UNIVERSITY OF ECONOMICS AND LAW

FACULTY OF INFORMATION SYSTEMS

FINAL PROJECT REPORT

BUSINESS INTELLIGENCE AND DECISION SUPPORT SYSTEM

TOPIC: BUILDING A BI SOLUTION TO SUPPORT MEASURING EMPLOYEE SALES PERFORMANCE

Lecturer: Msc. Le Ba Thien

Group: Wabisabi

Ho Chi Minh City, March 2024

Members of Group Wabisabi

NO.	Full name	Student ID	Point / 10 (Individual Contribution)	Signature
1				
2				
3				
4				
5	Lục Minh Phú	K214161342	10	

Acknowledgements

To begin with, we would like to extend our deepest appreciation to all the professors in the Department of Management Information Systems for imparting the knowledge, skills, and information crucial to the success of this project. Furthermore, we are profoundly grateful to our mentor, Mr. Le Ba Thien, for providing us the opportunity to pursue this research and for his invaluable guidance throughout the entire process.

Ho Chi Minh City, May 2024

Group WabiSabi

Commitment

We affirm that the results presented in this thesis are entirely our own work and have not been copied from any other source. The content of the project is either original or derived from various sources, with all references duly cited and quoted in accordance with regulations. We accept full responsibility and are prepared to face any disciplinary actions if any violations are discovered.

Ho Chi Minh City, May 2024

Group WabiSabi

Table of Contents

List of Tables

List of Figures

List of Algorithms

List of Acronyms

BI	Business Intelligence
PM	Project Management
CIO	Chief information officer
RAD	Rapid application development
UAT	User acceptance testing
WBS	Work Breakdown Structure
IT	Information Technology
POC	Proof Of Concept

Chapter 1. Introduction to the project

In the first part, our project discussed the importance of human resources in organizations. From that, BI solution to support measuring employee sales performance is built with the BI solution architecture as well as the general consideration of business strategy, business initiatives, deliverables, schedules, budget, resources in project.

1.1. Introduction

Human resource is considered the most important asset in any organization (Athukorala et all, 2020). Good human resource management will promote sales, thereby increasing the revenue and profits of the business. Therefore, good management as well as measuring employee sales performance is a very important thing for every enterprise, it determines the prosperity of the businesses. Besides, the IT industry has grown rapidly during the last few decades and the application of BI in businesses is also increasing. Applying BI solutions to measure employee sales performance will find insights to improve their sales performance as well as find out which factors (commission, demographic, shift, pay frequency) will greatly affect sales performance. To solve that problem, we decided to choose the topic "**Building a BI solution to support measuring employee sales performance**". With this topic, we will partly help businesses improve sales through Human resource management.

1.2. From business strategy to business initiatives

Business strategy:

- Goals: Improve overall sales effectiveness and maximize revenue generation through data-driven insights into employee sales activity, performance, and influencing factors.
- Metrics:
    - Targeted coaching and development for high-performing employees.
    - Increased sales revenue and profitability.

Business initiatives:

- Business objective:

- Improve the efficiency and productivity of sales teams to generate higher revenue.
- Utilize data analytics to gain actionable insights of salesperson activity and performance metrics with influence factors.

Deliverables

- Data source identification and integration:
  - Identify all data sources containing relevant employee information and sales performance data. This includes CRM systems, sales order data, commission structures, employee demographics (age, gender, tenure), shift schedules, and payroll data.
  - Ensure data quality and consistency across all sources.
  - Implement data integration processes to bring all data together into a central repository.
- Metrics definition:
  - Sales performance: Revenue generated, conversion rates, average deal size.
  - Commission structure analysis: Impact of commission structure on sales behavior (e.g., focus on high-margin products)
  - Demographic analysis: Identify any correlations between demographics (age, tenure) and sales performance.
  - Shift analysis: Analyze the impact of different work schedules (shifts) on productivity and conversion rates.
  - Pay frequency analysis: Explore potential relationships between pay frequency and employee motivation/performance.
- Data analysis and insights generation:
  - Analyze performance variations based on commission structures, demographics, shifts, and pay frequencies.
  - Identify high-performing employees and teams and uncover the underlying drivers of their success.
  - Pinpoint areas for improvement and opportunities for targeted coaching and development programs.

- **Schedule:** 3 months
- **Budget:** Allocate money to implement and monitoring cloud platform, reporting software and analytics team resources
- **Resources:** Data analyst, BI developer, sales leadership team, finance team, stakeholders.

1.3. BI solution

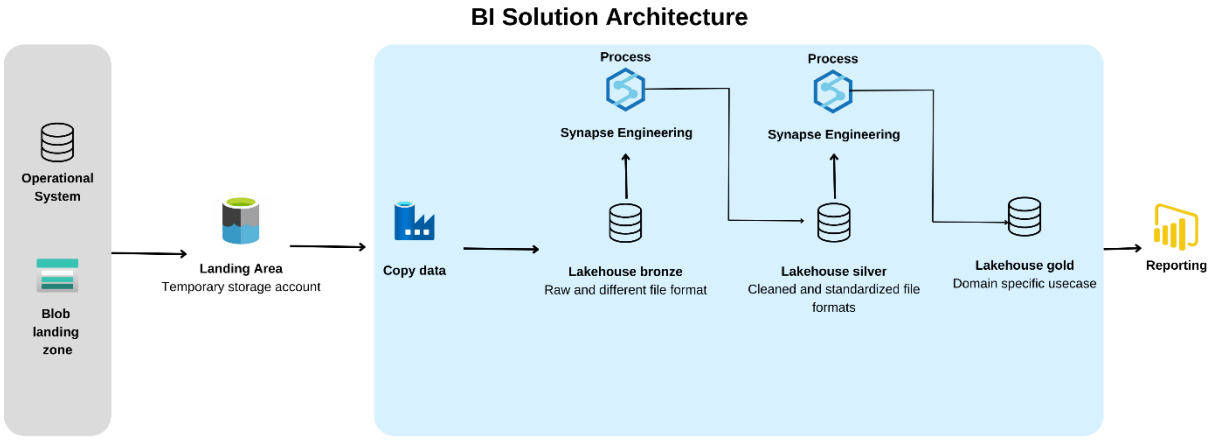


Figure 1.1. BI solution architecture

The BI solution is crafted to evaluate employees' sales performance based on diverse criteria such as commission, shift counts, and payroll data. It will use technologies like cloud-based data warehousing, business intelligence, and data-driven decision-making.

The data warehousing solution integrates streaming data from various sources into a unified cloud storage, facilitating easy data access, retrieval, and analysis. For visualization and comprehension, the business intelligence solution offers dashboards, aiding stakeholders in making informed decisions. The analytics solution will integrate machine learning methods and predictive analysis to identify trends in the data.

A multifunctional team comprising data engineers, data analysts, and business experts will design and deploy this BI solution. The project adheres to structured project management principles with clear goals, timelines, budgets, and resource requirements. The project manager oversees activities to ensure timely delivery within budget and stakeholder satisfaction.

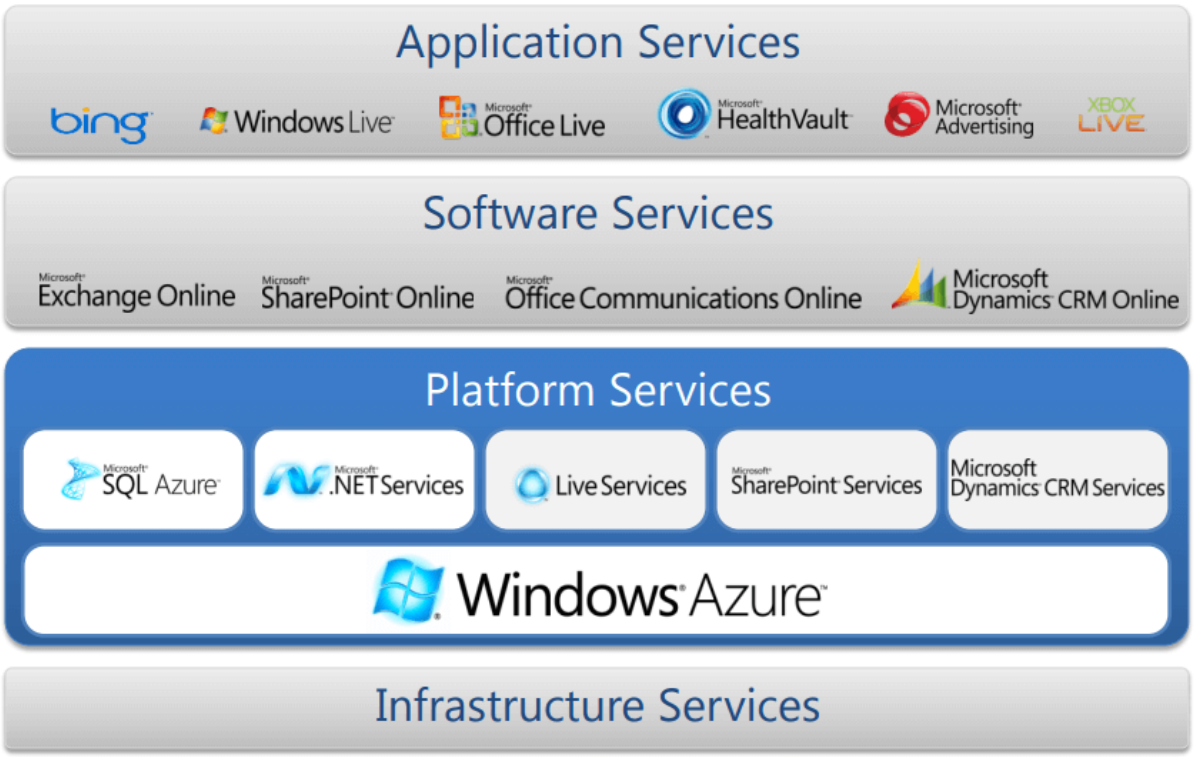
Chapter 2. Theoretical Basis

After project introduction, this part concerns the theory of concept, features, architectures of cloud platform Microsoft Azure and its services which are mainly applied in BI project. By understanding the method of the solution architecture, the role of each layer of Medallion architecture is explained clearly and detailly.

2.1. Cloud platform

2.1.1. Microsoft Azure

Microsoft Azure, initially called Windows Azure, is Microsoft's public cloud computing platform. It offers a broad range of cloud services, including compute, analytics, storage and networking. Released at the Professional Developers Conference (PDC) by Microsoft in October 2008, it marked Microsoft's entry into cloud computing providing a platform for developing and hosting applications in Microsoft's data centers.



[1]

Figure 2.1. Microsoft Azure services

Types of clouds in Microsoft Azure:

[1] Source: https://cmcts.com.vn/vi/microsoft-azure-la-gi-huong-dan-cach-su-dung-toan-dien-microsoft-azure.html

- IaaS (Infrastructure as a Service) forms the basic layer of cloud platforms. This Azure service is utilized by IT administrators for processing, storage, networking, and other essential computer operations.
- PaaS (Platform as a Service) provides a computing platform that includes an operating system, programming language execution environment, database, or web services. Developers and application providers use this Azure service to develop and deploy software. It enables clients to concentrate on development, operating systems, networking, and server management without worrying about the underlying hardware and infrastructure.
- SaaS (Software as a Service) refers to software that is centrally hosted and managed, with a single version used by all customers. It can be scaled to multiple instances to ensure optimal performance across various locations, and is typically licensed through a monthly or annual subscription.

2.1.2. Azure Data Factory

2.1.2.1. Definition

Azure's cloud-based ETL and data integration tool, Azure Data Factory, is built on Microsoft Learn and enables you to design data-driven workflows for coordinating data transportation and transforming data at scale. With Azure Data Factory, you can use compute services like Azure HDInsight Hadoop, Azure Databricks, and Azure SQL Database to create sophisticated ETL processes that change data visually with data flows. Data-driven workflows, also known as pipelines, can be planned and created to ingest data from many data sources.

2.1.2.2. Features of Azure Data Factory

- Data compression: You can write the compressed data to the target data source and compress the data during the Data Copy activity. This function facilitates data copying with optimal bandwidth use.
- Extensive Connectivity Support for Different Data Sources: Azure Data Factory offers a wide range of connectivity support for establishing connections with various data sources. When you wish to pull or write data from various data sources, this is helpful.

- Custom Event Triggers: Custom event triggers in Azure Data Factory let you automate data processing. With this capability, you can program an action to be performed automatically in response to a specified event.
- Data Preview and Validation: Tools for examining and checking data are offered throughout the Data Copy activity. This feature aids in making sure that data is accurately written to the target data source and copied.
- Customize Data Flows: You may design scalable data flows with Azure Data Factory. You can create unique stages or actions for data processing with this capability.
- Integrated Security: Role-based access control and connectivity with Azure Active Directory are only two of the integrated security capabilities that Azure Data Factory provides to manage dataflow access. Your data stay secure and data processing security is increased by this feature.

2.1.2.3. Azure Data Factory components

- Pipeline: A data factory may have a single pipeline or several. A pipeline is a logical arrangement of operations that carries out a specific set of duties. As an illustration, a pipeline can comprise a series of operations that first ingest data from an Azure blob and then split the data using a Hive query on an HDInsight cluster. Rather than controlling each activity separately, they let users manage the activities as a group. Pipelines might have activities that run independently in parallel or that are coupled to run sequentially.
- Mapping data flows: To change data of any scale, create and maintain graphs of data transformation logic. By creating a library of reusable data transformation functions, you can use your ADF pipelines to scale up and down these processes. You no longer need to manage or maintain clusters since Data Factory performs your logic on a Spark cluster that scales dynamically to meet your demands.
- Activity: Within a pipeline, activities are discrete processing stages. A copy activity, for example, moves data between data stores, whereas a Hive activity transforms or analyzes data by executing a Hive query on an Azure HDInsight cluster. Three categories of operations are supported by Data Factory: control, transformation, and movement of data.

- Datasets: Datasets represent data structures in data stores, pointing to or referencing the data used as inputs or outputs in activities.
- Linked services: Similar to connection strings, linked services specify the connection details required for Data Factory to access outside resources. A dataset depicts the structure of the data, whereas a linked service defines the connectivity to a data source. In Data Factory, linked services are utilized for two things:
    - To symbolize a data store that might be an Oracle database, file share, SQL Server database, or Azure storage account, among others. See the activity article for a list of supported data storage.
    - To display a computer resource that is capable of hosting an activity. See the transform data article for a list of supported computer environments and transformation activities.
- Integration Runtime: provides the compute environment from which the activity can operate or be sent by connecting it to the associated services. This guarantees that the operation runs as close as feasible to the intended data store or computer service, maximizing efficiency and satisfying security and compliance requirements
- Triggers: determine when a pipeline execution should start, with different types of triggers for various events.
- Pipeline run: is an instance of pipeline execution. Pipeline runs are typically instantiated by passing the arguments to the parameters that are defined in pipeline. The argument can be passed manually or within the trigger definition.
- Parameters: are read-only configuration key-value pairs. The pipeline defines the parameters. When a trigger or manually operated pipeline creates a run context, it passes arguments for the defined parameters throughout execution. The following parameter values are consumed by pipeline activities:
    - A dataset is strongly typed parameter and a reusable/referenceable entity. An activity can reference datasets and can consume the properties that are defined in the dataset definition.

