in order of growth.

	Line Total	Percent of Total
All	63791994.838	100
Superior Bicycles	4555897.5	7.14
Professional Athletic Consultants	3058774.95	4.79
Chicago City Saddles	3029108.77500001	4.75
Jackson Authority	2553243	4
Vision Cycles, Inc.	2513742	3.94
Sport Fan Co.	2421619.2	3.8
Proseware, Inc.	2347422	3.68
Crowley Sport	2237800.95	3.51
Greenwood Athletic Company	2237800.95	3.51
Mitchell Sports	2193922.5	3.44

Figure 5.3. Top 10 Vendor by Line Total

The table displays the Line Total revenue and its corresponding percentage contribution by customer or organization. The overall revenue generated amounts to approximately $637.92 million. Among the listed entities, Superior Bicycles contributes the highest share, accounting for 7.14% of the total, followed by Professional Athletic Consultants with 4.79%, and Chicago City Saddles close behind at 4.75%. Other notable contributors include Jackson Authority (4%), Vision Cycles, Inc. (3.94%), and Sport Fan Co. (3.8%). At the lower end of the top contributors, Mitchell Sports represents 3.44% of the total revenue. These top ten customers collectively contribute a significant portion of the overall line total, indicating a relatively concentrated revenue base, with a few key accounts driving a large share of the business.

	All	2011	2012	2013	2014
All	2348637	15977	140890	763758	1428012
SUPERSALES INC.	125000	(null)	7500	40000	77500
Custom Frames, Inc.	115500	(null)	7700	36850	70950
Chicago City Saddles	98450	2750	7150	29150	59400
Victory Bikes	79200	(null)	3300	29700	46200
Professional Athletic Consultants	78100	(null)	3850	26950	47300
Circuit Cycles	69300	1650	4950	23100	39600
Compete Enterprises, Inc	69300	1650	3300	24750	39600
Compete, Inc.	68750	1100	2200	25850	39600
First Rate Bicycles	67650	(null)	4950	21450	41250
Electronic Bike Repair & Supplies	67100	(null)	4950	21450	40700
Jackson Authority	66000	(null)	4950	19800	41250
Proseware, Inc.	66000	1650	4400	20350	39600
Vision Cycles, Inc.	66000	(null)	3300	23100	39600
Sport Fan Co.	64350	(null)	4950	18150	41250
Capital Road Cycles	56100	1100	3300	18700	33000
Comfort Road Bicycles	56100	1100	3300	18700	33000
Competition Bike Training Systems	56100	1100	3300	17600	34100
Crowley Sport	56100	1100	3300	17600	34100
Greenwood Athletic Company	56100	(null)	4400	17600	34100
Hill's Bicycle Service	56100	(null)	4400	17600	34100

Figure 5.4. Annual Top 20 Vendors by OrderQty

The table shows the annual order quantity from 2011 to 2014 for the top 20 vendors. The overall order volume across all vendors totaled 2,348,637 units during this period. Among the vendors, SUPERSALES INC. leads with the highest cumulative order quantity of 125,000, demonstrating consistent annual performance with steady increases, peaking at 77,500 units in 2014. Other major contributors include Custom

Frames, Inc. (115,600 units), Chicago City Saddles (98,450 units), and Victory Bikes (79,200 units).

The majority of vendors show growth in order quantities over the years, particularly from 2012 to 2014, with 2014 being the peak year for most vendors. For instance, Professional Athletic Consultants and Circuit Cycles both saw significant growth in 2014, reaching 47,300 and 39,600 units respectively.

Interestingly, many vendors had no recorded orders in 2011, indicating that business relationships or data tracking may have begun from 2012 onward. Overall, the data reflects a strong upward trend in order volume across vendors, highlighting expanding operations or increasing demand year over year.

5.2. Building the KPIs system

After completing the construction of the cube, the team proceeded to create calculations to accurately measure the indicators serving the KPI system. An illustrative example is presented below, showing how the team dragged and dropped attributes from the Measure group, named the calculation, and adjusted the format in the Additional Properties section. This approach was consistently applied to all remaining calculations.

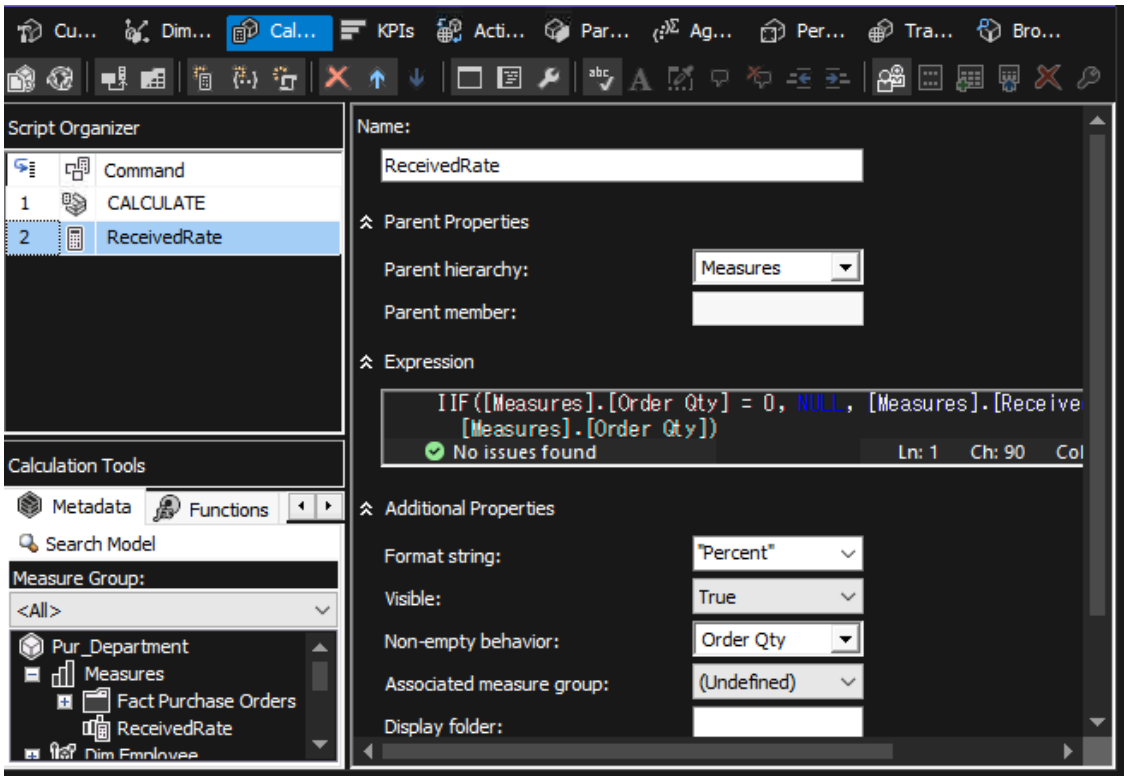


Figure 5.5. ReceivedRate Calculation in SSAS

The ReceivedRate KPI is established to evaluate the effectiveness of suppliers in delivering orders as requested. This indicator reflects the ratio between the quantity of goods received and the quantity of goods ordered, thereby helping the Purchasing Department monitor the reliability and performance of each supplier.

Value Expression: [Measures].[ReceivedRate] – The value is calculated based on the ratio of received quantity to total ordered quantity, as previously defined in the cube using the formula: IIF([Measures].[Order Qty] = 0, NULL, [Measures].[Received Qty] / [Measures].[Order Qty]).

Goal Expression: 0.99 – The target is set at 99%, meaning that at least 99% of the ordered quantity is expected to be received.

Status Indicator: Shows the KPI achievement level, where green indicates the goal is met, red indicates it is not met, and yellow represents a warning or borderline status.

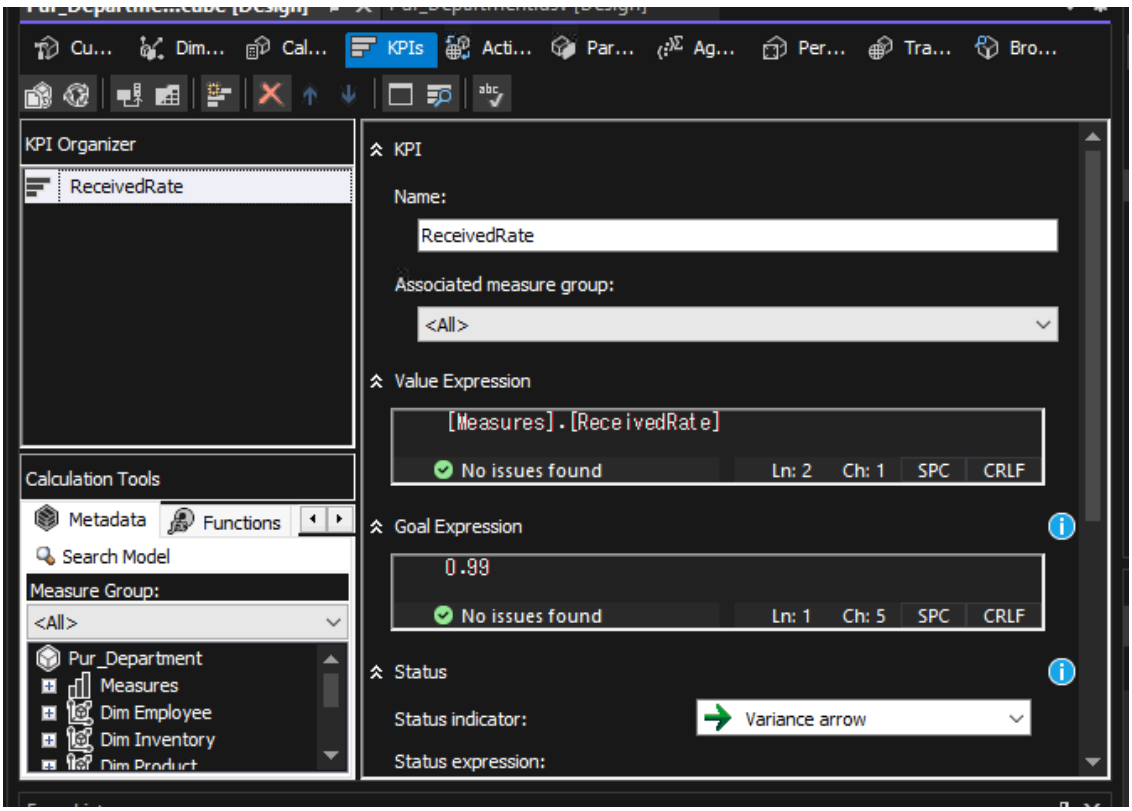


Figure 5.6. Setting up the ReceivedRate KPI in SSAS

After deploying and setting up the ReceivedRate KPI in SSAS, the data is connected and visually presented using a Pivot Table in Excel. The use of Pivot Tables allows for the aggregation, analysis, and comparison of the received rate by supplier and by year, while also incorporating the goal target and KPI status indicator to assess performance effectiveness.