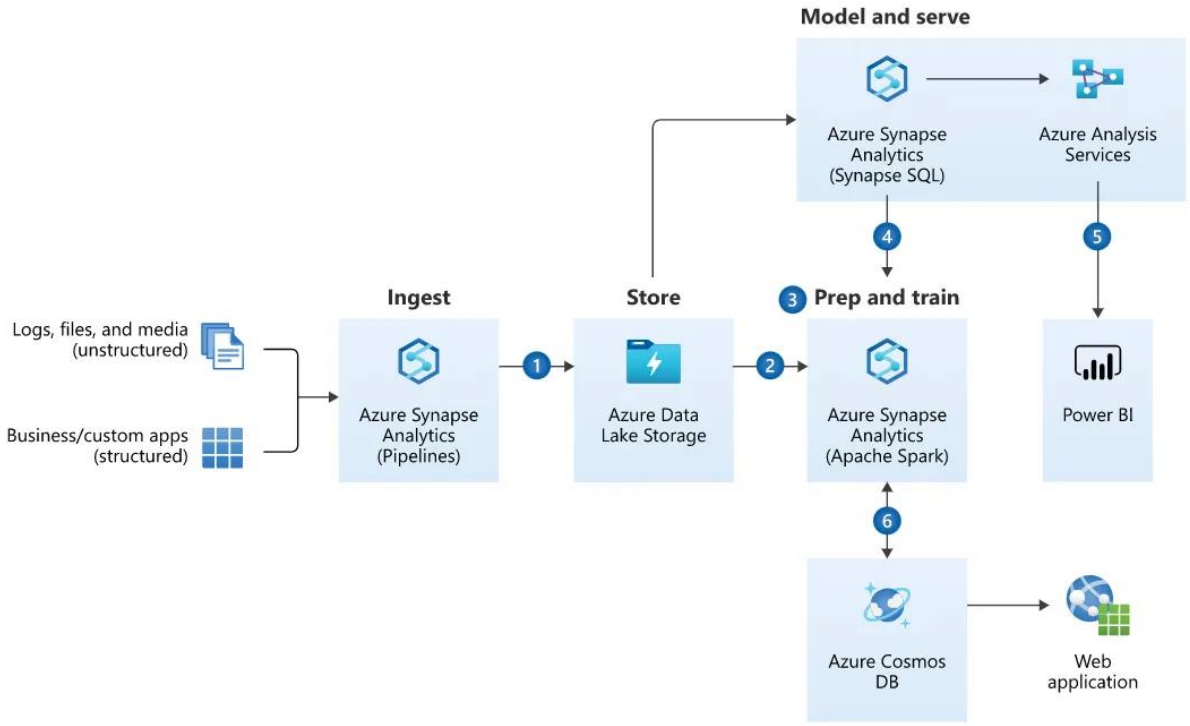
- A highly typed parameter also called a linked service holds the connection details to a compute environment or data store. It is an entity that may be referenced and reused.

- Control flow: a highly typed parameter called a linked service holds the connection details to a compute environment or data store. It is an entity that may be referenced and manages pipeline operations, such as branching, specifying pipeline parameters, passing arguments, and executing pipeline operations in a trigger-triggered or on-demand manner. Additionally, it has looping containers—that is, For-each iterations—and custom state passing.
- Variables: can store temporary values within pipelines and work with parameters to pass values between pipelines, data flows, and other activities.

2.1.3. Azure Synapse

2.1.3.1. Definition

Azure Synapse is an enterprise analytics service that accelerates time to insight across data warehouses and big data systems. The best Spark and SQL technologies for big data, Data Explorer for log and time series analytics, pipelines for data integration and ETL/ELT, and deep integration with other Azure services like Power BI, CosmosDB, and AzureML are all combined in Azure Synapse.



[2]

Figure 2.2. Azure Synapse services

2.1.3.2. Azure Synapse features

- SQL and Spark engines: Azure Synapse Analytics offers both Spark and SQL engines for data processing. Massively parallel processing (MPP) systems like the SQL engine are capable of processing queries over vast amounts of semi-structured and relational data. Batch, streaming, and machine learning workloads can all be handled by the distributed computing architecture known as Spark engine. Both engines work seamlessly together in the same workspace, allowing you to query data from various sources and formats.
- Data lake integration: It has native integration with Azure Data Lake Storage Gen2, a safe, scalable big data analytics storage solution. Without copying or moving any data, you may use both the SQL and Spark engines to access and query data in your data lake. You may explore, manage, and analyze your data lake files using Azure Synapse Studio, a web-based client.
- Data ingestion and transformation: Several techniques are available with Azure Synapse Analytics for ingesting and manipulating data into your data warehouse

[2] Source: https://medium.com/vedity/azure-synapse-analytics-69c19427eb00

from various sources. Your data pipelines may be orchestrated and automated by using Azure Data Factory, a fully managed data integration service. As an alternative, Azure Synapse Pipelines provides a graphical interface that doesn't require coding for creating and tracking data flows. Additionally, you may create custom data transformation logic with Azure Synapse SQL Scripts, which employ a language based on T-SQL.

- Data security and governance: For your data warehouse, this solution provides multiple tiers of protection and control. Users and apps can be authorized and authenticated using Azure Active Directory, a cloud-based identity and access management solution. The network-level security function Azure Synapse Firewall aids in managing the incoming and outgoing traffic to your workspace. For improved data safety, Azure Synapse Data Classification also enables you to locate and mark sensitive data in your tables and columns.
- Data exploration and visualization: A range of tools and frameworks are available for users to use in order to examine and visualize your data. Python, Scala, or.NET code may be written and run in an interactive environment with Azure Synapse Notebooks. You can execute ad hoc queries against your data lake files using the query-as-a-service feature provided by Azure Synapse SQL Serverless. Additionally, you can create and share dashboards and reports with Azure Synapse Power BI, a business intelligence solution.
- Data development and collaboration: It encourages an agile and cooperative approach to data development. Version control is available with Azure Synapse Git Integration to help you manage your projects and code. Continuous integration and delivery are made possible by Azure Synapse DevOps Integration, which lets you automate testing and deployments. To aid in your learning and development, the Azure Synapse Knowledge Center also provides examples, best practices, and tutorials.

2.1.4. Azure Blob Storage

2.1.4.1. Definition

Azure Microsoft’ s cloud-based object storage solution which is called Blob Storage is optimized for storing massive amounts of unstructured data. Data that does

not fit into a specific data model or definition, such text or binary data, is referred to as unstructured data.

2.1.4.2. Azure Blob Storage features

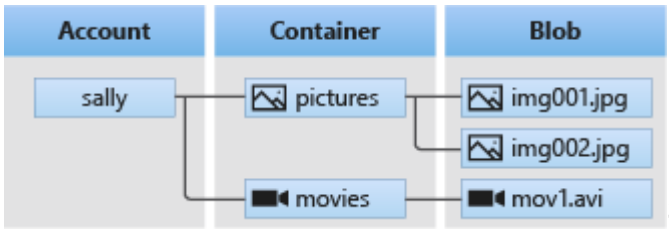
Blob Storage is designed for:

- Providing documents or images to a browser directly
- Holding onto files for shared usage
- Audio and video streaming
- Composing data to log files
- Data storage for disaster recovery, archiving, and backup and restoration
- Preserving data for analysis via an Azure or on-premises solution

Anywhere in the worldwide can use HTTP/HTTPS to access Blob Storage assets for users or client applications. Azure Storage REST API, Azure PowerShell, Azure CLI, and Azure Storage client libraries can all be used to access objects stored in Blob Storage. For a variety of languages, including .NET, Java, Node.js, Python, and Go, client libraries are available. Additionally, clients can mount Blob Storage containers using the Network File System (NFS) 3.0 protocol and establish secure connections to Blob Storage via the SSH File Transfer Protocol (SFTP).

2.1.4.3. Blob Storage resources

Blob Storage offers three types of resources:



[3]

Figure 2.3. Blob Storage resources

- Storage accounts: provides a unique namespace in Azure for your data. Every object that you store in Azure Storage has an address that includes your unique account name. The combination of the account name and the Blob Storage endpoint forms the base address for the objects in your storage account.

[3] Source: https://learn.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction

- Containers: organizes a set of blobs, similar to a directory in a file system. A storage account can include an unlimited number of containers, and a container can store an unlimited number of blobs.
- Blobs: stand for a “Binary Large Object”, a data type that stores binary data. They can be complex files like images or videos. Although they can consist of either structured or unstructured data, Blobs are mostly used in SQL (Standard Query Language) to store unstructured data files. Because Blobs are used to store multimedia files, they are often large pieces of data, up to several gigabytes.

2.2. Medallion architecture

The Medallion Architecture is a three-layered data management framework designed to optimize data storage, processing, and analysis within a cloud environment like Microsoft Azure. It divides data processing into three distinct layers—Bronze, Silver, and Gold—each tailored to handle data at different stages of refinement, making it a cornerstone for robust Business Intelligence systems. These layers will be clearly explained in the content below.

2.2.1. Bronze layer

The Bronze layer acts as the staging area for all raw data collected from various sources. In this project, this includes raw sales data, employee performance metrics, and other operational data from CRM systems, ERP systems, and direct inputs from business units. And at this level, the team will load raw data from the database with data tables including those of the Sales department with tables such as: SalesOrderDetail, SalesOrderHeader, SalesPerson. Product department with tables such as: Product, ProductSubCategory, ProductCategory. HumanResource department with tables like: EmployeePayHistory, Employee, Shift and Person department.

In the Bronze Layer, data has undergone preliminary cleansing and transformation while retaining its fundamental structure and format. Processing steps here often focus on removing invalid data, standardizing values, handling basic data inconsistencies, and making necessary transformations to align the data with the overall data model of the system. The Bronze Layer plays a vital role in preparing data for subsequent processing and analysis. Data at this layer often serves as the source for

creating higher-level data tiers such as Silver and Gold, where data is further transformed and refined to serve more specific business purposes.

2.2.2. Silver layer

The Silver Layer within business intelligence (BI) architecture plays a pivotal role in refining data sourced from the Bronze Layer by implementing various transformation, cleansing, and enrichment processes, ensuring it is well-prepared for in-depth analysis. Serving as a trusted repository of clean and query-optimized data, the Silver Layer does not aggregate data but rather focuses on providing a unified view by integrating and normalizing data from diverse sources, such as standardizing sales data from different regions.

Therefore, in this phase, the team cleans and filters the necessary columns for use in analyzing the solution. The team will divide the data groups into 7 small tables. With 2 large Fact tables including Fact_SalesTransactions and Fact_EmployeeShifts

After finalizing the selection of columns for analyzing the team's solution, they proceeded to convert the file format to Parquet, optimizing storage efficiency and enabling faster extraction into the Lakehouse Silver layer, facilitating streamlined data access and analysis processes.

2.2.3. Gold layer

In Gold layer phase of the Business Intelligence (BI) project, the focus shifts towards the final stages of data processing, specifically refining and optimizing the data for the Gold Layer. This layer is crucial as it represents the synthesis of cleaned and aggregated data ready for high-level business analysis and decision-making.

The process begins with data aggregation for each use case. Here, raw data from various sources is transformed into meaningful metrics that directly support business objectives. For example, raw data is processed to extract key business metrics, such as sales or employee performance, based on specific end-user requirements. Each round of this synthesis is designed to ensure consistency and accuracy of information, preparing for further analysis steps.

After successful testing, the data is loaded into the Gold Layer. This layer is the culmination of the data processing pipeline, where data is not only stored but also

formatted and optimized for direct use in business analysis and reporting. It is here that data becomes a strategic asset, readily accessible for comprehensive business intelligence initiatives. Once the data is aggregated, it is stored in a data warehouse where it is structured for optimal query performance. This warehouse serves as a centralized repository that supports fast and efficient data retrieval, which is essential for dynamic BI tools like PowerBI.

To leverage this refined data, a linked service with PowerBI is created. This setup integrates the Gold Layer with PowerBI, enabling real-time data visualization and interactive reporting. This integration facilitates immediate access to business metrics, empowering stakeholders to make informed decisions swiftly.

Chapter 3. Objectives and scope

Before implementing BI solution, this stage plans objectives, scope, constraint, resources in project and designs Work Breakout Structure and Gantt chart for controlling and tracking progress, identifying and dividing each member's tasks, suitably allocating resources,… Simultaneously, we describes dataset which team used in project, builds the schema to discover relations between tables in dataset, then metrics and KPIs are come up with in order to compute, measure, analyze and evaluate sales performance of enterprise employees,

3.1. Objective

3.1.1. General objective

The general objective of this project is to help businesses improve sales through employee management. BI solutions allow businesses to track productivity and analyze the performance of each employee, thereby providing accurate individual assessments based on diverse criteria. Additionally, data storage solutions, business intelligence solutions, and analytics solutions will serve as the foundation for building a data-driven decision-making system, enabling businesses to gain deep insights into their employees. From there, they can identify new opportunities and make data-driven decisions.

3.1.2. Specific objectives

In the process of building a Business Intelligence (BI) solution to support employee sales performance measurement, a comprehensive list of specific goals is required to complete the project, including:

- Build a BI solution that effectively visualizes data and tracks employee productivity and performance.
- Develop a comprehensive data warehouse architecture that can store entire sales history and personnel data, helping to securely store and manage employee performance data.
- Guidance documents for businesses and employees so they can use and maintain data warehouses effectively.

3.2. Scope

3.2.1. In scope:

The scope of the project is limited to designing a business intelligence system and developing a data warehouse model to support businesses in making human resource decisions and improve the company's business performance. The project can help the human resources department perform their tasks more effectively. The expected time to complete the project is within two months, the project starts on March 5, 2024.

3.2.2. Constraints

- Technical infrastructure: Prioritize open source and sponsored software, technology, ecosystem,..
- Stakeholder support: Assume that stakeholders, including the customer, business, project developers, and management, support the project goals and are willing to follow up and support promptly throughout the development and testing phases.
- Budget and resources: The project is limited in technology, limiting Microsoft accounts to student and economic versions to perform tasks within only about $100 per account.

3.3. Resources

Table 3.1. Resource roles

Stakeholders	Role	Description
Le Ba Thien	Sponsors, CTO	Consulting and providing necessary knowledge about BI solutions and cloud service.
Nguyen Thi Thu Thao	Project Manager	Primarily responsible for planning and motivating the team to achieve project goals and ensure timely completion. Roles of Project Manager includes a comprehensive understanding of project operations, effective communication skills, and the ability to overcome challenges that may arise.

Stakeholders	Role	Description
Hoang Tran The Phuc	Data Engineer	Data Engineers' responsibilities include data modeling, batch processing, data matching, data searching, and duplicate record checking. Make sure the data tells the right story. Ensure data architecture for customers' Business Intelligence (cloud service) and Business Analytics platforms.
Phan Thanh Giang	Database Administration	Detailed administration and management of Database issues, including data backup and recovery, capacity allocation planning, configuration settings, database design, migration, performance monitoring, security and troubleshooting when they occur.
Luc Minh Phu	Data Analysts	Work with data to bring about effective change in your company's business. Help customers understand the vast amount of data they own.
Nguyen Hoang Duy Thong	BI Developer	A BI developer's primary tasks include creating, implementing, and managing BI tools and interfaces. However, BI interface development requires deep experience in software engineering, databases, and data analytics.
Nguyen Thi Thu Thao	Business Analyst	Collect and systematize requirements from customer. Prepare documents for BI solution development

Stakeholders	Role	Description
Phan Thanh Giang	Tester	Test and ensure the quality of the program and perform error checking work before delivering the final product to the customer.

3.4. Data description

Employee:

Table 3.2. Data description of Employee table

Name	Data Type	Description
EmployeeKey	int	Primary key for Employee records.
FirstName	Nvarchar(50)	First name of the Employee
LastName	Nvarchar(50)	Last name of the employee
JobTittle	Nvarchar(50)	Work tittle.
CommissionRate	smallmoney	Commission percent received per sale
PayFrequency	tinyint	1= Salary received monthly 2= Salary received biweekly

Sales Transaction:

Table 3.3. Data description of Sales Transaction table

Name	Data Type	Description
TransactionKey	int	Primary key
EmployeeKey	int	Primary key for Employee records.
ShiftKey	tinyint	Primary key for Shift records.
TimeKey	int	Identification of date records.

Name	Data Type	Description
SalesOrderID	int	Primary key
ProductKey	int	Product identification number
CustomerID	int	Primary key, Customer identification number
UnitPrice	money	Vendor's selling price of a single product.
OrderQty	int	Product quantity to build
Status	tinyint	Order current status. 1 = In process; 2 = Rejected; 3 = Approved; Default: 1
StandardCost	money	Standard cost of the product

Shift:

Table 3.4. Data description of Shift table

Name	Data Type	Description
ShiftKey	tinyint	Identifies which 8-hour shift the employee works.
ShiftName	Nvarchar(50)	Shift description
StartTime	Time(7)	Shift start time
EndTime	Time(7)	Shift end time

Employee Shifts:

Table 3.5. Data description of Employee Work Stats table

Name	Data Type	Description
EmployeeKey	int	Primary key for Employee records.

Name	Data Type	Description
ShiftKey	tinyint	Identifies which 8-hour shift the employee works.
TimeKey	int	Identification of date records.
SickLeaveHours	smallint	Number of available sick leave hours
Age	smallint	The age of employee.
PayrollRate	money	Salary hourly rate

Sales Order:

Table 3.6. Data description of Sales Order table

Name	Data Type	Description
SalesOrderID	int	Primary key
OrderDate	datetime	Dates the sales order was created
TotalDue	money	Total due from customer

Product:

Table 3.7. Data description of Product table

Name	Data Type	Description
ProductKey	int	Primary key for Product records
ProductName	Nvarchar(50)	Name of the product
ProductCategoryName	Nvarchar(50)	Category description

Date:

Table 3.8. Data description of Date table

Name	Data type	Description
DateKey	int	Identification of date records
Date	Datetime	Date key
Year	Int	Year (2011, 2012, 2013, 2014, 2015)
Quarter	Int	Quarter (1, 2, 3, 4)
Month	Int	Twelve months in one year
Week	Int	Week in month (1,2,3,4)
DayOfWeek	Nvarchar(255)	Name of the day in week

3.5. Metrics/KPI

3.5.1. Sales Transaction

Sales Amount:

- Measure: the total monetary value of sales transaction.
- Formula: *SalesAmount = OrderQty * UnitPrice*

Revenue:

- Measure: The total income generated from sales after accounting for the cost of goods sold (COGS)
- Formula: *Revenue = SalesAmount - COGS*

Conversion Rate:

- Measure: Customer conversion rate to purchase
- Formula: *ConversionRate = Count(Status = 3)*100 / Count(Status)*

KPI Sales Completed Rate:

- Measure: A Key Performance Indicator (KPI) that measures the percentage of the sales quota achieved during a specific period.

- Formula: *KPISales = (Sum(SalesAmount) / SalesQuota)*100*

Average Number Customer:

- Measure: This index measures how many customers a Sales employee earns or serves.
- Formula: *AverageNoCustomer = Count(CustomerID)/Count(EmployeeKey)*

3.5.2. Employee Shifts

Male Rate:

- Measure: the ratio of male and female employees in the department.
- Formula: *MaleRate = (Count(Male)/Count(EmployeeKey)) * 100*

Average Age:

- Measure: The average age of employees in the department.
- Formula: *AverageAge = Average(Age)*

Average Payroll:

- Measure: The average payroll rate of employeesin the department
- Formula: *AveragePayroll = Average(PayrollRate)*

3.6. Schema development

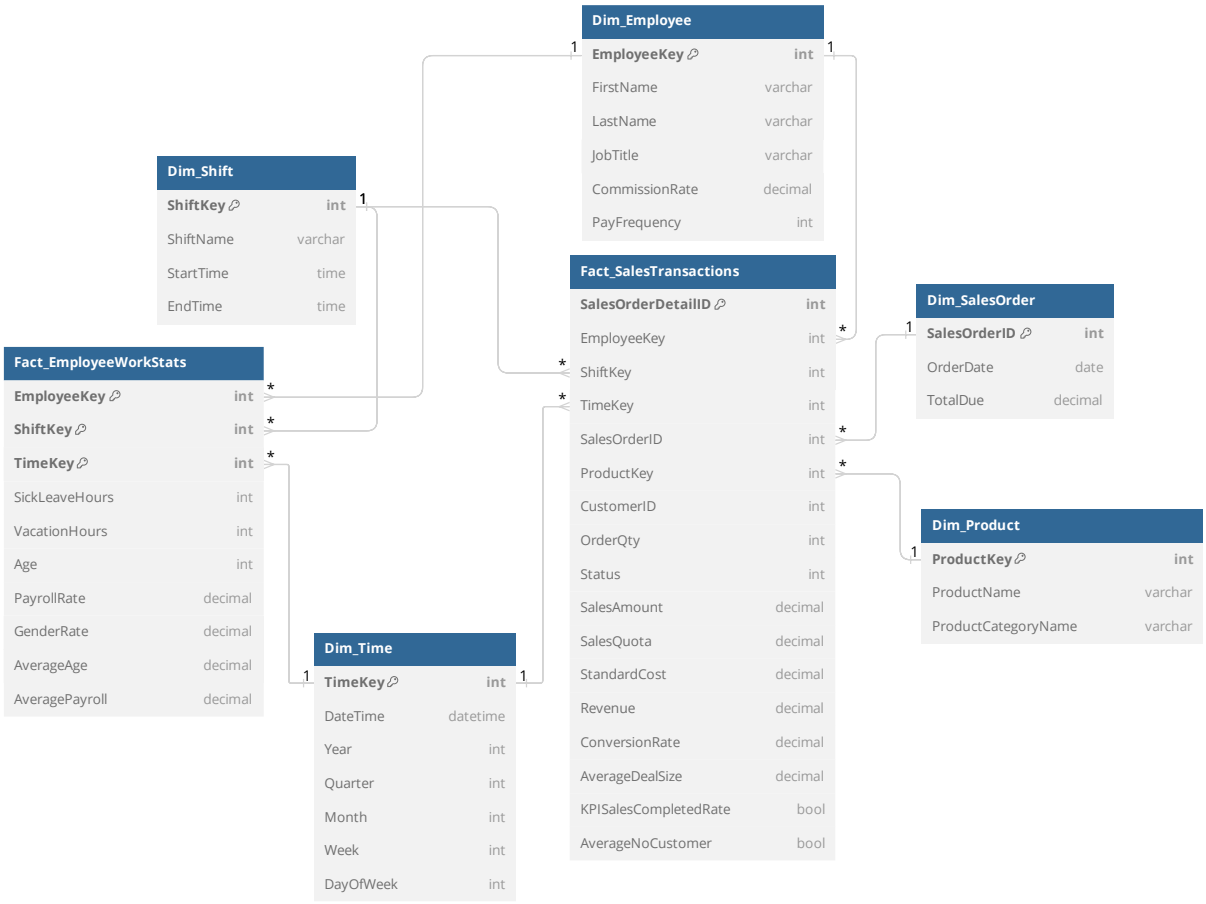


Figure 3.1. Schema for data warehouse of tracking sales performance

The table relationships in this schema play a crucial role in enabling comprehensive analysis and reporting of salesperson performance and sales transactions. Here's an analysis of how these relationships contribute to the effectiveness of the schema:

- Fact_EmployeeWorkStats and Dim_Employee**:**

- The EmployeeKey foreign key in Fact_EmployeeWorkStats establishes a one-to-many relationship with Dim_Employee, allowing each employee's work statistics to be associated with their personal details like name, job title, commission rate, and pay frequency.
- This relationship enables analysis of work metrics (e.g., sick leave hours, vacation hours, age, payroll rate) by employee attributes, facilitating performance evaluation, compensation analysis, and workforce planning.

- Fact_EmployeeWorkStats and Dim_Shift**:**

- The ShiftKey foreign key in Fact_EmployeeWorkStats creates a one-to-many relationship with Dim_Shift, connecting employee work statistics to the corresponding shift details.
- This relationship allows for analyzing work metrics based on shifts, enabling comparisons across different work schedules, identifying productivity patterns, and optimizing shift assignments.

- Fact_EmployeeWorkStats, Fact_SalesTransactions, and Dim_Time:

- Both fact tables (Fact_EmployeeWorkStats and Fact_SalesTransactions) have a TimeKey foreign key referencing Dim_Time, establishing a one-to-many relationship.
- This relationship enables time-based analysis of work statistics and sales transactions, allowing for trend analysis, period-over-period comparisons, and identification of seasonal patterns.
- It supports reporting and analysis across different time granularities (e.g., year, quarter, month, week, day), providing flexibility in analyzing performance metrics.

- Fact_SalesTransactions and Dim_Employee:

- The EmployeeKey foreign key in Fact_SalesTransactions connects sales transactions to individual employees, creating a one-to-many relationship with Dim_Employee.
- This relationship enables tracking and analyzing sales performance at the employee level, allowing for performance evaluation, incentive calculation, and sales territory management.

- Fact_SalesTransactions and Dim_SalesOrder:

- The SalesOrderID foreign key in Fact_SalesTransactions establishes a one-to-many relationship with Dim_SalesOrder, linking sales transactions to their corresponding sales orders.
- This relationship supports order-level analysis, such as tracking order sizes, revenue, and profitability by order attributes like order date and total due.

- Fact_SalesTransactions and Dim_Product:

- The ProductKey foreign key in Fact_SalesTransactions creates a one-to-many relationship with Dim_Product, associating sales transactions with product details.
- This relationship enables product-level analysis, such as monitoring sales performance by product categories, identifying top-selling products, and conducting product mix analysis.

By leveraging these table relationships, the schema facilitates a comprehensive understanding of salesperson performance and sales transactions from multiple perspectives. It allows for drill-down and roll-up analyses, aggregations, and slicing and dicing of data based on various dimensions, enabling informed decision-making and strategic planning for the organization.

3.7. Work breakdown structure

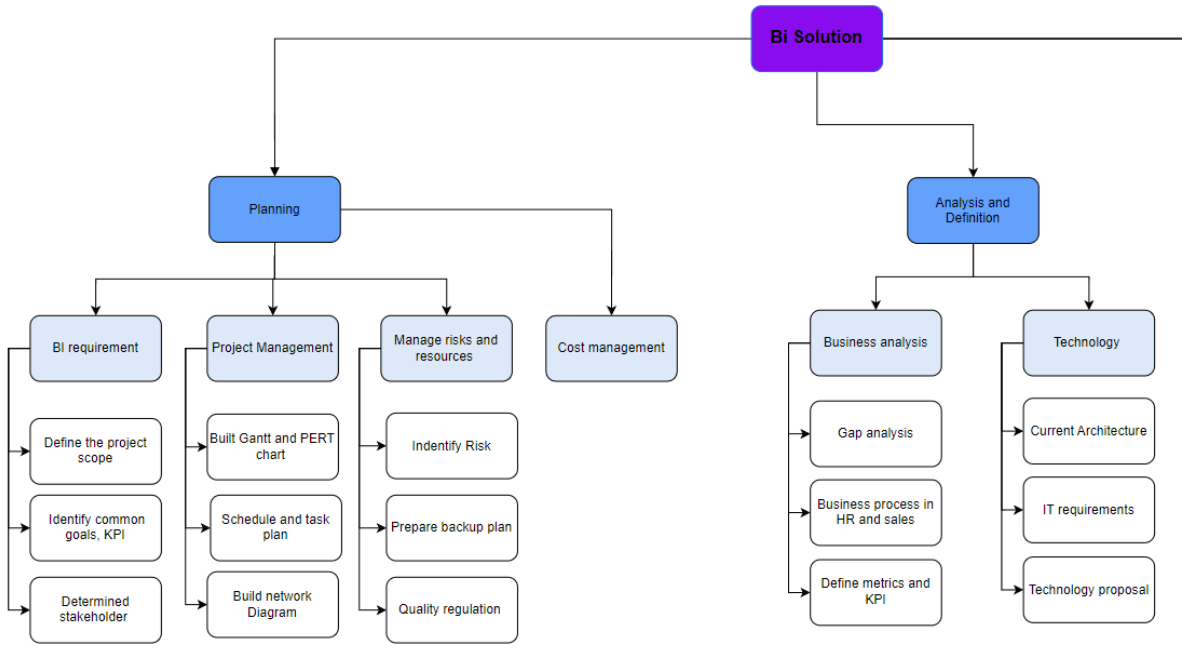


Figure 3.2. Work break down with phase Planning and Analysis

The planning phase serves as the foundation for building and implementing a Business Intelligence (BI) solution, involving detailed planning and thorough preparation for the project. Initially, it's crucial to clearly define the BI requirements, including the project scope, overall objectives, and key performance indicators (KPIs). Next, in the Project Management section, creating Gantt and PERT charts will help

outline the necessary time and resources, as well as schedule and plan specific tasks. Developing network diagrams also contributes to more effective project management. Another essential step is risk and resource management, which includes identifying potential risks, preparing contingency plans, and ensuring quality standards are adhered to throughout the project execution.

In the analysis and definition phase, the goal is to thoroughly analyze and shape the business and technical requirements for the BI solution. This process starts with a Gap Analysis, assessing the disparity between the current situation and desired objectives. Simultaneously, it's important to identify and analyze business processes related to personnel and sales, particularly by defining specific indicators and KPIs. On the technological side, evaluating the current architecture and determining the necessary IT requirements for the solution is essential. Finally, the technology proposal is the last step, where suitable technological solutions are suggested to optimally support the developing BI solution.

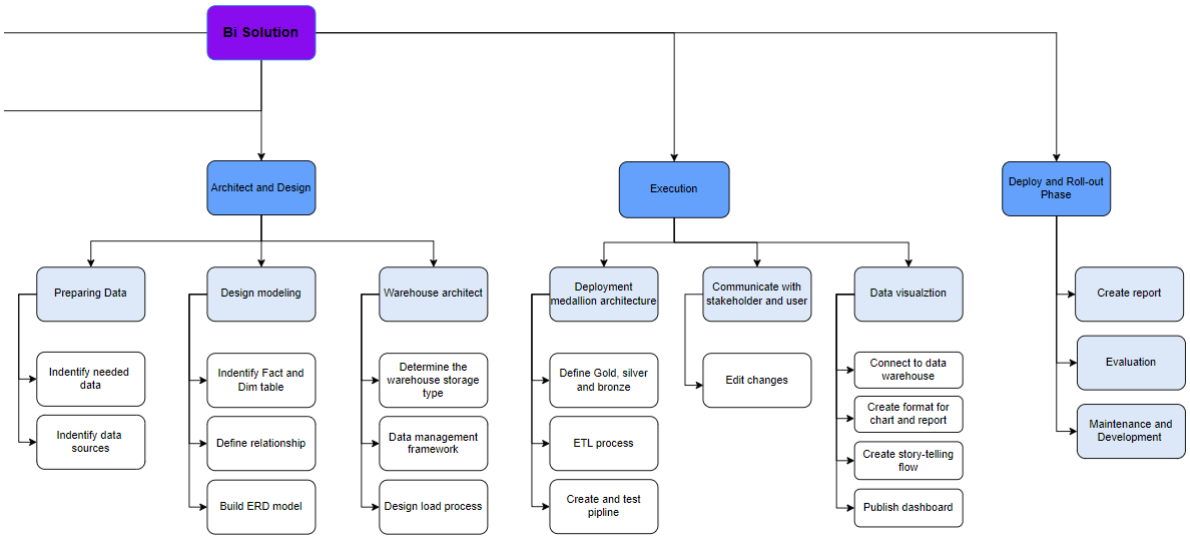


Figure 3.3. WBS with Architect, Execution and Deploy phase

Architect and design phase the architect and design phase focus on the system design and architecture, beginning with Data Preparation, where the necessary data is identified, and data sources are recognized. The next step is Model Design, which includes defining fact and dimension tables, establishing relationships between tables, and constructing an Entity-Relationship Diagram (ERD). Finally, Data Warehouse

Architecture involves determining the storage type for the data warehouse, setting up the data management framework, and designing the data loading processes.