Row Labels	ReceivedRate	ReceivedRate Goal	ReceivedRate Status
A. Datum Corporation		0,99	◆
Advanced Bicycles	100,00%	0,99	●
Allenson Cycles	98,54%	0,99	◆
American Bicycles and Wheels	99,35%	0,99	●
American Bikes	98,83%	0,99	◆
Anderson's Custom Bikes	98,25%	0,99	◆
Aurora Bike Center	100,00%	0,99	●
Australia Bike Retailer	100,00%	0,99	●
Beaumont Bikes	99,66%	0,99	●
Bergeron Off-Roads	100,00%	0,99	●
Bicycle Specialists	98,25%	0,99	◆
Bike Satellite Inc.	100,00%	0,99	●
Bloomington Multisport	100,00%	0,99	●
Burnett Road Warriors	100,00%	0,99	●
Business Equipment Center	100,00%	0,99	●
Capital Road Cycles	99,22%	0,99	●
Carlson Specialties	99,55%	0,99	●
Chicago City Saddles	99,36%	0,99	●
Chicago Rent-All	98,53%	0,99	◆

Figure 5.7. ReceivedRate KPI by Supplier

This table presents the ReceivedRate of suppliers in the system, compared against the established target. The ReceivedRate Status column uses KPI indicators to reflect performance levels:

- A green circle icon indicates that the supplier has a received rate ≥ 99%, meeting the KPI.
- A red diamond icon indicates a received rate < 99%, failing to meet the KPI.

For example, suppliers such as Advanced Bicycles, Aurora Bike Center, and Bergeron Off-Roads all have a 100% rate, while Allenson Cycles (98.54%) and American Bikes (98.83%) do not meet the requirement.

The ReceivedRate KPI by supplier helps the company identify reliable suppliers with consistent delivery performance to maintain partnerships, while also highlighting underperforming vendors for potential adjustment or replacement.

Row Labels	ReceivedRate	ReceivedRate Goal	ReceivedRate Status
2010		0,99	◆
2011	99,31%	0,99	●
2012	99,00%	0,99	●
2013	99,12%	0,99	●
2014	99,08%	0,99	●
Unknown		0,99	◆
Grand Total	**99,09%**	**0,99**	●

Figure 5.8. ReceivedRate KPI by Year

The ReceivedRate KPI by year reflects fluctuations in the company's inbound delivery performance from 2010 to 2014. From 2011 onward, the ReceivedRate consistently remained above 99%, indicating a significant improvement and stabilization in delivery performance over time. Overall, the average rate across the period reached 99.09%, successfully meeting the established KPI target. This table enables management to monitor operational trends, assess the effectiveness of procurement policies, and make appropriate adjustments for each period.

In addition to the ReceivedRate used to evaluate delivery efficiency, the team also developed the ReturnRate KPI to track the proportion of goods returned due to not meeting quality standards. This is an important KPI that reflects the quality of incoming goods and serves as a basis for assessing supplier reliability. By combining both KPIs, the Purchasing Department can not only measure whether goods are delivered as ordered, but also evaluate whether those goods meet the required technical and quality standards.

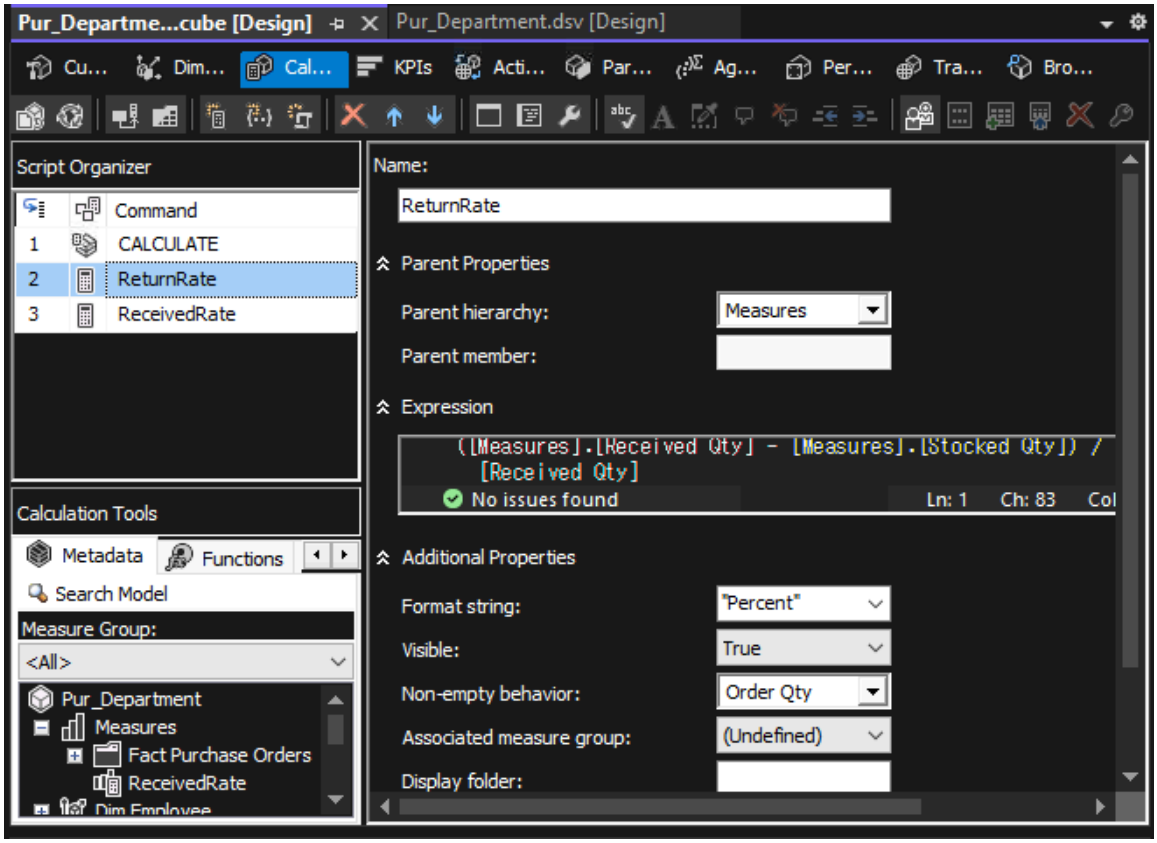


Figure 5.9. ReturnRate Calculation in SSAS

Row Labels	ReturnRate	ReturnRate Goal	ReturnRate Status
A. Datum Corporation		0	
Advanced Bicycles	1,01%	0	
Allenson Cycles	1,19%	0	
American Bicycles and Wheels	0,66%	0	
American Bikes	3,76%	0	
Anderson's Custom Bikes	5,48%	0	
Aurora Bike Center	0,93%	0	
Australia Bike Retailer	1,14%	0	
Beaumont Bikes	0,23%	0	
Bergeron Off-Roads	0,73%	0	
Bicycle Specialists	3,78%	0	
Bike Satellite Inc.	0,00%	0	
Bloomington Multisport	1,96%	0	
Burnett Road Warriors	1,96%	0	
Business Equipment Center	1,96%	0	
Capital Road Cycles	5,33%	0	
Carlson Specialties	3,60%	0	
Chicago City Saddles	1,80%	0	
Chicago Rent-All	3,48%	0	

Figure 5.10. ReturnRate KPI by Supplier

The table shows the ReturnRate KPI — the percentage of goods returned after receipt - for each supplier, compared against the target of 0%. The icon in the ReturnRate Status column indicates KPI performance: green represents suppliers who meet the KPI (ReturnRate = 0%), while yellow indicates those who do not (ReturnRate > 0%).

Notably:

- Bike Satellite is the only supplier with a green icon, indicating a return rate of 0%.
- Meanwhile, suppliers such as Anderson's Custom Bikes (5.48%) and Capital Road Cycles (5.33%) have significantly higher return rates and should be flagged for monitoring, contract review, or quality improvement requirements.

The ReturnRate KPI helps the Purchasing Department identify suppliers with high quality risks. It serves as a basis for decisions regarding continued partnerships, adjustments to vendor evaluation policies, or enhancements to incoming goods inspection processes.

Row Labels	ReturnRate	ReturnRate Goal	ReturnRate Status
⊞2010		0	●
⊞2011	8,16%	0	▲
⊟2012			
A. Datum Corporation		0	●
Advanced Bicycles	0,00%	0	●
Allenson Cycles	4,97%	0	▲
American Bicycles and Wheels	0,00%	0	●
American Bikes	38,30%	0	▲
Anderson's Custom Bikes	33,33%	0	▲
Aurora Bike Center	0,00%	0	●
Australia Bike Retailer	0,00%	0	●
Beaumont Bikes	0,00%	0	●
Bergeron Off-Roads	0,00%	0	●
Bicycle Specialists	0,00%	0	●
Bike Satellite Inc.	0,00%	0	●
Bloomington Multisport	0,00%	0	●
Burnett Road Warriors	33,33%	0	▲
Business Equipment Center	33,33%	0	▲
Capital Road Cycles	34,47%	0	▲

Figure 5.11. ReturnRate KPI by Supplier and by Year

The ReturnRate of each supplier is broken down by year, allowing the company to monitor the quality performance of incoming goods over time. The data reveals a significant spike in return rates in 2011 (8.16%), indicating a potential issue or decline in quality control during that period. In contrast, years like 2010 and 2012 show multiple suppliers maintaining a return rate of 0%, demonstrating high reliability.

This table supports the Purchasing Department in identifying high-risk periods, evaluating the consistency of each supplier, and making informed decisions on whether to maintain, improve, or replace suppliers based on actual performance data.