In the execution phase, essential activities are carried out to deploy the BI solution. Initially, the implementation of the medallion architecture is conducted, which involves defining the gold, silver, and bronze structure for data, performing ETL (Extract, Transform, Load) processes, and creating data processing pipelines. Another crucial aspect is Communication with stakeholders and users, making adjustments based on feedback, and providing continuous updates to ensure the solution meets real-world needs.

The final phase of the project is Deploy and Roll-out, where the BI solution is put into practical operation. During this phase, the creation of Reports and Evaluation of the solution's performance is crucial to ensure the project's objectives are met. Lastly, Maintenance and Development ensure the system can be updated and adjusted over time to adapt to changing business and technological needs. This phase guarantees that the solution remains relevant and continues to deliver value as circumstances evolve.

3.8. Gantt chart

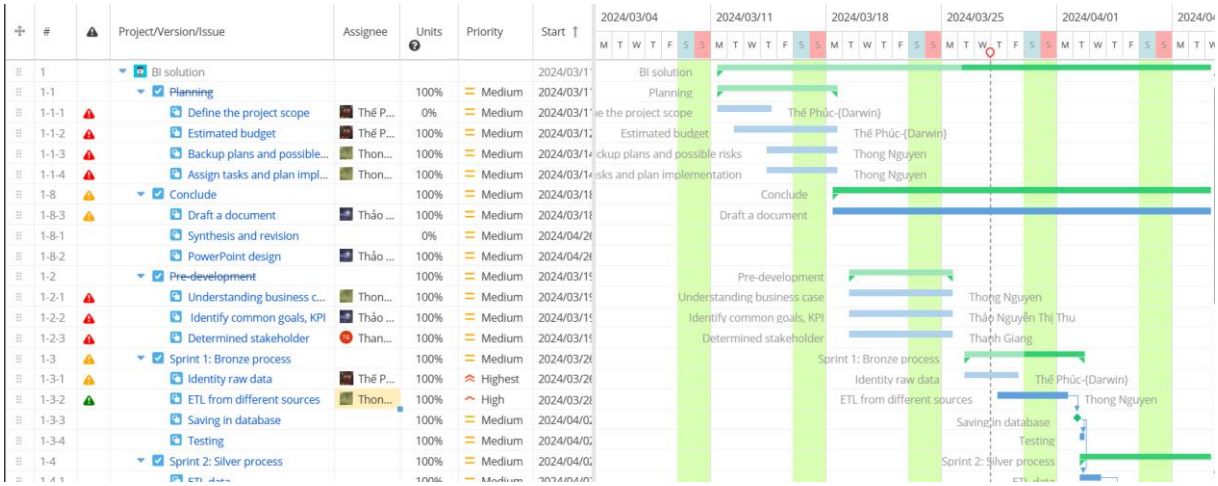


Figure 3.4. Gantt chart

In the planning phase, the initial steps of the project are shaped with the aim of clarifying the scope, budget, and detailed planning. First, defining the project scope involves clearly identifying the objectives, scope of activities, and final outcomes the project aims to achieve. Next, budget estimation focuses on calculating the necessary costs for the project, including resources, materials, and other expenses. Then, carrying out the work and detailed planning includes dividing tasks and creating detailed schedules for time and personnel for each specific activity. Finally, risk and issue management is essential, involving the development of contingency plans for potential risks and unforeseen issues, ensuring the project can proceed smoothly and achieve the set goals.

During the development phase, the main emphasis is on carrying out essential activities to meet the established goals. Initially, comprehending the fundamental business context requires studying and evaluating the project's business environment. Following this, the project objectives are detailed based on the previous research and analysis. Identifying stakeholders includes recognizing and evaluating the key individuals or groups that affect or are affected by the project. Furthermore, refining and optimizing processes focuses on assessing and enhancing work procedures for better efficiency. Lastly, validating through diverse sources involves testing and assessing the project with various information and data to ensure its feasibility and effectiveness.

Next, the project is divided into three sprints, each focusing on a specific phase of the data processing workflow to ensure flexible and efficient project management. In

Sprint 1: Bronze Process, the emphasis is on handling raw data, including identification, classification, testing of data pipelines, and loading processed data into the Bronze layer. Sprint 2: Silver Process builds upon this by refining data quality through deeper processing and filtering out unnecessary elements. Sprint 3: Gold Process aims to finalize and optimize data quality before its use in analytical applications. This involves aggregating data, storing it in a data warehouse, ensuring accuracy, and establishing direct data access through PowerBI. Each sprint concludes with detailed planning to facilitate smooth execution and resource allocation. This structured approach ensures systematic progress and enhances the overall efficiency and quality of the data processing workflow.

In final step, design dashboard, the team continues to develop and refine visualization and reporting tools to support data analysis. Tasks in this sprint include designing data dashboards, focusing on effective and intuitive information presentation. Developing visualization and reporting solutions using BI tools like PowerBI, ensuring data connectivity, and updating for analysis and visualization. Additionally, tasks involve designing data storytelling frameworks to decide how dashboards are organized and creating mockups for reports to determine their appearance and structure before final development. The sprint also includes detailed planning for tasks and testing the accuracy and aesthetics of the designed dashboards to ensure they are accurate, user-friendly, and engaging. These steps contribute to creating quality, easy-to-use, and appealing visualization and reporting tools, supporting end-users in understanding and interacting with data efficiently.

Chapter 4. BI Assessment

In BI assessment, it has three phrases: discovery phase for considering and understanding requirements; the analysis involves examining the current BI architecture, identifying gaps between the current state and desired future state, and evaluating the necessary steps to bridge these gaps by analyzing information, data, technical, and product aspects, with a focus on improving data sources, processing, and integration using Azure services; the recommendation phase suggests improvement to align the project's current state with its objectives and ensure successful implementation of the BI solution.

4.1. Discovery phase

4.1.1. Business Requirements:

- Desired reporting and analysis: Real-time monitoring of individual sales performance metrics such as sales, average deal size, geography of customer.
- Needed data for reporting: Employee information, Sales transaction, customer geographic, top sales staff has high customer retention, product volume.
- Current reporting and data shortcomings and workarounds: Lack of real-time visibility into individual sales metrics, manual compilation of sales reports leading to delays in performance analysis.
- Priorities: Implementing a BI solution for real-time performance of sales staff dashboards.
- Analytical skills: Sales team members possess basic to intermediate skills in using sales CRM tools but need training on advanced BI reporting tools for deeper analytics.

4.1.2. IT requirements:

- IT direction and plan: Implementing the cloud-based (Azure) data warehouse to integrate data from sales person, territory, customer and sales data.
- Project methodologies: This project using Agile for management.
- Collaboration Tools: Use project management and collaboration tools (Jira)
- Skill required:

    - Database: Strong SQL skills for data querying, manipulation, and optimization in cloud databases (Azure SQL Database).
    - Reporting/BI: Proficiency in Power BI for dashboard development, report creation, and data visualization.
    - Data Modeling: Understanding of dimensional modeling concepts for designing efficient data models in BI solutions.
    - Cloud platform: Understanding architecture, core software of Azure.
- Skill levels evaluation: IT team members has a good knowledge in database management, data modelling, reporting tool and understanding of cloud platform architecture as well as its software.
- Demo of reporting:
    - Customer demographic for top sales person.
    - Territory sales report.
    - Product sales value and volume of top sales person.
    - Customer retention rate of top sales person.
    - Shift sales efficiency.
    - Commission and sales correlation

4.2. Analysis phase

4.2.1. Current Architecture

The analysis involves examining the architecture, organization, and requirements in terms of where you are now and where you would like to move your BI environment. Determine the gaps between the two, validate your ability to achieve your future end state, and determine what you would need along all three of these dimensions to achieve success. With Architecture WBS architecture in our project, the graph is divided into 4 aspects: information, data, technical, product.

Firstly, in the information aspect, departments and staff defining and using system like: HR Manager, Sales Manager, Salesperson and other business stakeholders are human factors. Besides, there also includes business process in Sales and HR are impacted through accessing business applications such as: SCM and HRM. About used business and data subjects, it consists of: commission and sales correlation, Shift sales efficiency (by sales per hour, average sales per shift), customer demographic (by age, gender,

location, segments and so on), report of sales territory (by comparison between salesperson performance in territories), customer retention rate (by month, by quarter), product sales value and volume analysis (by breakdown by categories, product lines).

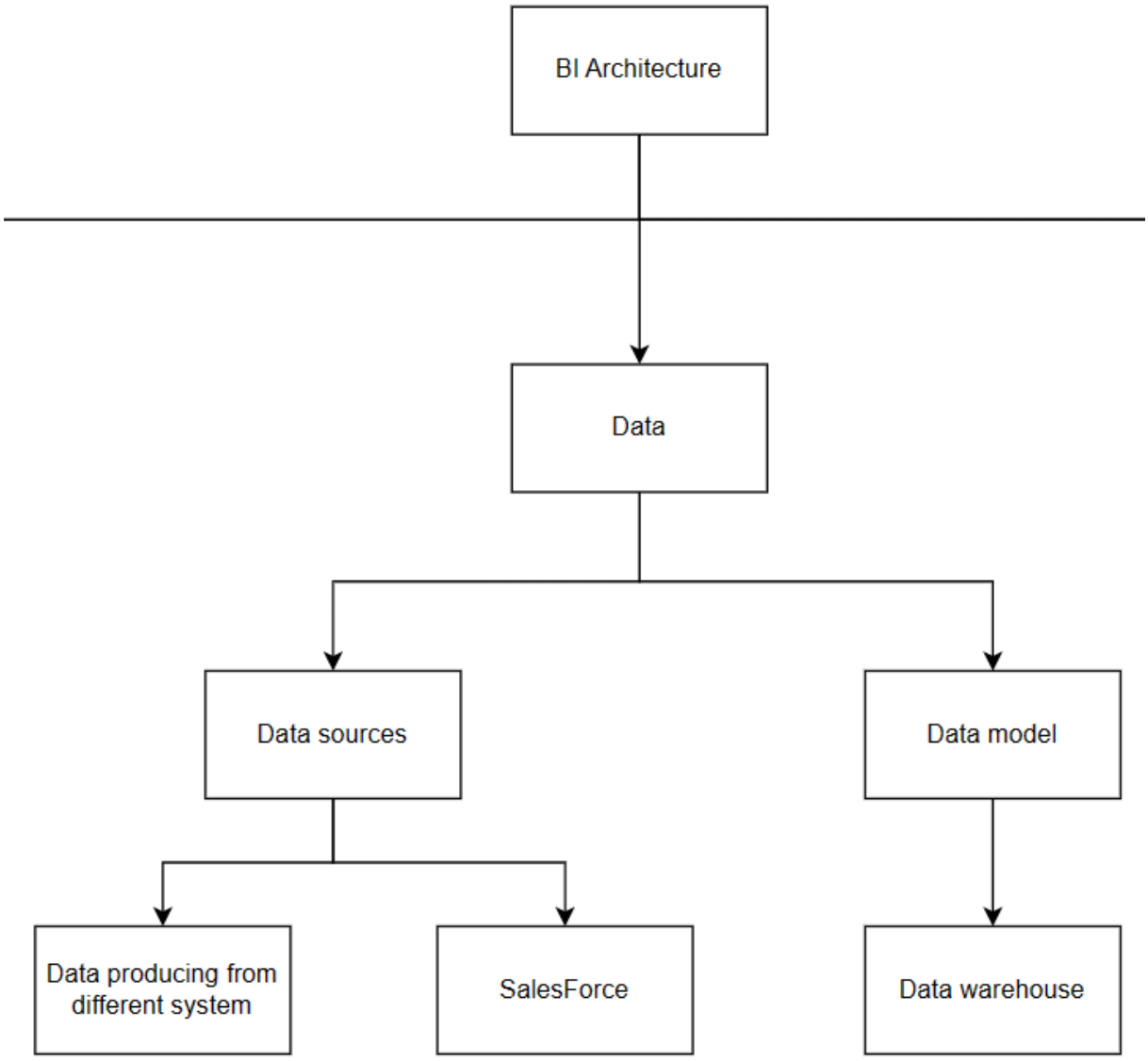


Figure 4.1. WBS architecture (Data)

Secondly, about aspect of data, its architecture are data sources containing Manual data entry source from different system and SalesForce, and Data warehouse is applied in data model.

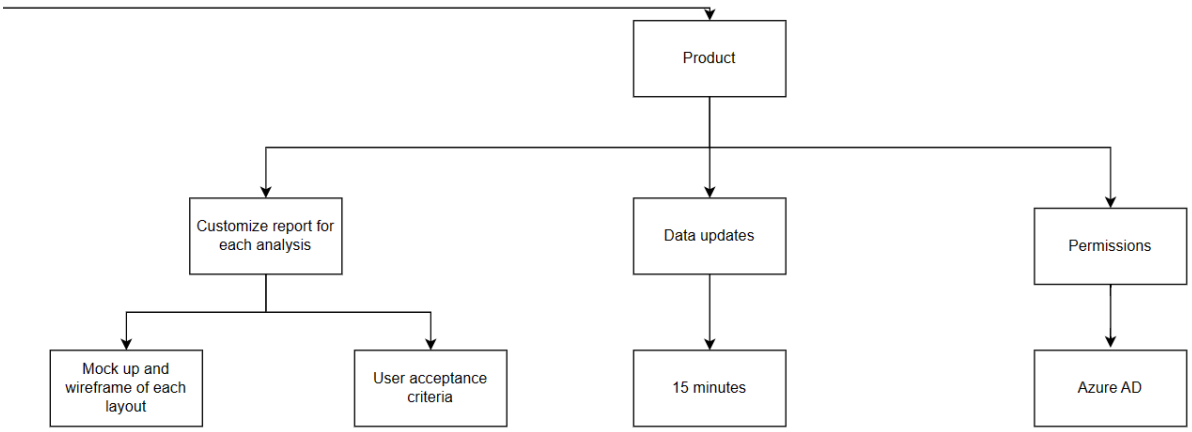


Figure 4.2: WBS Architecture (Product)

To implement the project, technique is an essential condition for the development team. In technical aspect of architectures, we define using Microsoft Azure services, so we consider the architecture of data processing, infrastructure, reporting tools, data storage, data integration related to Azure. More detailly, data processing is Azure Synapse, Infrastructure is Azure Cloud Infrastructure, reporting tools is Power BI, data integration is ADF (Azure Data Factory).