5.3. Introduction to the structure of the reporting system

5.3.1. Report highlights

- Cover Page
- Dashboard 1: Purchasing Overview
- Dashboard 2: Inventory Dashboard

- Dashboard 3: Vendor Performance
- Dashboard 4: Delivery Dashboard
- Dashboard 5: Employee Dashboard
- Dashboard 6: Product Dashboard
- Solution

5.3.2. Overall insight of dashboards

Using the ETL process, the team extracted and transformed data from the data warehouse to develop six interactive dashboards. These dashboards are designed to support business decision-making through targeted insights:

- *Dashboard 1: Purchasing Overview.* This dashboard provides a high-level snapshot of purchasing operations, including total orders, procurement expenses, and receipt completion rates. It offers stakeholders a quick grasp of purchasing performance without delving into specific suppliers or products.
- *Dashboard 2: Inventory Dashboard.* Presents both summary and detailed views of current inventory levels, product stock statuses, and reorder needs. This aids in inventory optimization and strategic purchasing decisions.
- *Dashboard 3: Vendor Performance.* Offers a comprehensive evaluation of suppliers based on order fulfillment, delivery times, and cost effectiveness. This enables supplier performance tracking and relationship optimization.
- *Dashboard 4: Delivery Dashboard.* Focuses on logistics efficiency, showcasing delivery lead times, shipment statuses, and any delivery-related issues. It supports continuous improvement in distribution processes.

- *Dashboard 5: Employee Dashboard.* Highlights key employee metrics related to purchasing and logistics operations. This includes staff performance (e.g., number of orders processed), workload distribution, approval times, and involvement in procurement or inventory tasks. The dashboard helps management assess team effectiveness, identify training needs, and ensure resource allocation aligns with operational demands.

- *Dashboard 6: Product Dashboard.* Displays product-level data, such as purchase volume, cost trends, and inventory turnover rates. This facilitates effective portfolio management and demand planning.

5.4. Analyze and visualize data with Power BI

5.4.1. Transform and load data

After the data is prepared from the Data Warehouse, the next step is to transform and load it into Power BI to support data visualization and analysis. The team chose to use Import mode to retrieve data from the database instead of Direct Query. With Import mode, the data is stored locally within Power BI, and preloading it significantly improves performance, providing users with faster and smoother report interactions, while also enabling offline analysis without relying on a direct connection to the data source.

Unlike Direct Query, which requires a continuous connection to the data source—resulting in query latency and dependence on the performance of the underlying database system—Import mode helps reduce latency and lessen the load on the data source. This ensures that the analysis process is not interrupted and is especially beneficial in scenarios where internet connectivity is unstable or limited.

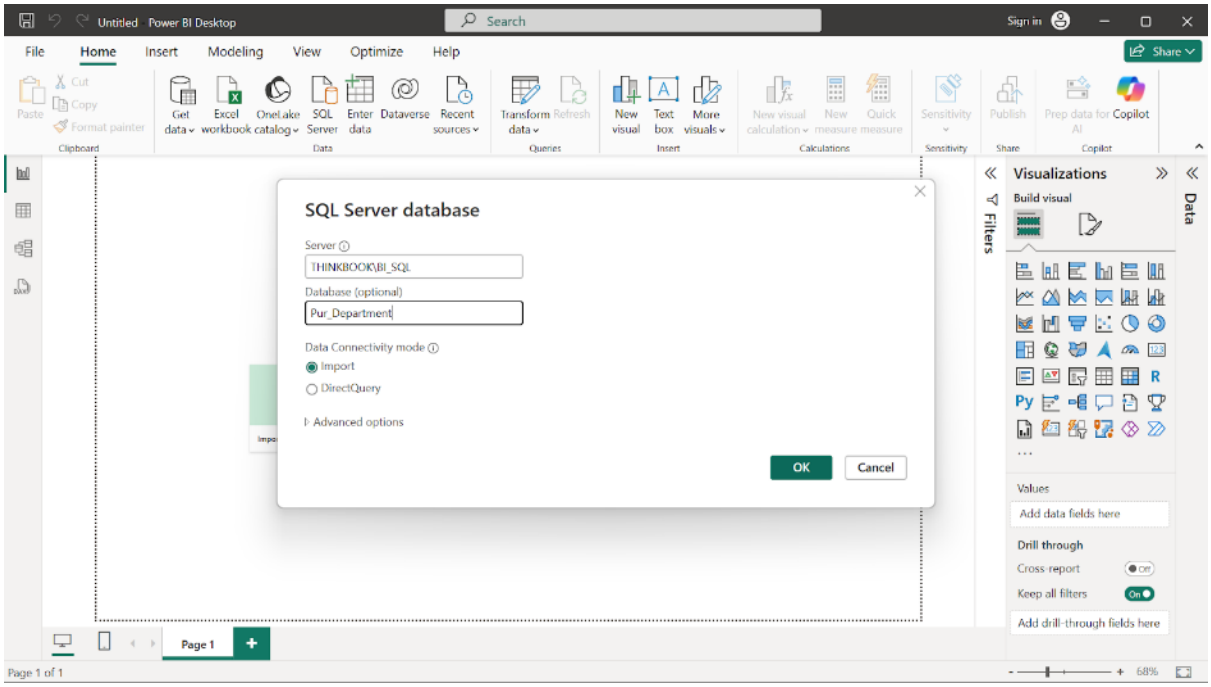


Figure 5.12: Import SQL Server database

After loading the data using Import mode, the team performed several data transformation steps before proceeding with analysis. Key transformation tasks included:

- Merging columns to better align with analytical objectives. For example, combining the FirstName, MiddleName, and LastName columns into a single FullName column.
- Creating new tables and measures to support more in-depth analysis.
- Generating calculated columns such as Line Total, Lead Time, Received Rate, and Return Rate to facilitate KPI measurement.
- Establishing relationships between tables to ensure the relational data model accurately reflects the underlying business structure.

This transformation and loading process plays a foundational role in ensuring the quality of the reporting system, guaranteeing that the data is complete, accurate, consistent, and ready for analysis. As a result, Power BI can operate at its full potential in supporting data-driven analysis and decision-making.

5.4.2. Creating model relationship

Figure 5.13 illustrates the data model designed in Power BI, which follows a galaxy schema structure. At the center of the model is the FactPurchaseOrders table, which serves as the main fact table. Surrounding it are six dimension tables: DimProduct, DimVendor, DimShipMethod, DimEmployee, and DimTime. Each of these dimension tables is connected to the fact table through one-to-many relationships, where the dimension table represents the "one" side and the FactPurchaseOrders table represents the "many" side, enabling efficient filtering and slicing across different analytical perspectives.