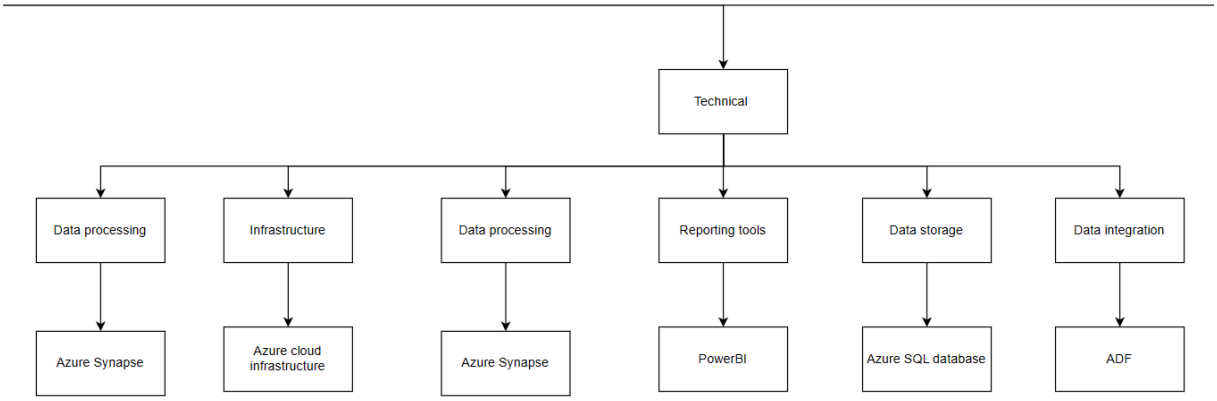


Figure 4.3: WBS Architecture (Technical)

The final factor we need to explore is product. Detailly, we review mockup and BI Architecture WBS (Technical)wireframe of each layout as well as user acceptance criteria in customizing report for each analysis. Besides, we defined that data updates is 15 minutes, permissions are Azure AD (Azure Active Directory).

4.2.2. Gap analysis

- What is desired:

- Build a comprehensive data warehouse encompassing Sales and HR department data.
- Analyze data to derive actionable insights and address predefined objectives related to sales and HR department

- What is reasonable:
  - We defined some definitions which are necessary in the project about: business case, some goals, objectives and determined the role of related resources, in order to get best results in implementation, gain accomplishment satisfying the requirements.
  - The data source of our BI project is collected from departments as well as the understanding of each data attribute to apply analysis process to get insights
  - Our team must have some major knowledge related to analytics and data, including concept and practice, especially understandings about Data storage, data integration, data processing in order to build the data architecture and discover data through analytics in on-premise software (SSIS, SSAS).
  - About soft skills, team members has some skill of analytical thinking, teamwork, presentation, communication, problem solving. They are also essential for resources in BI project to contribute effectively to the project's success and drive value for the organization.
- What gets added to the to-do list:
  - Data cleansing: Identify and prioritize data cleansing activities to ensure that the data being used for analysis is accurate, consistent, and reliable. This may involve removing duplicates, correcting errors, standardizing formats, etc.
  - Master data management (MDM): Implement MDM practices to establish a single, consistent view of key business entities such as customers, products, and employees across the organization. This will help ensure data integrity and facilitate accurate analysis and reporting.

- Data governance framework: Develop and implement a data governance framework to define policies, procedures, and responsibilities for managing data assets effectively throughout their lifecycle. This includes ensuring compliance with regulatory requirements, maintaining data security and privacy, and establishing data quality standards.
- Data integration strategy: Define a comprehensive data integration strategy to bring together data from disparate sources into the data warehouse. This may involve selecting appropriate ETL (Extract, Transform, Load) tools, designing data pipelines, and establishing data synchronization processes.
- Advanced analytics capabilities: Explore opportunities to enhance the BI project with advanced analytics techniques such as predictive analytics, machine learning, or natural language processing. This can enable more sophisticated insights and predictive modeling to support decision-making.
- Stakeholder engagement plan: Develop a stakeholder engagement plan to involve key stakeholders from Sales, HR, IT, and other relevant departments throughout the project lifecycle. This will help ensure alignment of objectives, manage expectations, and secure buy-in for the BI initiative.

4.3. Recommendations phase

From gap analysis, we understand where we are and where we desired in this project. Recommendations stage can discuss suggestions involve the architecture (information, data, technology) and your organization (program/project plans, resources/skills/training, and rough cost estimates) through feedback and recommendations on the current state of your readiness in relation to your requirements and capabilities:

- Information: Documents on working processes in departments to better grasp the activities of these departments.
- Data: With the variety of data sources, project can deploy them to meet the business requirements. However, if possible, our team maybe collect some

additional data from the Sales and HR departments to deeply current state, such as:

- Sales:
  - Conversion rate: Analyze lead-to-customer conversion rates to evaluate the performance of marketing and sales strategies.
  - Sales cycle completion time: Measure the time from when a lead is generated to when the deal is completed to determine the performance of the sales process.
  - Customer feedback: Collect and analyze reviews and feedback from customers to understand the positives and improve the weaknesses of the product or service.
- HR:
  - Recruitment success rate: Analyze the recruitment success rate compared to the total number of recruitments to evaluate the performance of the recruitment process.
  - Turnover rate: Measure the annual employee turnover rate to evaluate employee satisfaction and stability.
  - Employee surveys: Collect feedback from employees through surveys to understand satisfaction levels and improve the work environment.

- Technology: Instead of applying on-premises, team need to use some cloud platform, for instance, Microsoft Azure provides a wide range off cloud-based services, including computing power, storage solutions, networking capabilities, databases, analytics and reporting tools.
- Resources: Because of limited resources, team members need to support each other in spite of different roles in project to reduce each person's workload in each sprint.
- Skills: Beside soft skills team members have, we need to learn more soft skills about critical thinking, time management, adaptability, storytelling. Next, we have to cultivate highly specialized skills in Data engineer, Data Visualization, Testing and be able to deploy and apply knowledge then build a data warehouse on cloud tools (Microsoft Azure) optimally and ensure data security.

- Training: By participating in workshop or training about project management methodologies, domain-specific knowledge related to industry or business domain of the BI project.

Chapter 5. Experimental Results

In this implementation stage, our project presents the details of each layer in Medallion architecture. Bronze layer is the layer for raw data to be loaded with different file formats, Silver layer is the layer for processing, cleaning data, selecting columns in tables, and storing all files in Parquet format. Then after calculating metrics, pipeline is built to load metadata into Gold layer. This data is moved from data lake to data warehouse as dedicated pool for building Dashboards.

5.1. Bronze layer

In this step, we collected all the data related to this project, from Sales to HR department from three different formats: json, csv and xml. Raw data sources include:

- Customer.xml
- Person.xml
- Product.xml
- ProductCategory.xml
- ProductSubCategory.xml
- DepartmentHistory.csv
- Employee.csv
- Shift.csv
- EmployeePayHistory.csv
- SalesOrderDetail.json
- SalesOrderHeader.json
- SalesPerson.json

Then, we uploaded it to data lake storage gen 2 in Microsoft Azure. The data is stored at Bronze layer in data lake and we got the results as below:

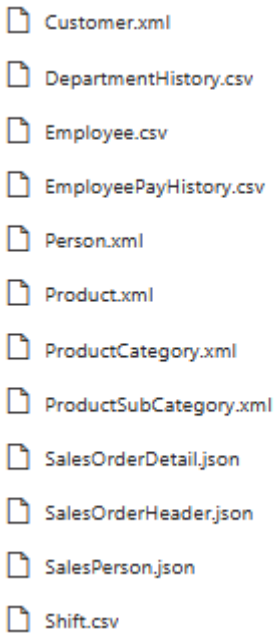


Figure 5.1: Output of Bronze layer

This is also the output of bronze layer since it acted as staging area.

5.2. Silver layer

Before loading data into the Silver layer, our team will select the columns we need as well as process the data such as removing duplicates and filtering columns.

First, our team made the connection to Azure Data Lake Storage Gen 2 through Azure Databricks.

```
container_name = "datalake"
storage_account = "adlsk21416ck214162145"
key = "5Gj7eCzWgZW1slBbI6CUcmecW7ymZRgQyAvdwQLpb0Uyfvl7rhwJ4ggUz9jjTADTKGWMRqUtCyJE+AStDgjKTw=="

url = "wasbs://" + container_name + "@" + storage_account + ".blob.core.windows.net/"
config = "fs.azure.account.key." + storage_account + ".blob.core.windows.net"
mount_folder = "/mnt/"
dbutils.fs.mount(source = url, mount_point = mount_folder, extra_configs = {config : key})
```

Figure 5.2: Connect with Azure Data Lake Storage Gen 2

Our team will declare the name of the container in Azure Blob Storage to be connected, the name of the Azure Blob Storage account, the key to connect to that account, the URL, and the path to connect.

Next, our team will read the tables related to Sales:

```
header_df = spark.read.json("/mnt/Bronze/SalesOrderHeader.json")
detail_df = spark.read.json("/mnt/Bronze/SalesOrderDetail.json")
detail_df = detail_df.withColumnRenamed('SalesOrderID','detail_SalesOrderID')
detail_df = detail_df.withColumnRenamed('ModifiedDate','Date')
detail_df = detail_df.withColumnRenamed('ProductID', 'ProductCode')

df_product = spark.read.format("xml").option("rowTag", "record").load('/mnt/Bronze/Product.xml')
df_category = spark.read.format("xml").option("rowTag", "record").load('/mnt/Bronze/ProductCategory.xml')
df_subcategory = spark.read.format("xml").option("rowTag", "record").load('/mnt/Bronze/ProductSubCategory.xml')
df_customer = spark.read.format("xml").option("rowTag", "record").load('/mnt/Bronze/Customer.xml')
```

Figure 5.3: Read all Sales file

Tables related to Sales include SalesOrderHeader, SalesOrderDetail, then change column names such as SalesOrderID to detail_SalesOrderID, ProductID to ProductCode, etc. Read data files in xml format and save to DataFrame Spark. Our group proceeds to join Sales data:

```
sales_df = detail_df.join(header_df, header_df.SalesOrderID == detail_df.detail_SalesOrderID)
sales_df = sales_df.join(df_product, df_product.ProductID == sales_df.ProductCode)
```

Figure 5.4: Join Sales data

The purpose of these joins is to combine data from different sources (SalesOrderID, ProductCode) into a single DataFrame sales_df. Then we will write this data into the Sales.json file, this data file will be saved in the Bronze layer:

```
sales_df.write.format("json").mode('overwrite').save("/mnt/Bronze/Sales.json")
```

Figure 5.5: Write Sales file to Bronze

Files related to HR also do the same as Sales. After preprocessing the data (renaming columns, joining data together) and writing to the Bronze layer, the group next filters the necessary columns and loads them into the Silver folder. The tables are Fact_SalesTransaction, Fact_EmployeeShifts, Dim_Employee, Dim_SalesOrder, Dim_Shift, Dim_Product respectively.

```
hr_df1 = hr_df['BusinessEntityID', 'ShiftKey']
test_df = spark.read.json('/mnt/Bronze/Sales.json')
test_df = test_df.join(hr_df1, hr_df1.BusinessEntityID == test_df.SalesPersonID)
selected_columns = ['SalesOrderDetailID', 'SalesPersonID', 'SalesOrderID', 'ProductID', 'ShiftKey', 'UnitPrice'
sales_df = test_df.select([col(column) for column in selected_columns])
output_path = "/mnt/Silver/Fact_SalesTransaction.parquet"
sales_df.repartition(1).write.mode('overwrite').parquet(output_path)
```

Figure 5.6. Filter columns and write table Fact_SalesTransaction

```
selected_columns = ['BusinessEntityID', 'FirstName', 'LastName', 'JobTitle', 'CommissionPct', 'PayFrequency']

dim_employee = hr_df.select([col(column) for column in selected_columns])
output_path = "/mnt/Silver/Dim_Employee.parquet"
dim_employee.repartition(1).write.mode('overwrite').parquet(output_path)
```

```
selected_columns = ['BusinessEntityID','ShiftKey','SickLeaveHours','VacationHours','Gender', 'BirthDate','Rate','Date']

fact_employeeshifts = hr_df.select([col(column) for column in selected_columns])

output_path = '/mnt/Silver/Fact_EmployeeShifts.parquet'
fact_employeeshifts.write.mode('overwrite').parquet(output_path)
```

Figure 5.7. Filter columns and write table Fact_EmployeeWorkStats

```
selected_columns = ['SalesOrderID', 'OrderDate', 'TotalDue']

dim_salesorder = header_df.select([col(column) for column in selected_columns])
output_path = "/mnt/Silver/Dim_SalesOrder.parquet"

dim_salesorder.repartition(1).write.mode('overwrite').parquet(output_path)
```

Figure 5.8. Filter columns and write table Dim_SalesOrder

```
selected_columns = ['ShiftKey', 'Name', 'StartTime', 'EndTime']

dim_shift = shift_df.select([col(column) for column in selected_columns])
output_path = "/mnt/Silver/Dim_Shift.parquet"
dim_shift.write.mode('overwrite').parquet(output_path)
```

Figure 5.9. Filter columns and write table Dim_ShiftKey

```
selected_columns = ['ProductID', 'ProductName','ProductCategoryName']
dim_product = df_product_joined.select([col(column) for column in selected_columns])
dim_product.write.mode('overwrite').parquet('/mnt/Silver/Dim_Product.parquet')
```

Figure 5.10. Filter columns and write table Dim_Product

5.3. Gold layer

After completing the Silver layer, the data has been cleaned, the necessary columns selected, and saved in Parquet format. Before loading the data into the Gold layer, our team will write the metrics that were initially defined into two fact tables.

- Sales Transaction metrics:

```
ALTER TABLE [gold].[Fact_SalesTransaction]
ADD SalesAmount money
UPDATE [gold].[Fact_SalesTransaction] SET SalesAmount = OrderQty * UnitPrice
```

Algorithm 5.1: Code metric about Sales Amount

```
ALTER TABLE [gold].[Fact_SalesTransaction]
ADD Revenue money
UPDATE [gold].[Fact_SalesTransaction] SET Revenue = SalesAmount - StandardCost
```

Algorithm 5.2: Code metric about Revenue

```
ALTER TABLE [gold].[Fact_SalesTransaction]
ADD ConversionRate Decimal(10,2)
```

```
UPDATE [gold].[Fact_SalesTransaction] SET ConversionRate = (SELECT COUNT(Status) FROM
[gold].[Fact_SalesTransaction]
```

Algorithm 5.3: Code metric about ConversionRate

```
ALTER TABLE [gold].[Fact_SalesTransaction]
ADD AverageDealSize Decimal(10,2)
UPDATE [gold].[Fact_SalesTransaction]
SET AverageDealSize = (
SELECT SUM(SalesAmount) / COUNT(DISTINCT SalesOrderID)
          FROM [gold].[Fact_SalesTransaction]
      )
```

Algorithm 5.4: Code metric about Average Deal Size

```
ALTER TABLE [gold].[Fact_SalesTransaction]
ADD SalesQuota money
UPDATE [gold].[Fact_SalesTransaction]
SET SalesQuota = 6655000 WHERE Year(Date) = 2014
```

Algorithm 5.5: Code metric about SalesQuota

The Sales Quota is set annually by the group. In 2011, the Sales Quota was $5,000,000. Each year, the quota increases by 10%. By the final year, 2014, the Sales Quota will be $6,655,000.

```
ALTER TABLE [gold].[Fact_SalesTransaction]
ADD KPISales Decimal(10,2)
UPDATE [gold].[Fact_SalesTransaction]
SET KPISales = (SELECT SUM(SalesAmount) * 100 FROM [gold].[Fact_SalesTransaction] as
sub
WHERE sub.EmployeeKey = [gold].[Fact_SalesTransaction].EmployeeKey) / SalesQuota
```

Algorithm 5.6: Code metric about KPI Sales

```
ALTER TABLE [gold].[Fact_SalesTransaction]
ADD AverageNoCustomer Decimal(10,2)
UPDATE [gold].[Fact_SalesTransaction]
SET  AverageNoCustomer = (SELECT  COUNT(distinct CustomerID)/COUNT(distinct
EmployeeKey) from [gold].[Fact_SalesTransaction])
```

Algorithm 5.7: Code metric about Average Number Customer

- Employee Shift metrics:

```
ALTER TABLE [gold].[Fact_EmployeeShifts]
ADD AveragePayroll DECIMAL(10,2)
UPDATE [gold].[Fact_EmployeeShifts] SET  AveragePayroll = (Select AVG(PayrollRate)
from [gold].[Fact_EmployeeShifts])
```

Algorithm 5.8: Code about Average Payroll

```
ALTER TABLE [gold].[Fact_EmployeeShifts]
ADD Age smallint
UPDATE [gold].[Fact_EmployeeShifts] SET Age = 2011 - YEAR(Birthdate)
```

Algorithm 5.9: Code about Age

```
ALTER TABLE [gold].[Fact_EmployeeShifts]
ADD MaleRate Decimal(10,2)
```

```
UPDATE [gold].[Fact_EmployeeShifts] SET MaleRate = (Select Count(Gender) from
[gold].[Fact_EmployeeShifts] where Gender = 'M')*100 / (SELECT COUNT(EmployeeKey) from
[gold].[Fact_EmployeeShifts])
```

Algorithm 5.10: Code about Male rate

```
ALTER TABLE [gold].[Fact_EmployeeShifts]
ADD AverageAge Decimal(10,2)
UPDATE [gold].[Fact_EmployeeShifts] SET AverageAge = (SELECT AVG(Age) from
[gold].[Fact_EmployeeShifts])
```

Algorithm 5.11: Code about Average Age

After adding metrics, all of that data will then be loaded into the Gold layer. The general flow of this data loading step is shown in the following diagram:

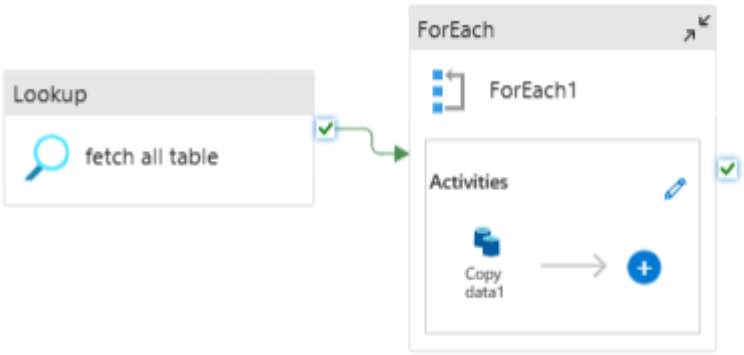


Figure 5.11. Pipeline of SQL Database to Gold layer

First, the group will use the lookup function to look up files from the SQL database (where the data is stored in Bronze format), and then the data will go through a ForEach loop. This loop contains Copy Data activities, the purpose of which is to return multiple different files.

More details about lookup function:

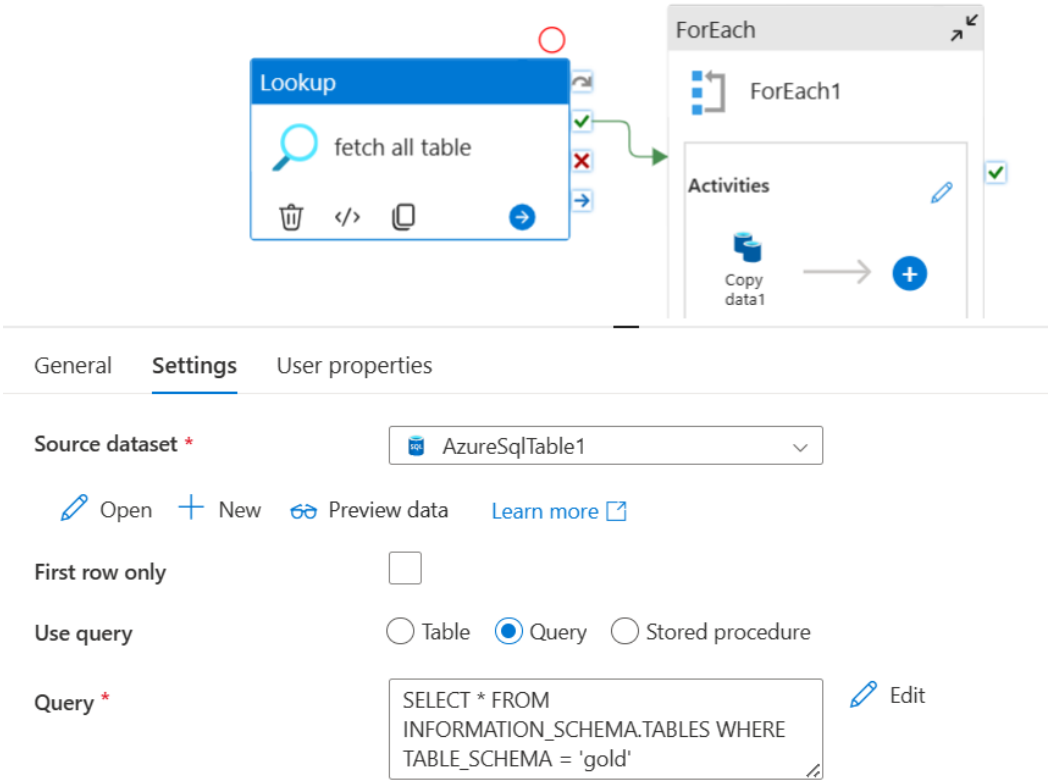


Figure 5.12. Lookup function in Gold layer

Regarding the source dataset, the designated files here are in parquet format stored in the "gold" folder using Get Metadata. Then, all files in the "gold" folder will be retrieved with the selected field list being Child items. Then, you'll configure the ForEach function. In the settings section, the team will add dynamic content in the item section. This expression will retrieve the output of the Lookup function and use its value as an input.

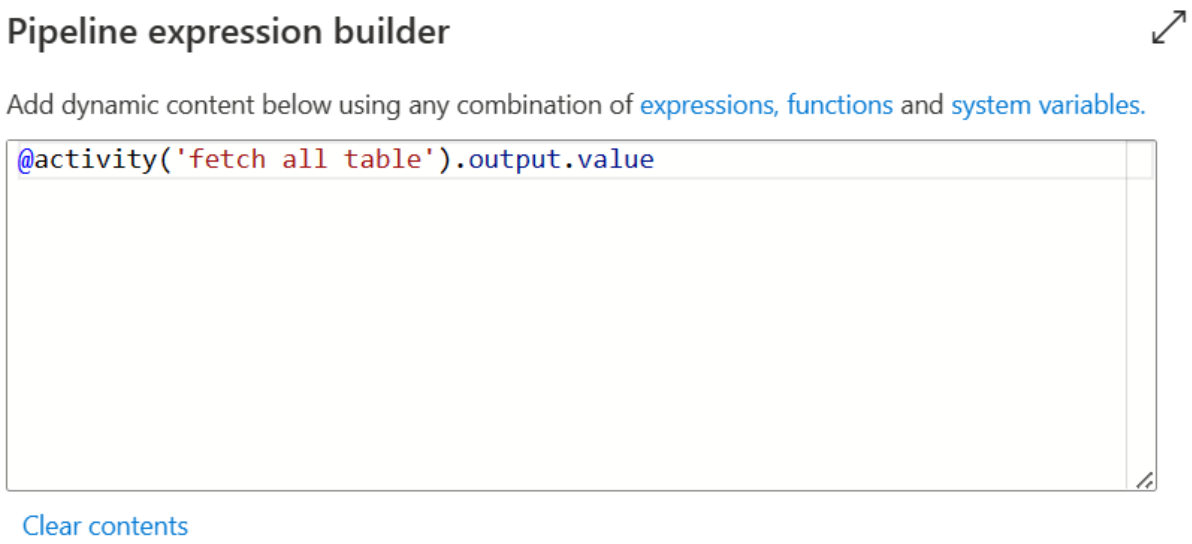


Figure 5.13: Config ForEach function

Next, you will add Copy data activities within the ForEach loop. In the Source settings step, the team will create two parameters: TABLE_SCHEMA and TABLE_NAME. In the Table selection part, you'll create two dynamic content for the schema and name. Here, the team selects "Gold" as the schema, while the tables will be all tables with the schema "Gold" in the database.

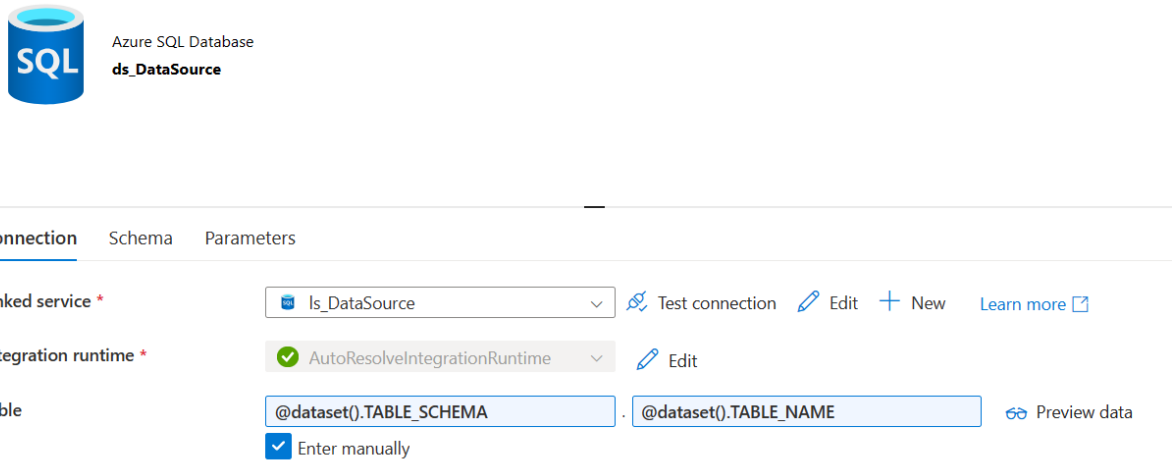


Figure 5.14: Data source of Gold layer

In the destination section of the "gold" layer, the team will create a linked service connecting to Data Storage Gen 2, specifying the location of the data as the "gold" layer. For the table selection part, the team will also create a dynamic content named FILE_NAME. It will automatically create tables with names corresponding to the tables loaded into the destination in the previous data source setting step.

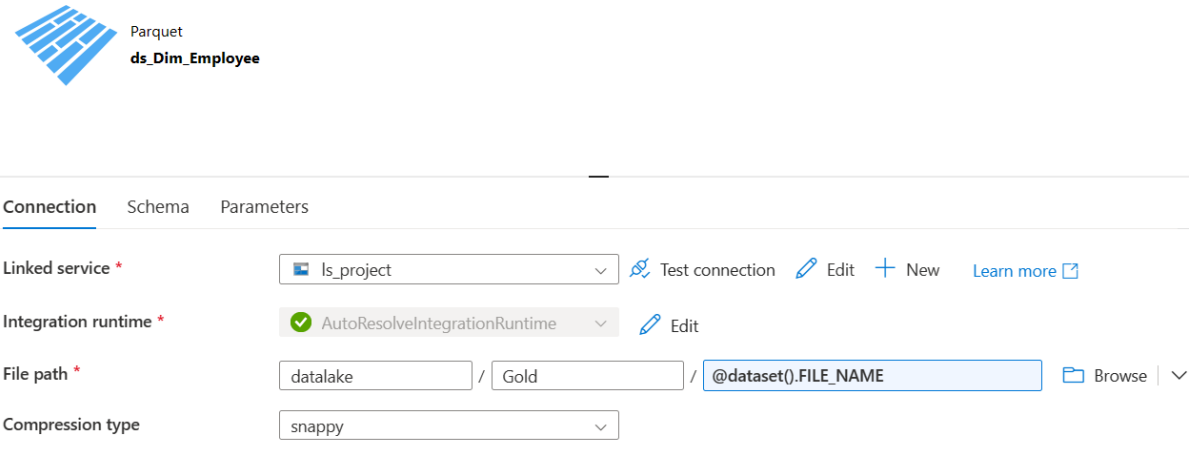


Figure 5.15: Sink of Gold layer

In the value section of the FILE_NAME variable, the team will also set a dynamic content as shown below. The result of this value will return in the format of

schema_name.table_name.parquet.

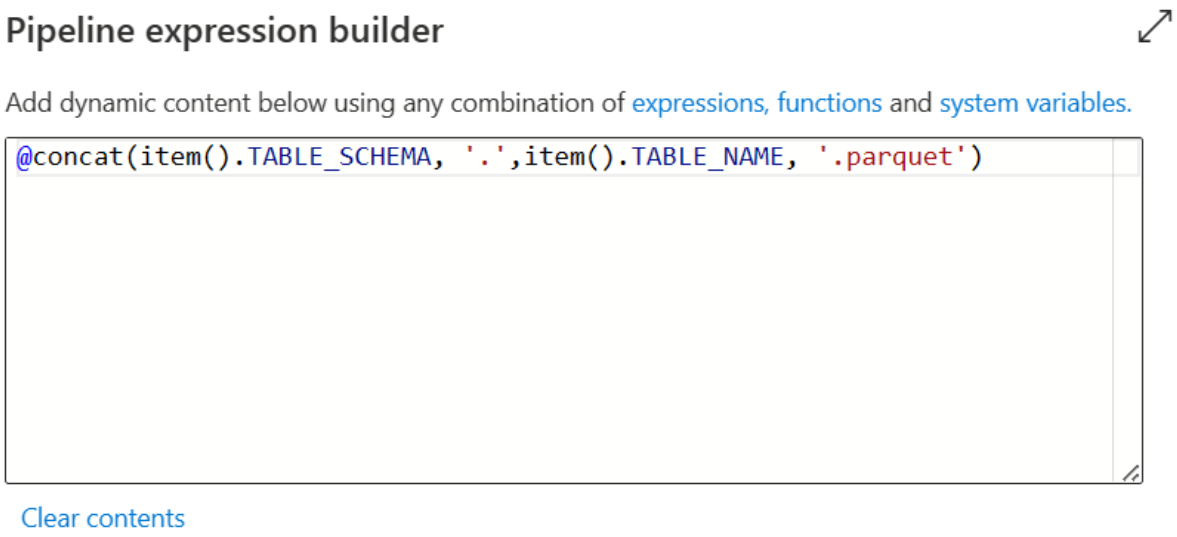


Figure 5.16: FILE_NAME parameter's value

5.4. Load data to dedicated pool

After constructing the three data layers in the Data Lake, the team will load the data from the "gold" layer into the data warehouse, which in this case is a dedicated pool. The diagram below provides an overview of the flow when loading data from the data lake into the data warehouse.