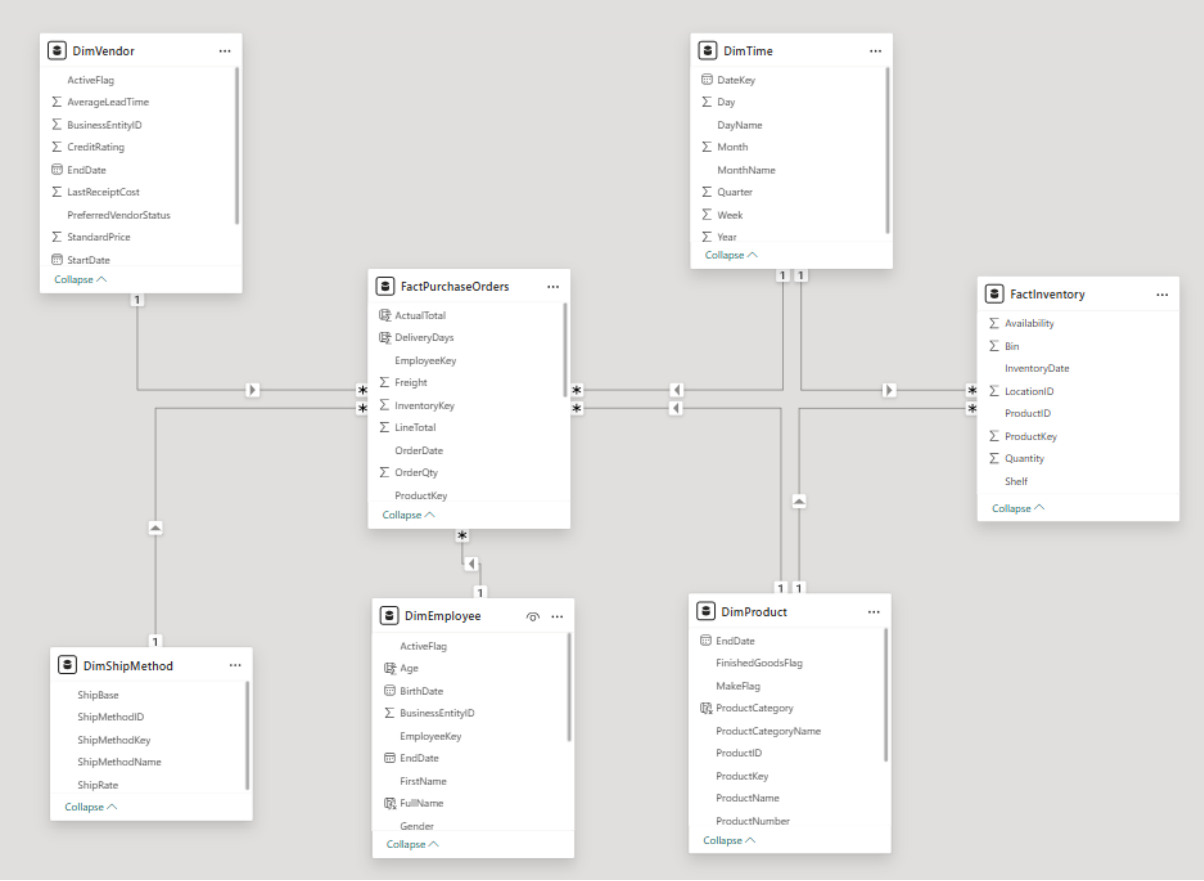


Figure 5.13. Model relationship

5.4.3. Business analytics

5.4.3.1. Home page

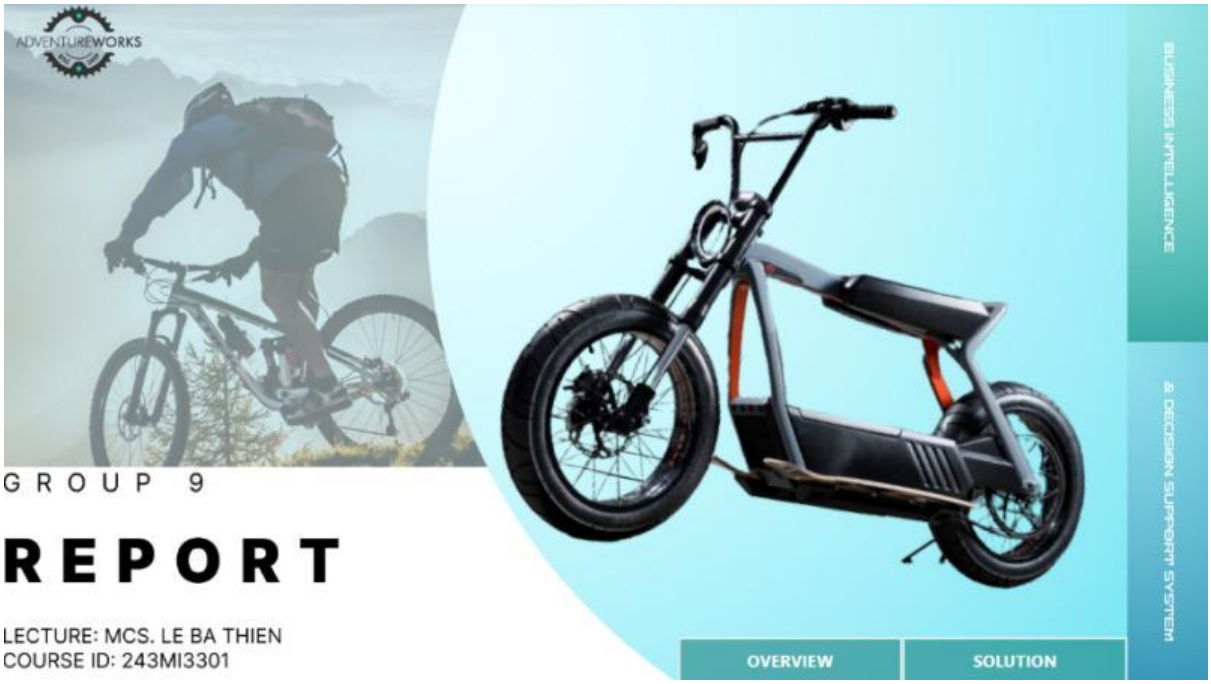


Figure 5.14. Home Page

5.4.3.2. Overview Dashboard

Figure 5.15. Overview Dashboard

The Purchasing Overview dashboard provides a holistic view of the purchasing performance at AdventureWorks over the years 2011 to 2014. The dashboard is structured into key sections that highlight core procurement metrics such as total purchase orders, total spending, rejection rates, delivery timeliness, and average unit price. It further supports decision-making by presenting visualizations and detailed tables on employee activity, vendor performance, product distribution, and temporal trends.

Key Metrics and Insights

The total number of purchase orders recorded is 8,845, which reflects a high level of procurement activity throughout the observed period. The total purchasing expenditure amounts to $63.79 million, indicating the organization's significant investment in raw materials and components. With an average unit price of $34.743, cost management appears to be stable. The rejection rate of 3.12%—while not alarming—suggests potential for improvement in quality control or supplier compliance. Notably, the on-time delivery rate is exceptionally high at 99.88%, emphasizing strong logistical coordination and vendor reliability.

Visual Charts and Tables

The dashboard incorporates several visual elements to illustrate procurement trends and performance distributions. A bar chart detailing Purchase Amount by Employee shows that procurement responsibilities are spread across key individuals, with Mikael Sandberg and Linda Meisner leading in total purchase value. This breakdown can help assess employee workload and procurement accountability.

The Purchase Amount by Time line chart tracks monthly spending from 2011 to 2014. Peaks observed in mid-year months (June–August) indicate seasonal procurement patterns, likely aligned with production cycles or promotional periods. This data is instrumental for strategic budgeting and planning.

The Product Category donut chart visualizes purchase distribution by product type. Components make up the largest share (41.47%), followed by Clothing (26.59%)

and Accessories (19.2%). This insight emphasizes which categories require more rigorous supplier monitoring or cost-saving initiatives.

The Vendor Overview table lists suppliers along with metrics such as total received quantity, purchase amount, rejection rate, and vendor performance score. Vendors like American Bikes and Allenson Cycles stand out in both volume and value, but varying rejection rates (some over 3%) highlight inconsistencies in product quality or delivery standards.

The Purchase Orders by Year matrix presents product-level order history over the four-year period. Items such as Chainring, Classic Vest, and Cable Lock exhibit consistent ordering patterns, suggesting they are high-demand or frequently restocked products. Meanwhile, the Purchase Orders by Status chart reveals that the majority of orders (8.17K) are successfully processed, affirming overall process efficiency.

Objectives and Recommendations

Short-Term Objectives:

- Enhance order fulfillment by reducing the rejection rate. This involves working closely with suppliers to ensure product quality and accurate delivery.
- Implement cost optimization strategies during peak months. Analyzing high-expenditure periods may uncover opportunities for price negotiations or bulk purchase discounts.

Long-Term Objectives:

- Strengthen partnerships with high-performing suppliers who demonstrate timely and accurate deliveries. Simultaneously, contracts with underperforming vendors should be revisited.
- Leverage historical purchasing data for strategic planning. Forecasting future demand, managing inventory proactively, and preparing for

seasonal spikes will contribute to more efficient operations and improved cost control.

5.4.3.3. Inventory Dashboard

Figure 5.16. Inventory Dashboard

Key Metrics and Insights

The total inventory quantity currently stands at 307,000 units, reflecting the organization's overall stock volume across all locations. Out of this, only 50,450 units are immediately available for use or sale, which may imply that a significant portion of stock is either reserved, in transit, or awaiting processing. The company manages 405 distinct products, a testament to the breadth of its offerings. Notably, only 4 products are out of stock, indicating a well-maintained supply chain with minimal service risk.

A key performance indicator featured in the dashboard is the Order Accuracy Rate, which tracks fulfillment reliability over time. From 2011 to 2014, this metric shows a consistent upward trend—from approximately 0.89 in 2011 to 0.946 in 2014—signifying improvements in inventory handling, order processing, and supplier coordination.

Visual Charts and Tables

The Order Accuracy Rate line chart illustrates a steady improvement in operational precision over the four-year span. Beginning below 0.89 in 2011 and rising to approximately 0.946 by 2014, this upward trend reflects ongoing enhancements in order fulfillment processes. Such progress may be attributed to better coordination between departments, improved inventory tracking systems, or stricter quality control mechanisms.

Complementing this is the Value by Category donut chart, which provides a clear picture of how inventory value is distributed across product lines. Components dominates the inventory value, accounting for 41.71%, followed by Others (34.97%), Accessories (19.94%), and Clothing (3.38%). This distribution indicates that strategic planning, budgeting, and risk mitigation efforts should prioritize the highest-value categories—particularly Clothing, given its substantial financial impact on overall inventory.

The Products table offers detailed insights into stock levels across individual products. Notably, items such as Hex Nut 7 (1911 units), Hex Nut 23 (1901 units), and Thin-Jam Hex Nut 3 (1901 units) appear to be highly stocked, which may reflect fast-moving items, core components, or deliberate buffer stock policies. Conversely, other items show significantly lower quantities, suggesting potential reorder requirements, limited demand, or recent stock depletion. This product-level visibility helps guide procurement and replenishment decisions.

Lastly, the Quantity by Shelf treemap visualizes the physical layout and concentration of inventory within warehouse space. Shelf labels such as "N/A", "G", "E", and "A" hold substantial stock volumes, while a large number of shelf identifiers contain little to no inventory. This uneven distribution highlights opportunities to optimize shelf utilization, streamline picking paths, and potentially redesign warehouse layout to improve operational efficiency and space management.

Objectives and Recommendations

Short-Term Objectives:

- Resolve the stockout of the 4 unavailable items to maintain product continuity.
- Redistribute items more evenly across shelves to balance space utilization and improve accessibility in warehouse operations.

Long-Term Objectives:

- Continue tracking and improving the Order Accuracy Rate by investing in better inventory control systems and employee training.
- Focus inventory monitoring efforts on high-value categories such as Clothing and Components, ensuring optimal stock levels and turnover.
- Conduct periodic audits of shelf usage and storage layout to streamline warehouse capacity and reduce underutilized areas.

5.4.3.4. Vendor Dashboard

Figure 5.17. Vendor Dashboard Overview

The Vendor Dashboard is a critical element of Adventure Works Cycles' purchasing analytics system. It provides a comprehensive overview of supplier performance, delivery quality, and procurement spending across the vendor network. By combining key performance indicators with intuitive visualizations, this dashboard enables the purchasing team to assess vendor reliability and efficiency for strategic supply chain decisions.

Users can explore the data through interactive filters such as year, unit of measure, and currency. These tools facilitate both high-level insights and detailed drill-downs, helping stakeholders monitor ordering trends, evaluate delivery reliability, track rejection rates, and analyze cost variations by vendor.

Key Metrics and Analysis

- Total Vendors: 104. Indicates the size of the current supplier network.
- Average Credit Rating: 1.36. Reflects the average financial trustworthiness and payment terms of vendors.
- Average Lead Time: 18.66 days. Measures how quickly vendors fulfill orders — shorter times indicate better supply responsiveness.
- Rejected Rate per Vendor: 3.12%. Represents the percentage of returned or rejected orders, typically due to quality or delivery issues.
- Total Orders: 4,012. Denotes total purchase activity over the reporting period.

Visual Insights

- Orders by Vendor (Horizontal Bar Chart): Displays order volume from each key vendor.
- Vendor Credit Rating Distribution (Donut Chart): Shows the proportion of vendors by credit rating, offering insight into financial risk.

- Order Quantity and Lead Time Trends (Area + Line Chart): Tracks how order volumes and lead times have changed over time.
- Top 10 Vendors by Received Quantity (Combo Column & Line Chart): Highlights high-volume suppliers alongside their rejection rates to assess consistency.
- Last Receipt Cost per Vendor by Year (Line Chart): Helps identify pricing trends for better negotiation and cost planning.

Objectives

Short-term:

- Lower rejection rates by improving inbound quality control.
- Prioritize fast and price-stable suppliers to ensure uninterrupted operations.

Long-term:

- Conduct regular supplier assessments based on credit score, rejection rate, and lead time.
- Optimize the vendor pool by removing underperformers and renegotiating with strategic partners.
- Use historical data to enhance procurement planning and match production needs.

Role-Based Recommendations

- Purchasing Manager: Should focus on delivery performance of high-volume vendors and track lead time variability to spot risks.
- Buyers: Must monitor high rejection vendors closely and prioritize reliable partners with steady pricing and short lead times.

- Purchasing Assistant: Should utilize dashboard filters to quickly detect outliers like cost spikes or rising defect rates and ensure supplier data is consistently maintained.

5.4.3.5. Delivery Dashboard

Figure 5.18. Delivery Dashboard

The Delivery Dashboard provides an overview and detailed insights into the purchasing and shipping operations of the business. Thanks to key indicators such as total number of orders, total revenue, number of shipping methods, and average shipping cost, this dashboard supports the management team in easily monitoring and evaluating the overall efficiency of the entire delivery process.

Overall Performance

Based on aggregated data from the "Group 9 Purchasing Report" system, the company's shipping system is currently operating with 5 main shipping methods. The total recorded shipping cost is 1,583,978.23 USD, with an average delivery time of 9.1 days, reflecting a relatively stable level of logistics activity. The Line Total revenue reached 63,791,994.84 USD, showing a large scale of operations and high transaction volume.

Analysis by Shipping Method

The chart “Line Total by Ship Method” shows that CARGO TRANSPORT 5 holds the dominant position, contributing 45.83% of total revenue, followed by ZY – EXPRESS (21.1%) and OVERNIGHT J-FAST (16.97%). This indicates that CARGO TRANSPORT 5 is currently the main pillar in the company's shipping network.

The distribution of orders by shipping method shows that CARGO TRANSPORT 5 also leads in order volume with approximately 4,000 orders, followed by OVERNIGHT J-FAST with around 1,500 orders and ZY – EXPRESS with about 1,100 orders. The consistency between revenue and usage frequency reflects the reliability and popularity of these methods in the supply chain.

The average shipping cost per order varies significantly across methods. Specifically:

- OVERSEAS – DELUXE has the highest cost, up to 1,534.10 USD/order.
- XRQ – TRUCK GROUND has the lowest cost, only 328.62 USD/order.

This difference suggests an opportunity to optimize costs by reconsidering the frequency of using expensive methods, especially in the context of increasing cost pressure.

The OVERSEAS – DELUXE method has the longest delivery time (11.7 days), while the other methods all remain under 10 days, showing a significant discrepancy in time efficiency among different shipping providers.

Trends and Performance Evaluation

The chart “Average Freight per Order by Year” shows that the average shipping cost of OVERSEAS – DELUXE increased sharply from 2012 to 2014. Meanwhile, other methods like CARGO TRANSPORT 5, XRQ – TRUCK GROUND, and OVERNIGHT J-FAST maintained stable or slightly declining costs over time. This trend reveals price instability and poor cost control from the OVERSEAS – DELUXE method, highlighting the need for reassessment and evaluation of its effectiveness.

The chart “Order Quantity by Year” shows a strong increase in the number of orders from 2011 to 2014, rising from about 0.9 million to nearly 1.5 million orders. This trend not only reflects market expansion and growing demand but also serves as a warning about the current shipping system’s capacity, requiring the company to plan for logistics capability expansion.

Recommendations

- Optimize transportation methods: CARGO TRANSPORT 5 plays a key role but is not cost-effective – renegotiation or route optimization is recommended. OVERSEAS – DELUXE has high cost and delivery time – it should be reserved for special orders only. XRQ – TRUCK GROUND offers low costs and should be utilized more if quality can be ensured.
- Control transportation costs: Monitor transportation expenses by year and by method, especially for OVERSEAS – DELUXE. Analyze reasons for cost increases and consider alternative suppliers with better pricing.
- Reduce delivery time: The current average delivery time of 9.1 days can be improved. Prioritize ZY – EXPRESS and OVERNIGHT J-FAST for urgent orders. Regularly assess on-time delivery rates.
- Expand logistics capacity: With a sharp rise in orders from 2011–2014, it's necessary to invest in warehouse expansion, implement technologies (ERP, tracking), and standardize processes to meet growing demand.
- Enhance monitoring tools: Introduce additional KPIs such as on-time delivery rate, defective orders, and transportation cost per 100 USD in revenue to comprehensively evaluate transportation performance.

5.4.3.6. Employee Dashboard

Figure 5.19. Employee Dashboard

Key Metrics and Insights

The procurement department comprises 12 employees, forming a moderately sized team responsible for managing vendor relationships and purchase orders. The gender ratio of 2.00 (M/F) reveals a male-dominant team structure, which could raise considerations for future gender diversity efforts. On average, each employee processes 334 purchase orders, with an average PO value of approximately $5.32 million, reflecting the substantial transactional responsibilities assigned to each member.

The role distribution within the team is skewed toward operational execution, with 75% designated as Buyers, 16.7% as Purchasing Assistants, and only 8.3% holding managerial positions. This suggests a lean oversight structure where Buyers must operate with high autonomy, requiring strong individual accountability.

From the performance perspective, Mikael Sandberg, Linda Meisner, and Eric Kurjan emerge as top contributors—each handling 360+ orders and managing over $6 million in purchases, with minimal rejection rates (0.01%–0.10%). In contrast, Sheela Word, the sole manager, processed only 160 orders and recorded a significantly high

rejection rate of 23.22%, indicating potential concerns around vendor selection or oversight execution. These variances highlight the need for individualized performance tracking beyond role hierarchy.

Visual Charts and Tables

The bar and line chart displaying number of orders versus rejection rate emphasizes inconsistencies in quality among employees. While most Buyers maintain rejection rates below 1%, Reinout Hillmann (12.77%) and Sheela Word (23.22%) stand out with considerably higher return rates, signaling quality control issues that warrant deeper investigation.

The Purchase Amount by Employee bar chart aligns closely with order volume, reinforcing a positive relationship between quantity and financial impact. However, some employees—such as Ben Miller and Arvind Rao—handle a comparable number of orders but with noticeably lower total purchase values, suggesting differences in deal size, vendor category, or pricing tiers.

The updated scatter plot labeled "Relationship between Purchase Value, Order Volume and Age by Employee and Tenure" offers a multidimensional view. Positioned by age and tenure, each bubble reflects individual purchasing performance. The cluster of mid-tenure employees (4–5 years) managing high order volumes and values reinforces the notion that mid-experience employees are the most productive segment.

The employee table at the bottom further supports this, combining tenure, age, vendor coverage, and rejection rate into a holistic view. Most high-performing Buyers fall in the 40–45 age range with 4–5 years of tenure, guiding both retention strategies and leadership grooming.

Objectives and Recommendations

Short-Term Objectives:

- Investigate causes behind high return rates in specific employees, especially in managerial and assistant roles.

- Provide targeted support or coaching for individuals with high order volumes but suboptimal quality scores.

Long-Term Objectives:

- Implement a performance dashboard combining order count, value, rejection rate, and vendor diversity per employee.
- Rebalance the team structure by increasing assistant or analyst support to Buyers.
- Introduce diversity and inclusion initiatives to address the current gender imbalance.

With a clearer understanding of individual performance patterns, the organization is well-positioned to enhance operational efficiency, reinforce accountability, and build a more data-driven and inclusive procurement function.

5.4.3.7. Product Dashboard

Figure 5.20. Product Dashboard

This dashboard, titled "Product", offers a detailed view of key metrics related to the purchasing activities of Adventure Works Cycles, with a specific focus on products. It helps purchasing personnel monitor product performance, track return trends, and analyze inventory levels efficiently. Users can interact with the report through filters such as year, unit, and currency to explore data from different perspectives. This flexibility allows purchasing teams to evaluate product effectiveness, supplier reliability, and order performance at both summary and detailed levels. To support strategic decision-making, insights and recommendations are provided for each of the following roles: Purchasing Manager, Buyer, and Purchasing Assistant.

Key Metrics and Insights

- Total Products Purchased: 265. Represents the number of distinct products that were purchased within the given period.
- Total Purchase Amount: $61,211.69K. Reflects the actual spending on product procurement.
- Total Quantity Purchased: 2,254.60K Indicates the overall number of units successfully acquired and stocked.
- Return Rate: 3.12%. Shows the percentage of purchased products that were returned, which may reflect issues in quality, packaging, or vendor performance.
- While the total quantity and cost of purchases are at a reasonable level, the return rate of 3.12% warrants further attention, especially in high-volume product categories.

Visual Insights

- Purchase Amount by Product Category. Components lead in terms of purchase value, followed by Accessories and Clothing. This indicates where the majority of procurement budgets are allocated.

- Top 10 Products by Order Quantity and Value. The chart shows which products dominate in terms of both volume and cost. HL Crankarm stands out as the most frequently ordered product with the highest associated cost.

- Return Rate by Product and Category. Products in the Components category exhibit the highest return rates, signaling the need for deeper investigation into potential defects or mismatches.

- Returned Quantity by Subcategory. Subcategories like *Pedals* and *Tires and Tubes* have the highest number of returned items, suggesting recurring quality or supplier issues.

- Return Rate and Inventory Quantity Over Time. A notable peak in return rate occurred in 2012 (9.59%), followed by a steady decline to 1.91% in 2014. Meanwhile, inventory quantities, particularly in Components, have grown — possibly indicating improved stock management but also a need to monitor overstocking.

Objectives

- Short Term: Improve the efficiency of product purchasing by reducing the frequency of returns to vendors due to defective items. Focus particularly on product categories like Components, where return rates are noticeably high. Strengthen inspection processes during receipt to detect quality issues early and reduce return processing time.

- Long Term: Maintain a high level of procurement accuracy while minimizing defective returns and the need for vendor compensation. Establish stronger vendor evaluation and quality assurance mechanisms to prevent recurring quality issues. Optimize inventory by aligning purchasing plans with historical return patterns and supplier reliability.

Recommendations by Role

- Purchasing Manager: The Purchasing Manager should regularly monitor net purchase trends to identify potential opportunities for cost reduction. It is also essential to assess fulfillment performance, especially in categories like *Components*, where low fulfillment rates may signal inefficiencies in the supply chain. Leveraging historical purchasing trends is critical for planning orders in advance of high-demand periods or anticipated price fluctuations, ensuring a proactive and stable procurement process.

- Buyers: For Buyers, enhancing quality control during the ordering process is vital to reduce the number of defective items returned to vendors. Maintaining high order accuracy is also necessary by carefully verifying order details before finalization. Additionally, Buyers should focus on replenishing products that are understocked or in high demand to maintain adequate inventory levels and avoid supply disruptions.

- Purchasing Assistant: Purchasing Assistants need to proactively monitor inventory levels to initiate timely reorders before stock reaches critical thresholds. Keeping purchasing records accurate and up to date—including order history and vendor performance—is crucial for supporting informed procurement decisions. Finally, they should use filtering tools to analyze past transactions and identify recurring patterns, helping improve the efficiency and effectiveness of future purchasing activities.