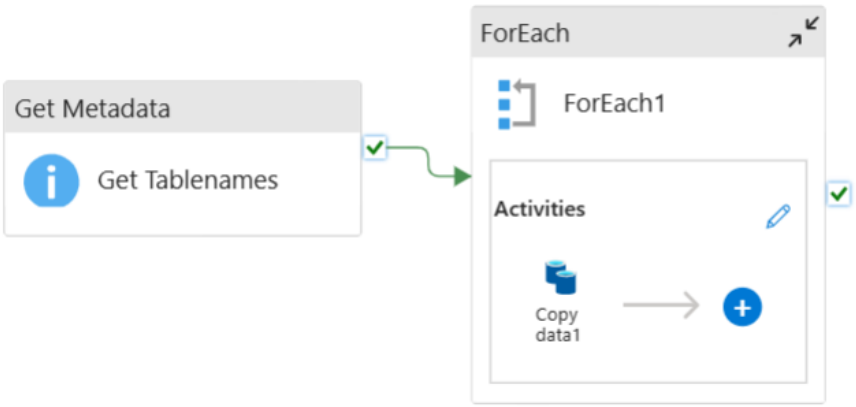


Figure 5.17: General flow load data to data warehouse

Firstly, regarding the Get Metadata function, it connects to the dataset named "ds_datalake" and retrieves metadata about the child items within this dataset. The main objective is to gather a list of files in the Azure Data Lake Storage.

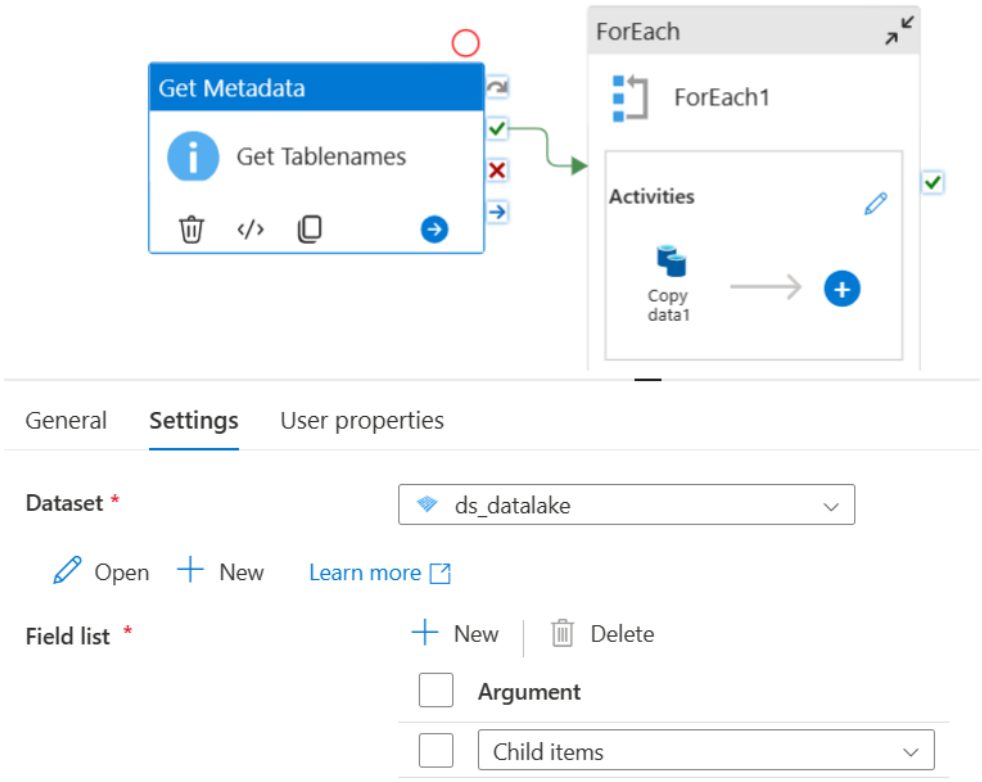


Figure 5.18: Get Metadata configuration

Next, in the ForEach activity, the team will add dynamic content in the item setting as shown in the diagram below. This expression is used to set the items that the ForEach activity will iterate over. "childItems" will contain a list of files in a "Gold" folder on Azure Data Lake Storage.

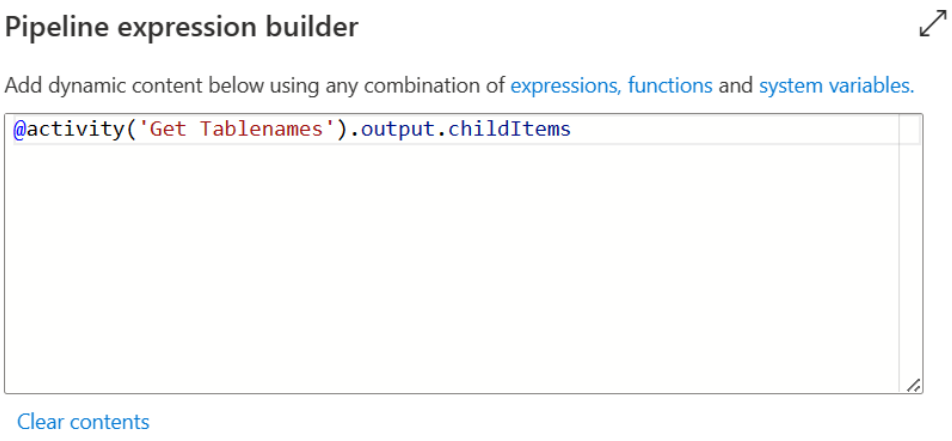


Figure 5.19: ForEach's config in Data warehouse

The team will add a Copy Data activity into the ForEach loop. In the Source settings, the team will define the data source for the Copy Data activity within the ForEach loop. It specifies that the source data is Parquet files stored in the

"datalake/Gold" directory, and the file name is dynamically determined by the parameter FILE_NAME. Within the ForEach loop, the value of FILE_NAME will change based on each item iterated through (file names from the Get Metadata activity).

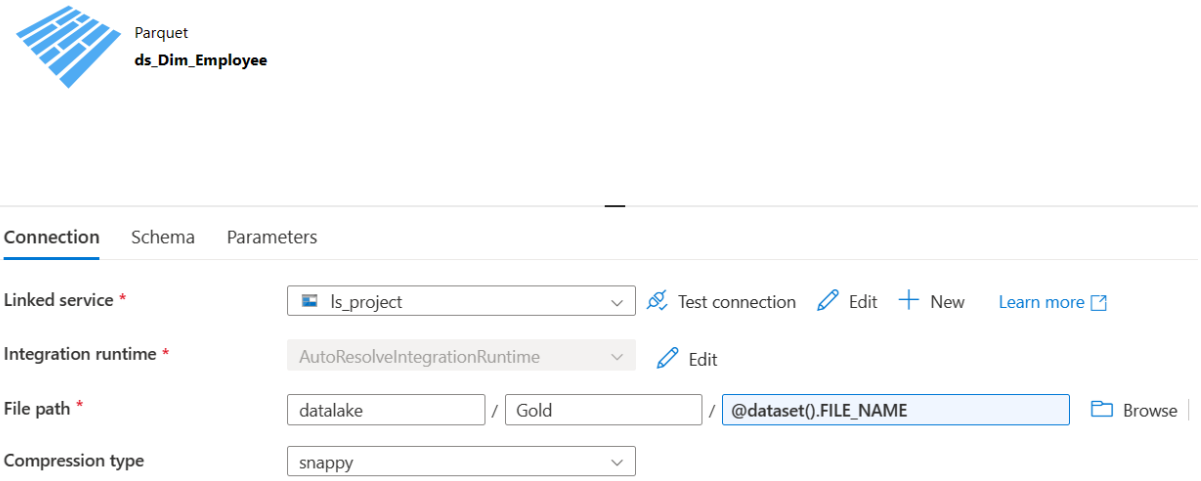


Figure 5.20: Setting Data Source for Copy Data in dedicated pool

Next, in the Copy Data activity's sink settings, the team will specify the data warehouse as the dedicated pool previously created. They will designate the schema as 'dwh' and dynamically generate the table name using a parameter named TableName.

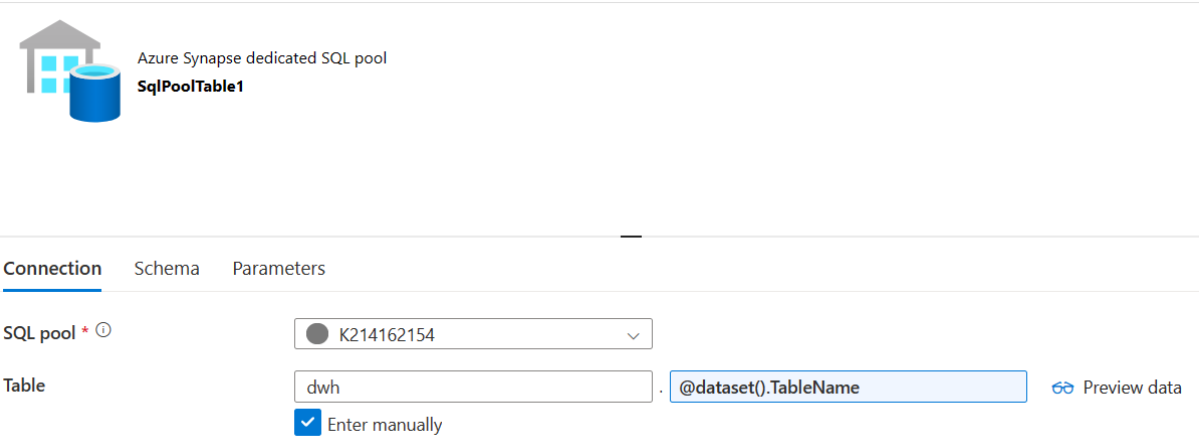


Figure 5.21: Setting Data Source in dedicated pool

In the setting for the value of the TableName parameter, the team will write a script to replace all occurrences of ".parquet" in the input file name with an empty string,

and replace the string 'gold.' in the input file name with an empty string.

Figure 5.22: Value setting for TableName parameter

This is the data stored in data warehouse:

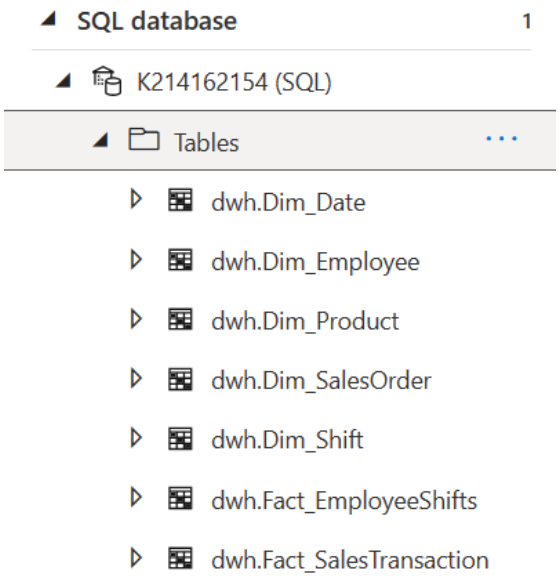


Figure 5.23: Data in Data warehouse

5.5. Create Dim_Time

After loading data to dedicated SQL pool, we created a Dim_Date table for analysis throughout time. The SQL script is below:

```
CREATE TABLE [dwh].[Dim_Date] (
   [DateKey] [int] NOT NULL,
   [Date] [date] NOT NULL,
   [Weekday] [tinyint] NOT NULL,
   [WeekDayName] [varchar](10) NOT NULL,
   [Month] [tinyint] NOT NULL,
```

```
        [MonthName] [varchar](10) NOT NULL,
        [Quarter] [tinyint] NOT NULL,
        [QuarterName] [varchar](6) NOT NULL,
        [Year] [int] NOT NULL,
        [FirstDateofYear] DATE NULL,
        [LastDateofYear] DATE NULL,
        [FirstDateofQuater] DATE NULL,
        [LastDateofQuater] DATE NULL,
        [FirstDateofMonth] DATE NULL,
        [LastDateofMonth] DATE NULL,
        [FirstDateofWeek] DATE NULL,
        [LastDateofWeek] DATE NULL);

    SET NOCOUNT ON;
    TRUNCATE TABLE [dwh].[Dim_Date];

    DECLARE @CurrentDate DATE = '2011-01-01';
    DECLARE @EndDate DATE = '2015-12-31';

    WHILE @CurrentDate < @EndDate
    BEGIN
        INSERT INTO [dwh].[Dim_Date] (
            [DateKey],
            [Date],
            [Weekday],
            [WeekDayName],
            [Month],
            [MonthName],
            [Quarter],
            [QuarterName],
            [Year],
            [FirstDateofYear],
            [LastDateofYear],
            [FirstDateofQuater],
            [LastDateofQuater],
            [FirstDateofMonth],
            [LastDateofMonth],
            [FirstDateofWeek],
            [LastDateofWeek]
        )
        SELECT
            DateKey = YEAR(@CurrentDate) * 10000 + MONTH(@CurrentDate) * 100
+ DAY(@CurrentDate),
            DATE = @CurrentDate,
            WEEKDAY = DATEPART(dw, @CurrentDate),
            WeekDayName = DATENAME(dw, @CurrentDate),
            [Month] = MONTH(@CurrentDate),
            [MonthName] = DATENAME(mm, @CurrentDate),
            [Quarter] = DATEPART(q, @CurrentDate),
            [QuarterName] = CASE
```

```
                WHEN DATENAME(qq, @CurrentDate) = 1 THEN 'First'
                WHEN DATENAME(qq, @CurrentDate) = 2 THEN 'second'
                WHEN DATENAME(qq, @CurrentDate) = 3 THEN 'third'
                WHEN DATENAME(qq, @CurrentDate) = 4 THEN 'fourth'
            END,
            [Year] = YEAR(@CurrentDate),
            [FirstDateofYear] = CAST(CAST(YEAR(@CurrentDate) AS VARCHAR(4)) +
'-01-01' AS DATE),
            [LastDateofYear] = CAST(CAST(YEAR(@CurrentDate) AS VARCHAR(4)) +
'-12-31' AS DATE),
            [FirstDateofQuater] = DATEADD(qq, DATEDIFF(qq, 0, GETDATE()), 0),
            [LastDateofQuater] = DATEADD(dd, -1, DATEADD(qq, DATEDIFF(qq, 0,
GETDATE()) + 1, 0)),
            [FirstDateofMonth] = CAST(CAST(YEAR(@CurrentDate) AS VARCHAR(4))
+ '-' + CAST(MONTH(@CurrentDate) AS VARCHAR(2)) + '-01' AS DATE),
            [LastDateofMonth] = EOMONTH(@CurrentDate),
            [FirstDateofWeek] = DATEADD(dd, -(DATEPART(dw, @CurrentDate) -
1), @CurrentDate),
            [LastDateofWeek] = DATEADD(dd, 7 - (DATEPART(dw, @CurrentDate)),
@CurrentDate);

        SET @CurrentDate = DATEADD(DD, 1, @CurrentDate);
    END;

    UPDATE [dwh].[Fact_EmployeeShifts]
    SET Date = CONVERT(DATE, Date);

    UPDATE [dwh].[Fact_SalesTransaction]
    SET Date = CONVERT(DATE, Date);

    UPDATE [dwh].[Fact_EmployeeShifts]
    SET DateKey = (
        SELECT DateKey
        FROM [dwh].[Dim_Date]
        WHERE [dwh].[Dim_Date].[Date] = [dwh].[Fact_EmployeeShifts].Date
    );

    UPDATE [dwh].[Fact_SalesTransaction]
    SET DateKey = (
        SELECT DateKey
        FROM [dwh].[Dim_Date]
        WHERE [dwh].[Dim_Date].[Date] = [dwh].[Fact_SalesTransaction].Date
    );

    ALTER TABLE dwh.Fact_SalesTransaction
    ADD DateKey INT;

    ALTER TABLE dwh.Fact_EmployeeShifts
    ADD DateKey INT;
```

Algorithm 5.12: Create Dim_Time

5.6. PowerBI Dashboards

5.6.1. Sales Report

In this section, our team will present the reports that have been developed to meet the project's goal of capturing the company's sales performance, specifically focusing on the salesperson. First, we will look at the general sales report before delving into salesperson performance.