5.4.3.8. Solution

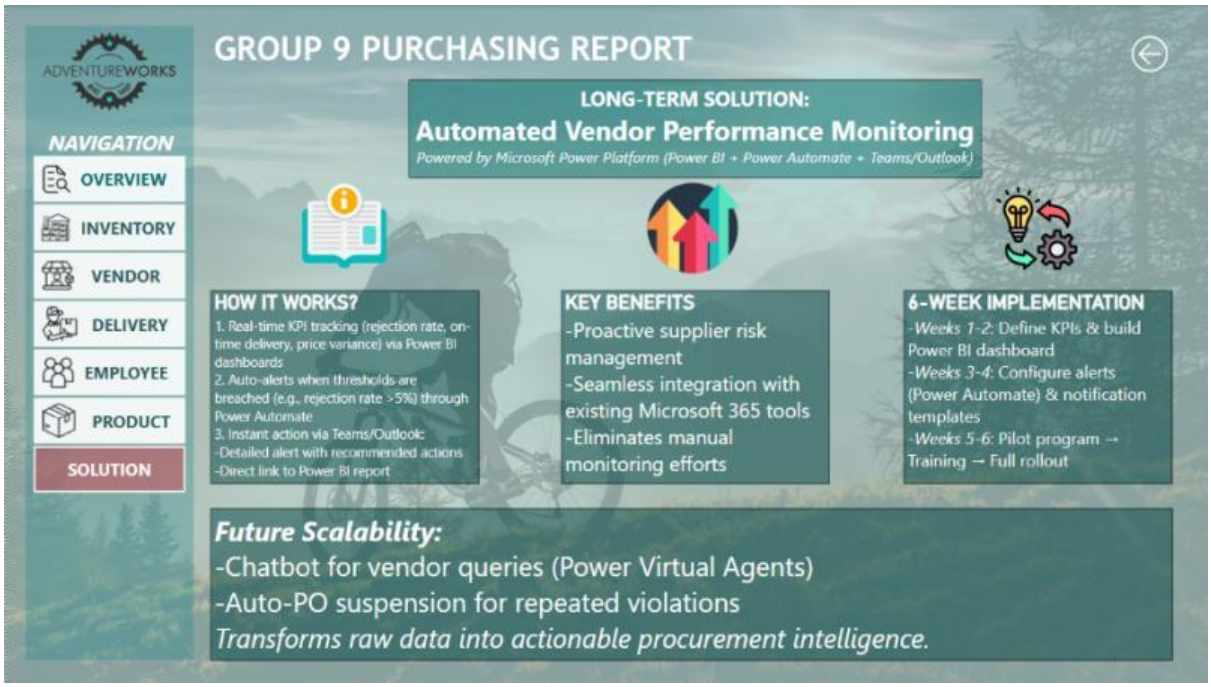


Figure 5.21. Solution Page

Long-Term Solution: Integrated Vendor Performance Workflow Using Microsoft Power Platform

To improve procurement effectiveness and ensure vendor performance is monitored and acted upon consistently, we propose implementing an integrated vendor performance workflow. This solution leverages existing enterprise tools—Power BI for analytics, Power Automate for alerts and workflows, and Microsoft Teams or Outlook for communication—to create a closed-loop system that supports timely, data-driven purchasing decisions.

The core of this solution is a live Vendor Performance Monitoring System. This system continuously evaluates suppliers using predefined KPIs, including rejection rate, delivery timeliness, and unit price variance. Power BI dashboards aggregate and visualize vendor performance data on a weekly basis. When any KPI exceeds a set threshold, Power Automate is triggered to send automated alerts to the procurement team. These alerts include a summary of the issue, links to relevant reports, and suggested actions, all delivered directly via Teams or email.

The workflow operates as follows. Data is pulled from existing sources such as SAP ERP, Excel, or SharePoint and updated daily. Power BI processes and visualizes this data in real time, highlighting key trends and anomalies. When an issue is detected—such as a vendor's rejection rate surpassing five percent—Power Automate generates an alert. This alert includes key details about the supplier, performance metrics, and direct links to the dashboard, enabling immediate review. Procurement staff receive this notification in Teams or Outlook and can take immediate action, such as contacting the supplier or flagging the issue for management. All follow-up actions can be tracked in a shared Excel file or Microsoft Planner.

For example, if Vendor X exceeds a rejection rate of 6.4% (above the 5% threshold), the system will automatically notify the responsible purchasing officer via Teams. The message includes the vendor’s name, a brief description of the issue, and buttons to open the report in Power BI or escalate to a procurement manager. This eliminates the need for manual report checking and ensures no critical issue goes unnoticed.

This integrated approach delivers significant value across four key areas. First, it enables real-time responses to supplier issues, preventing repeat problems. Second, it increases cross-tool visibility, ensuring insights reach stakeholders directly in the tools they use every day. Third, it enforces an action-oriented mindset, where every alert is tied to a response or escalation. Finally, it is easy to implement, leveraging familiar Microsoft 365 applications already in use by most businesses.

To deploy this system, we propose a six-week rollout plan:

- *Week 1:* Define KPIs, thresholds, and alert conditions in alignment with procurement goals.
- *Week 2:* Develop and configure the Power BI dashboard, integrating with existing data sources such as SAP or Excel.
- *Week 3:* Build and test Power Automate flows using sample data to validate logic and response behavior.

- *Week 4:* Design adaptive notification templates for Teams and Outlook, ensuring they are clear and actionable.
- *Week 5:* Pilot the system with two to three high-volume vendors, gather feedback, and refine as necessary.
- *Week 6:* Train the procurement team, finalize SOPs, and roll out the solution company-wide.

This solution is scalable and can be expanded in the future. For example, approval workflows can be added to pause vendor purchases automatically if thresholds are breached. A chatbot powered by Power Virtual Agents can allow staff to query vendor status by name. Additionally, Microsoft Forms can be used to collect post-delivery vendor feedback, enriching the performance dataset.

In summary, this is a practical, tool-supported, and action-driven solution that aligns closely with real-world procurement needs, minimizes risks, and turns Business Intelligence into real-time business impact.

5.5. Discuss and evaluate the results of data analysis and visualization

The project introduced a Business Intelligence (BI) solution designed to assist the purchasing department in monitoring and enhancing operational performance across several dimensions, including products, inventory, delivery timelines, and vendor reliability. Data was processed and organized within a dedicated data warehouse, where the central fact table, *FactPurchaseOrders*, served as the foundation. Based on this, a data cube was constructed using a galaxy schema to facilitate detailed analysis and flexible aggregation.

Reports and dashboards were developed using Power BI, offering dynamic visual representations of the data. The integration of Slicer tools allowed users to interact with the visualizations by applying filters—such as product categories, specific vendors, or delivery periods—tailoring the analysis to their specific needs.

The analytical results revealed that the BI system provided clear insights into key purchasing metrics, such as inventory fluctuations, supplier performance, and order efficiency. This enabled users to identify emerging trends, make timely adjustments to inventory levels to prevent shortages or excess, and make data-driven decisions when selecting suppliers based on historical performance.

In conclusion, the implemented BI solution proved to be a valuable asset in streamlining procurement operations and supporting strategic decision-making, ultimately contributing to improved efficiency and overall organizational growth.

5.6. Cloud-Based Deployment Model

During the deployment phase of the Business Intelligence system, the team connected SQL Server to Power BI Desktop using the Import mode, in which data is loaded into the report file and synchronized on a scheduled basis.

To ensure secure and automatic data updates between the on-premises environment and Power BI Service, an On-premises Data Gateway was configured, acting as a secure bridge between internal databases and cloud reports. Additionally, Schedule Refresh was set up to update data at regular intervals.

This Import-based approach provides stability and is suitable for the current data size. In later stages, the team may consider switching to DirectQuery to enable real-time data interaction without relying on scheduled updates.

Steps to Deploy Reports from Power BI Desktop to Power BI Service:

Installing and Activating the On-premises Data Gateway

The gateway was installed on the machine hosting the SQL Server database. It was configured as a Personal Gateway, allowing Power BI Service to access the internal source whenever the host is online.

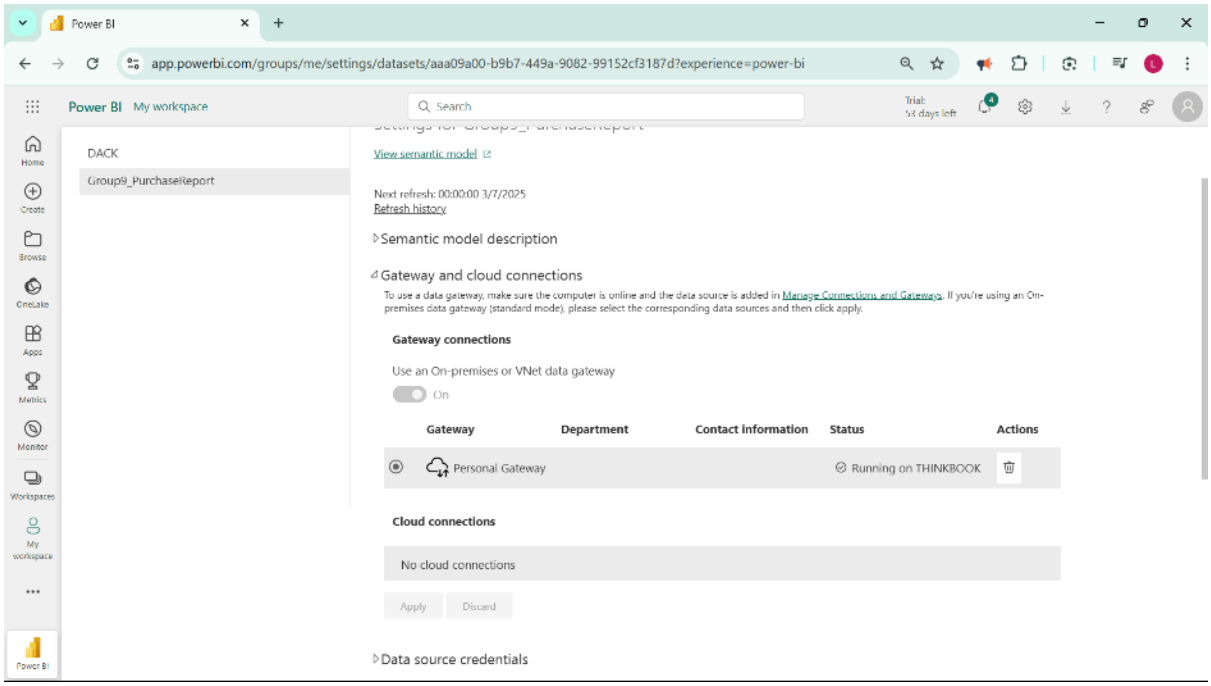


Figure 5.22. Activating a Personal Gateway in Power BI Service

Publishing Reports from Power BI Desktop

Once the report was finalized in Power BI Desktop, it was published to the designated workspace in Power BI Service, making it accessible for cloud-based collaboration and dashboards.

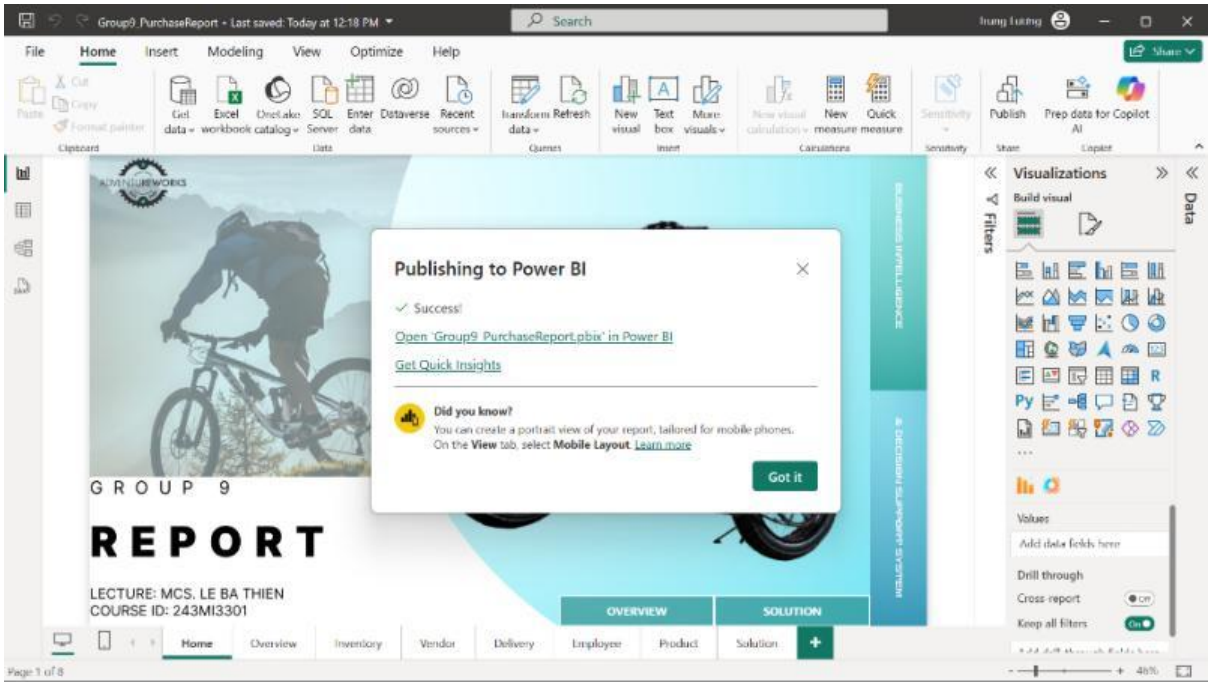


Figure 5.23. Successful publishing of a report from Power BI Desktop to Power BI Service

Configuring the Data Source within the Gateway

In Power BI Service, the team used the "Manage Gateways" section to register a new data source. The configuration included:

- Server and database names
- Authentication type (Basic or Windows)
- User credentials

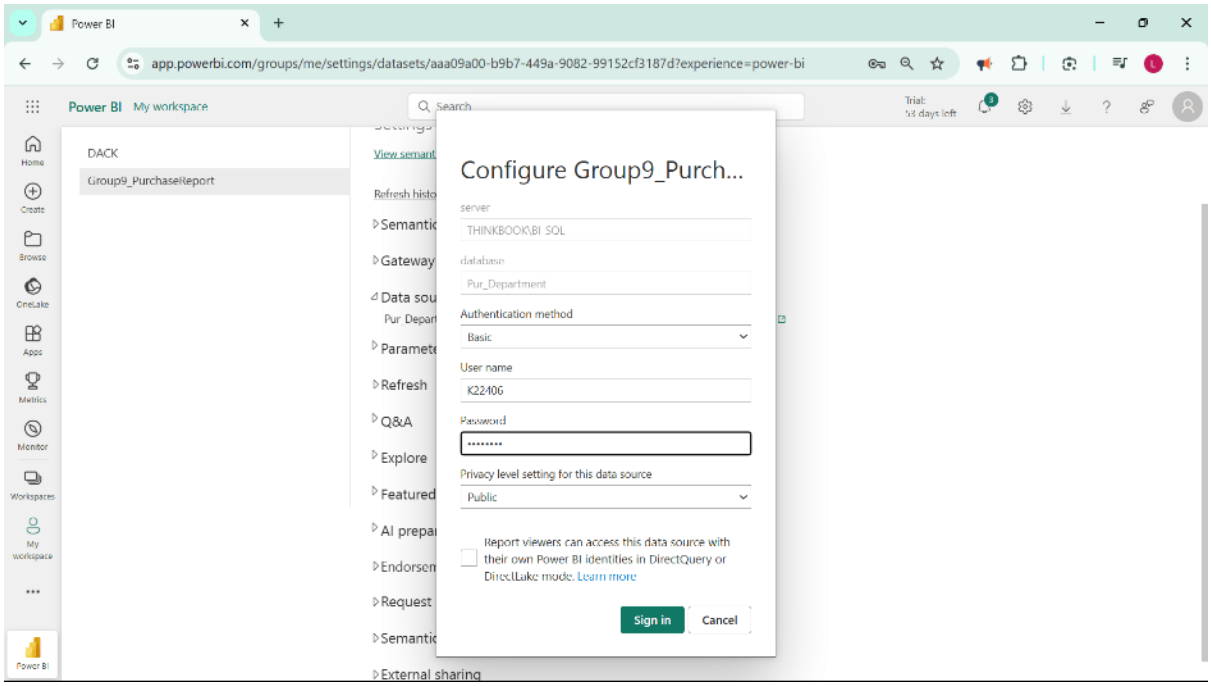


Figure 5.24. Data source configuration in the On-premises Gateway

Setting up Scheduled Refresh

Since the report is using Import mode, a Schedule Refresh was configured to automatically update the dataset daily. Users can also receive email alerts in case the refresh process fails.

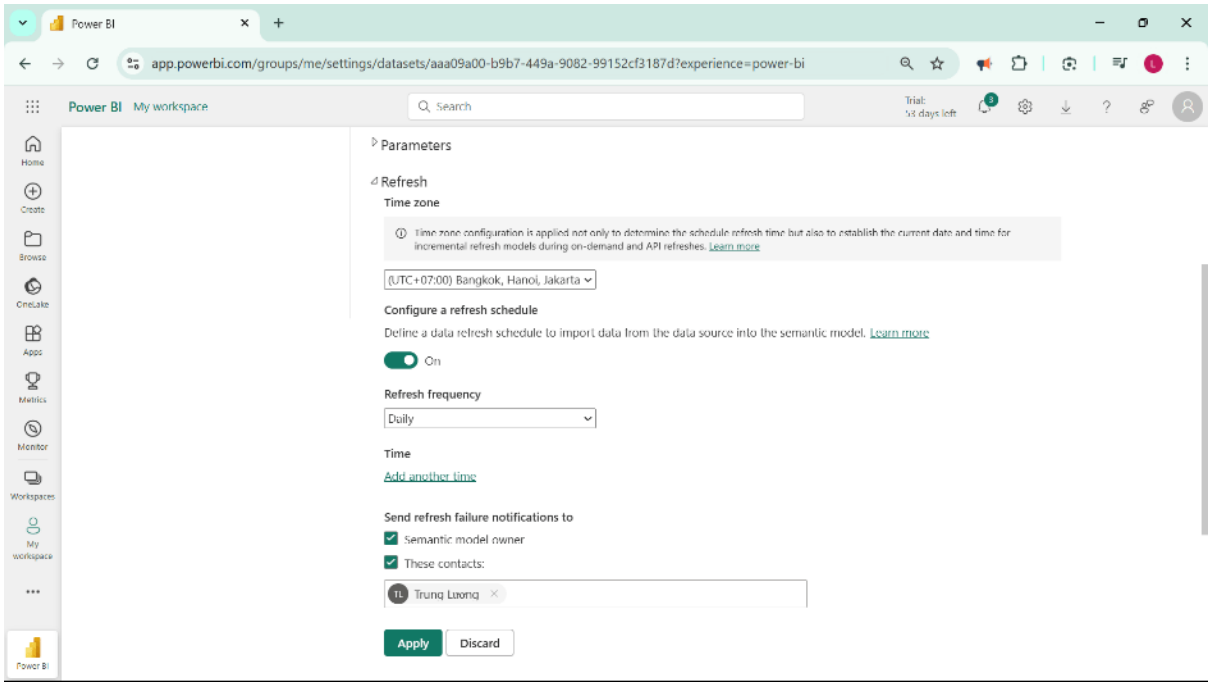


Figure 5.25. Schedule Refresh configuration in Power BI Service

CHAPTER 6. CONCLUSION

6.1. Summary of findings

This project focused on developing a Business Intelligence (BI) system for the Purchasing Department at AdventureWorks, aiming to leverage data from the purchasing database to improve procurement strategies and optimize operations. The results revealed valuable insights into procurement activities. A galaxy schema model was applied, with the FactPurchaseOrders table at the center, enabling efficient data aggregation and analysis. Additionally, the implementation of a data cube allowed for multidimensional analysis, providing the purchasing team with a comprehensive view of operational performance.

Key performance indicators such as Received Rate, Return Rate, and Order Quantity were defined to assess purchasing efficiency. Power BI was utilized to create interactive dashboards that visually presented inventory status, vendor quality, delivery performance, and other essential procurement metrics, thereby supporting quicker and more accurate decision-making. Finally, the study emphasized that to maximize the effectiveness of the BI system, it is crucial to provide training and support for users—particularly purchasing staff—to ensure proper usage and data interpretation.

6.2. Limitations

Although the implementation of the Business Intelligence (BI) system produced promising initial results, the research process revealed several key limitations. First and foremost, the project was conducted using simulated data from the AdventureWorks sample database rather than real-world business data. This limitation reduces the representativeness of the analysis and may affect the accuracy and practical relevance of the insights generated, particularly when applied to real operational contexts.

Additionally, the study was restricted to the purchasing department, which limited the scope of evaluating how BI tools could be leveraged across other functional areas within AdventureWorks. Another significant challenge was the lack of experience and technical knowledge among some team members. Insufficient familiarity with data

analysis principles and BI tools—such as SQL Server and Power BI—led to reduced efficiency and extended the time required for system development. Finally, the system's dependency on specific technologies may limit its flexibility and scalability, especially when adapting to evolving organizational requirements or emerging technological trends.

6.3. Future works

To advance beyond current limitations and amplify the impact of our findings, we propose a multi-faceted strategy for future development. First, integrating advanced analytics—specifically machine learning (both supervised and unsupervised models) and predictive analytics—will transform raw data into actionable intelligence. This shift will enable precise forecasting of purchasing trends, multidimensional evaluation of supplier risk/performance, and real-time anomaly detection in supply chain operations. Consequently, organizations can transition from reactive to predictive decision-making, optimizing inventory, reducing costs, and strengthening negotiation outcomes.

Furthermore, expanding the BI ecosystem to integrate seamlessly with core enterprise systems is critical. By bridging data silos between ERP (financials, inventory, procurement), CRM (sales pipelines, customer interactions), and external sources like market intelligence platforms, we create a unified 360-degree view of organizational performance. This holistic perspective reveals cross-functional insights—such as correlating procurement strategies with revenue impact—and drives alignment between operational efficiency and customer-centric outcomes.

Concurrently, establishing a structured user feedback mechanism ensures continuous solution refinement. Implementing digital feedback portals, periodic surveys, functional workshops, and power-user programs will capture granular insights into usability gaps and emerging needs. By systematically prioritizing and incorporating this input into development cycles, the BI solution evolves as a dynamic asset that remains tightly aligned with evolving user workflows, thereby maximizing adoption and ROI.

Finally, modernizing our data architecture via Microsoft Fabric will replace legacy SSIS/SSAS infrastructures with a fully cloud-native paradigm. Fabric's integrated environment—featuring OneLake unified storage, serverless compute elasticity, and native Power BI interoperability—delivers transformative scalability and performance. Vectorized processing accelerates large-scale analytics, while automated resource scaling and pay-as-you-go economics reduce operational burdens. This transition future-proofs our analytics foundation, enhancing agility, governance, and cost-efficiency while unlocking advanced AI/ML capabilities across the data lifecycle.

Collectively, these initiatives will not only resolve existing constraints but also position the organization at the forefront of data-driven innovation, fostering a culture where strategic decisions are informed by integrated, real-time intelligence.

6.4. Conclusion

The project to build a Business Intelligence (BI) system for the Purchasing Department of AdventureWorks has achieved many positive outcomes, clearly demonstrated through the successful implementation of a data warehouse architecture and the effective application of analytical and visualization tools. Specifically, the system has supported the aggregation and analysis of data from multiple sources, facilitating the monitoring of purchasing performance, supplier evaluation, and inventory management. The development of appropriate Key Performance Indicators (KPIs) has also enabled the management team to make more accurate and timely decisions. Additionally, the analytical process was systematically implemented—from data extraction, transformation, and loading (ETL) to OLAP-based data exploration and the presentation of results through dynamic reports—contributing to overall operational efficiency.

However, the project still has certain limitations. The quality of input data in some tables remains inadequate, affecting the accuracy of analyses. Moreover, the system has so far only been deployed within the Purchasing Department, without integration into related departments such as Sales, Production, or Finance. Some advanced analytical features have not been fully developed due to time and resource

constraints. Nevertheless, the results achieved have laid a solid foundation for the further development of a comprehensive BI system.

In the future, by expanding the BI system across the entire organization and continuing to address current challenges, AdventureWorks can fully leverage its data assets to enhance competitiveness and achieve sustainable growth in the dynamic bicycle industry. This project highlights the significant potential of BI in optimizing business processes and improving decision-making quality in today's modern business environment.

REFERENCES

1. Intelligence, M. (n.d.-b). *Bicycle market*. https://www.mordorintelligence.com/industry-reports/bicycle-market

2. MashaMSFT. (n.d.). *AdventureWorks sample databases - SQL Server*. Microsoft Learn. https://learn.microsoft.com/en-us/sql/samples/adventureworks-install-configure?view=sql-server-ver17&tabs=ssms

3. Oracle. (2023, June 8). *What is a data warehouse?* Oracle Vietnam. https://www.oracle.com/vn/database/what-is-a-data-warehouse/

4. Twin, A. (2025, June 15). *KPIs: What are key performance indicators? types and examples*. Investopedia. https://www.investopedia.com/terms/k/kpi.asp

5. Wikipedia contributors. (2025, April 7). *Performance indicator*. Wikipedia. https://en.wikipedia.org/wiki/Performance_indicator

6. *Types of business intelligence*. (n.d.). https://www.tutorialspoint.com/business-intelligence/types-of-business-intelligence.htm

7. *AdventureWorks*. (n.d.). https://dataedo.com/samples/html/AdventureWorks/

8. Dastaffo. (n.d.). *Training for Power BI*. Microsoft Learn. https://learn.microsoft.com/en-us/training/powerplatform/power-bi

9. Devart. (n.d.). *Devart SSIS Data Flow Components for cloud and database integration*. Devart Software. https://www.devart.com/ssis/what-is-ssis.html

10. Tutorial Gateway. (2025, January 26). *SQL Server Analysis Services (SSAS) tutorial*. https://www.tutorialgateway.org/ssas/

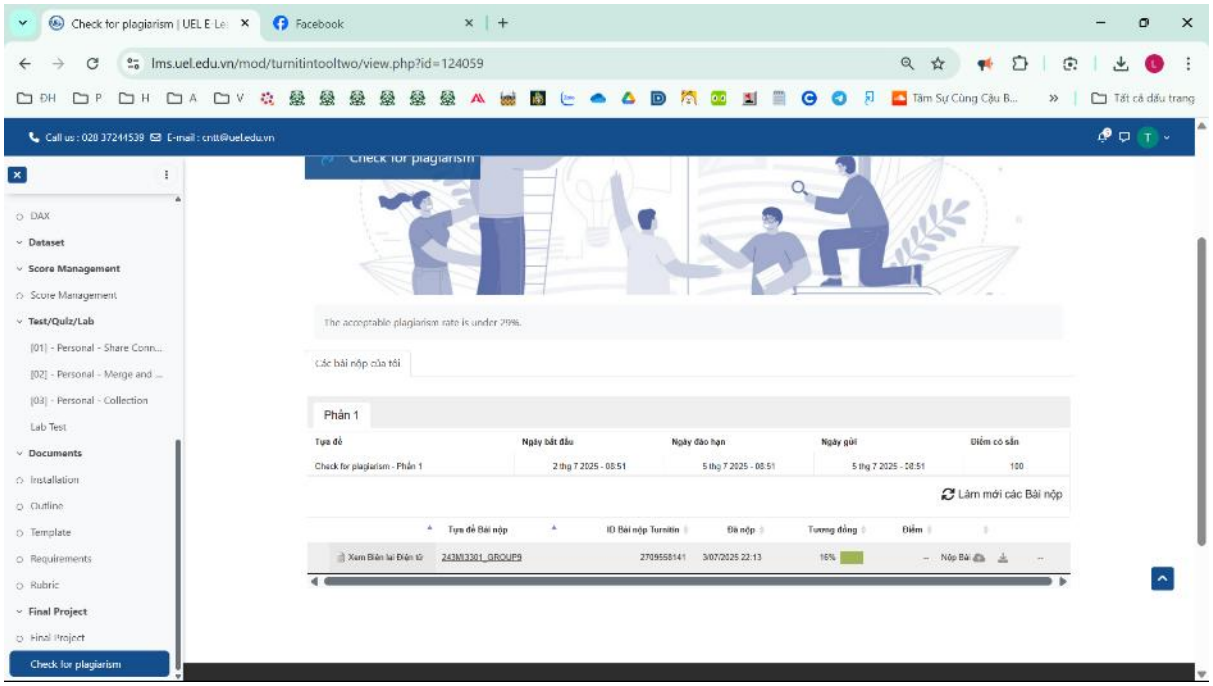
11. Cofsky, A. (2019, January 3). Connecting to datasets in the Power BI service from Desktop. *Blog di Microsoft Power BI | Microsoft Power BI*. https://powerbi.microsoft.com/it-ch/blog/connecting-to-datasets-in-the-power-bi-service-from-desktop/

APPENDIX: GROUP WORK ACTIVITY REPORT

No.	CONTENT	TASK	MAIN RESPONSIBLE	EVALUATION
1	OVERVIEW OF THESIS	Background	Trung Hieu	100%
		Project Objectives	Chi Trung	100%
		Research Subjects and Scope	Quy Hai	100%
		Technology Used		
		Research Significance		
		Report Structure	Chi Trung	100%
2	THEORETICAL BASIS	Overview of BI	Duc Luong	100%
		Overview of ETL		
		Data Warehouse and Data Mart	Van Tai	100%
		KPIs	Quy Hai	100%
		MDX Language and OLAP Techniques		
		Power BI	Trung Hieu	100%
3	USER REQUIREMENTS ANALYSIS &	Purchasing Department	Duc Luong	100%
		KPI Indicators	Quy Hai	100%

	DATA DESCRIPTION	BI System Requirements	Trung Hieu	100%
		Data Preparation	Chi Trung, Van Tai	100%
		Data Warehouse Design	Quy Hai, Duc Luong	100%
		ETL Process	All Team	100%
4	**DATA ANALYSIS AND VISUALIZATION**	Data Analytics with SSAS Technology	Van Tai	100%
		KPI System Development	Quy Hai	100%
		Introduction to Reporting System Structure	Duc Luong	100%
		Data Analysis and Visualization with Power BI	All Team	100%
		Discussion and Evaluation of Data Results	Trung Hieu	100%
		Report Implementation		
5	**CONCLUSION**	Summary of Findings	Quy Hai	100%

		Limitations	Trung Hieu	100%
		Future Works	Duc Luong	100%
		Conclusion	Chi Trung	100%
6	**REPORT COMPILATION AND EDITING**		All Team	100%
7	**PRESENTATION SLIDES**			


Check for plagiarism | UEL E Le
Facebook
lms.uel.edu.vn/mod/turnitintooltwo/view.php?id=124059
Call us : 028 37244539 E-mail : cntt@uel.edu.vn
DAX
Dataset
Score Management
Score Management
Test/Quiz/Lab
[01] - Personal - Share Conn...
[02] - Personal - Merge and ...
[03] - Personal - Collection
Lab Test
Documents
Installation
Outline
Template
Requirements
Rubric
Final Project
Final Project
Check for plagiarism
The acceptable plagiarism rate is under 29%.
Các bài nộp của tôi
Phần 1
Tựa đề
Ngày bắt đầu
Ngày đáo hạn
Ngày gửi
Điểm có sẵn
Check for plagiarism - Phần 1
2 thg 7 2025 - 08:51
5 thg 7 2025 - 08:51
5 thg 7 2025 - 08:51
100
Làm mới các Bài nộp
Tựa đề Bài nộp
ID Bài nộp Turnitin
Đã nộp
Tương đồng
Điểm
Xem Bản sao Điểm tô
243M3301_GROUP9
2709558141
3/07/2025 22:13
16%
Nộp Bài