Initially, in the Sales bookmark tab, at the top, we have cards displaying the total sales along with metrics such as conversion rate, average deal size, and average customer per salesperson. We also add some slicers for filtering necessary data, including unit, year, and product category. Moving down one row, we have two charts showing revenue by product category and sales trends over the months in different years. On a more detailed level, we break down the product categories and display them over time by month, combined with an average line to better visualize revenue trends. Finally, the table shows detailed revenue from the previous year, the current year's revenue, and the increase or decrease in them. The report reader can clearly understand the revenue information through the sales bookmark tab.

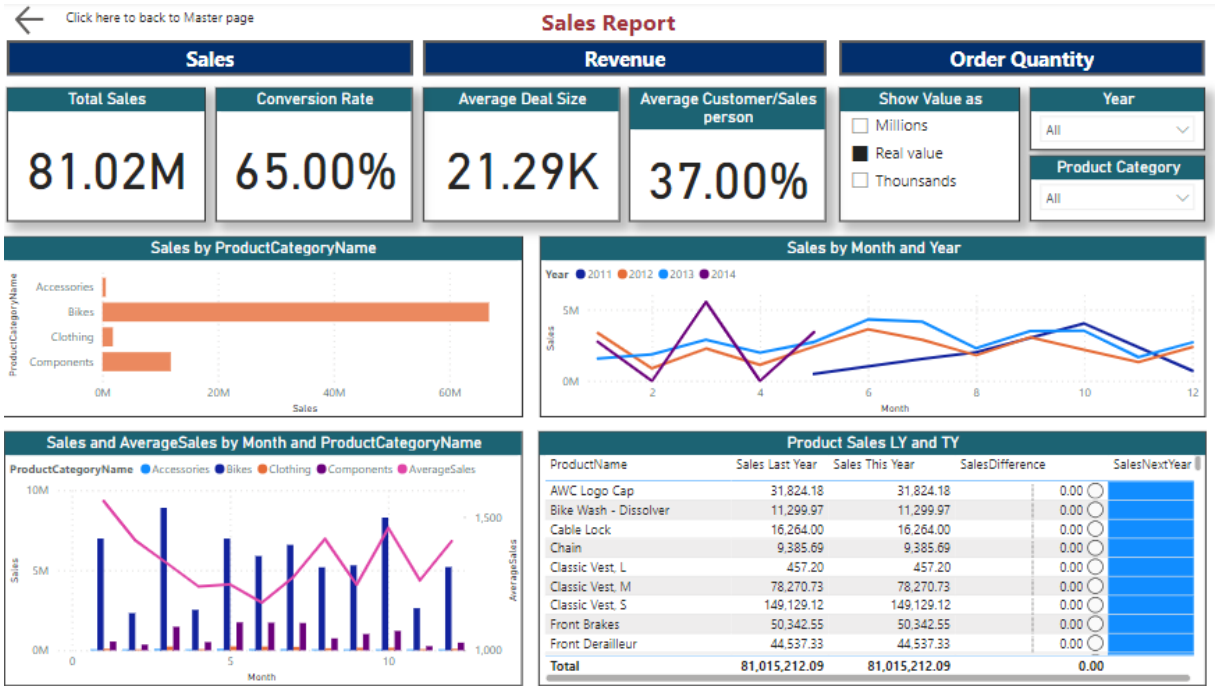


Figure 5.24. Sales report - Sales bookmark

Similarly, in the Revenue tab, it displays the total profit. Additionally, we have added a chart on the Profit Rate to assess the profitability of each product category. At

the bottom, the details are presented similarly to the Sales tab but applied to Revenue.

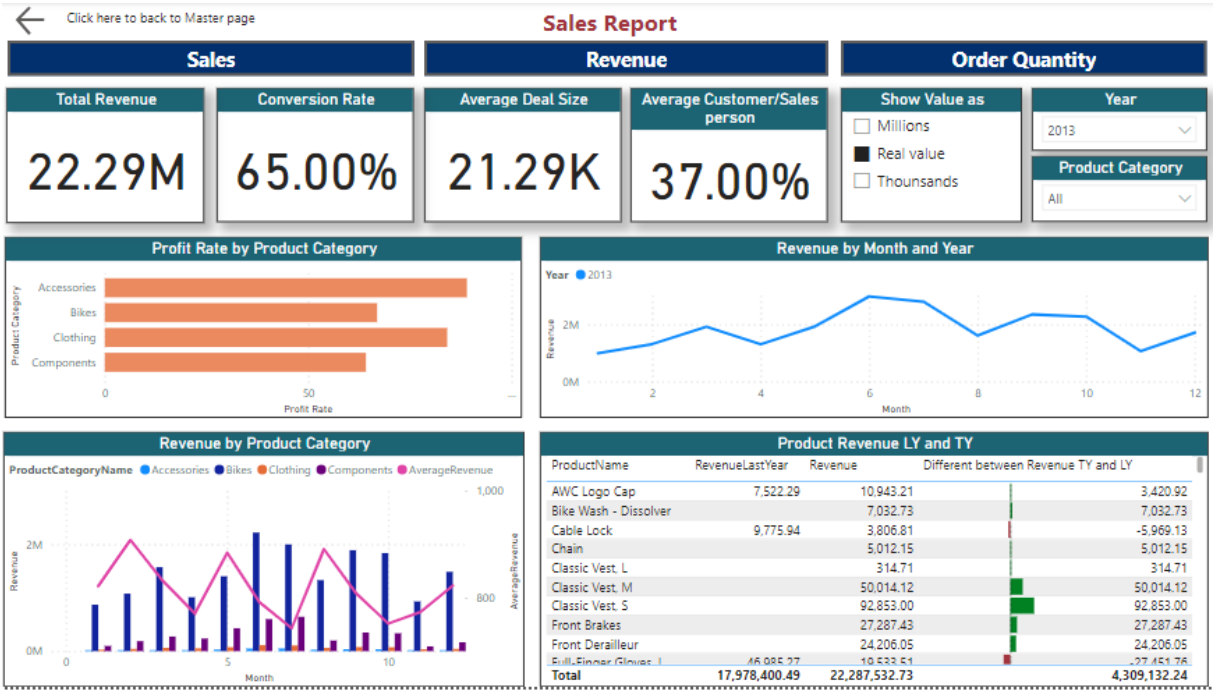


Figure 5.25. Sales report - Revenue bookmark

Ultimately, users can view product category charts to assess top-selling products and their trends.

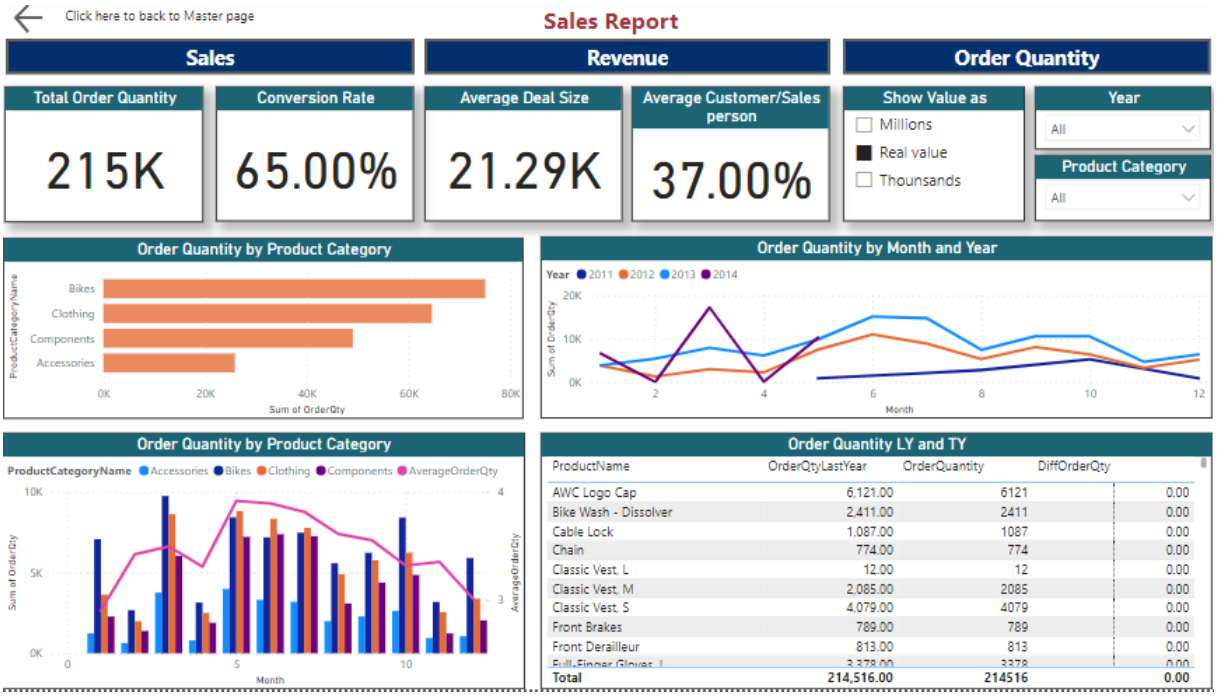


Figure 5.26. Sales report - Order Quantity bookmark

5.6.2. Salesperson Management Resources

This dashboard will focus on analyzing metrics related to salespersons. Currently, the company has a total of 17 employees in the Sales department, with an average age of 42.46, an average payroll of 27.49, and a male ratio of 55.6%. In addition to the slicers in the Sales report section (5.5.1), we have added a new slicer for job title to see the performance details of each role.

Furthermore, there are three bookmarks showing the top 5 employees with the highest sales from different perspectives: commission rate, gender, and age. Evaluating from various angles allows us to understand the factors affecting performance to some extent. Additionally, we have tracked the KPI completion rate of the top-performing salespersons, with impressive results exceeding 150%.

Below, we examine the correlation between age and commission rate to see if age is related to the commission percentage. Finally, a table displays the revenue of the employees and other metrics, such as the number of customers they acquired during that period.

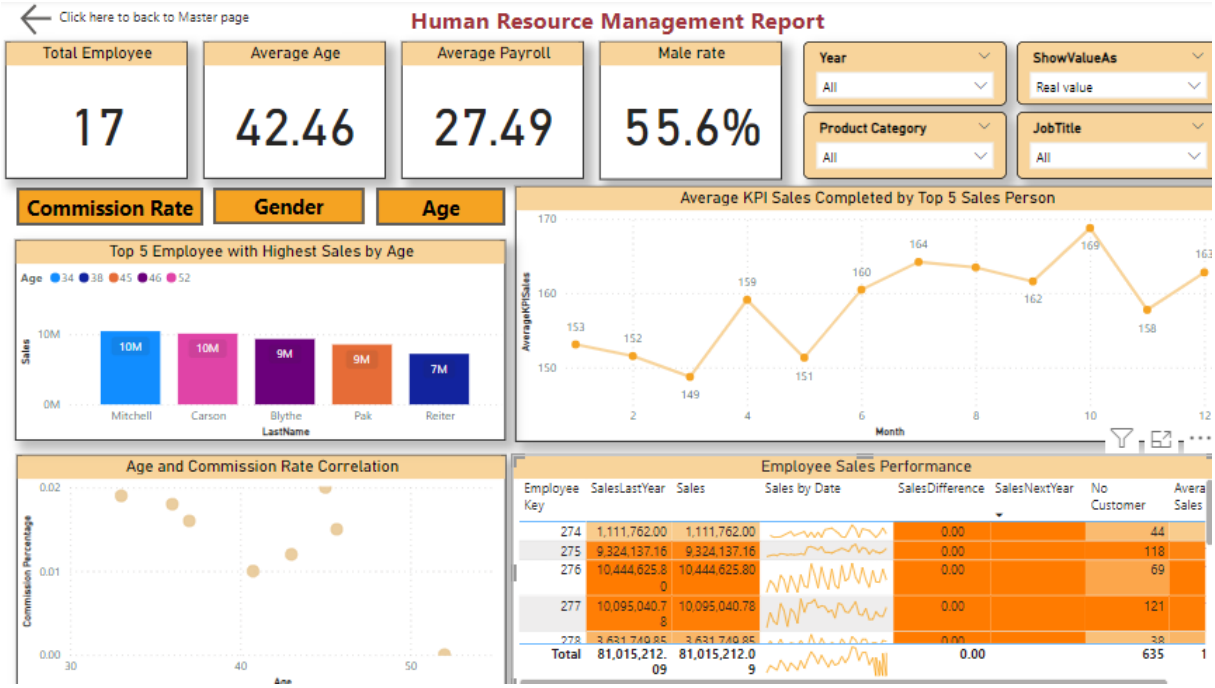


Figure 5.27. HR Report

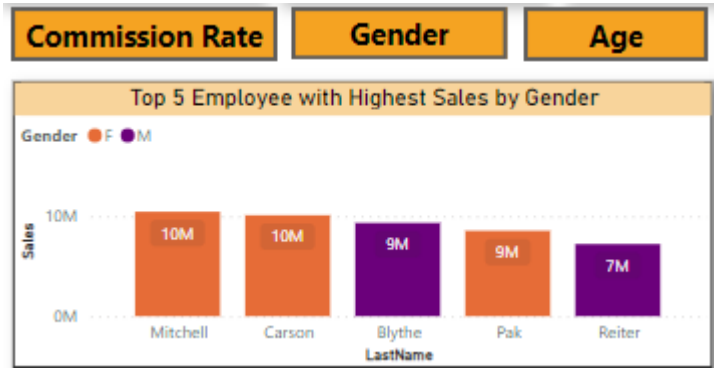


Figure 5.28. HR Report - Gender tab

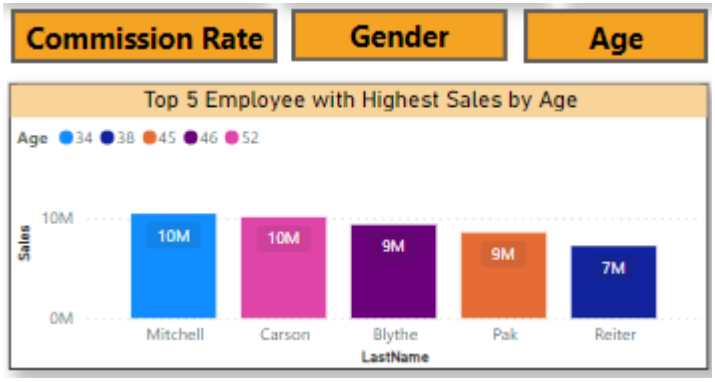


Figure 5.29. HR Report Age tab

Chapter 6. Discussion and Future Works

In final chapter, we conclude the project by bringing up the discussion of limitations, level of efficiency and optimization in solving use cases in this project as well as studying the future research directions which our team expects in next BI projects.

6.1. Limitations of Current Research

This research article only focuses on studying employees in the Sales department and their performance in AdventureWorks. The scope of this research article is in two main modules: Sales and Human Resource. Although this focused data analysis is profound, it does not consider many external influencing factors in practice. In addition, this analysis is also limited in that it can only be analyzed based on variables in the data, without expansion. Moreover, our team has not created a trigger to load data automatically every time there is a change in data, so if there is an edit to the data, the team will have to re-run each pipeline at each level manually, which takes a lot of time to edit. Therefore, the results of this research will be difficult to apply to the external business environment and to continuously changing data sources.

6.2. Future research directions

- Integrating new technology: A research focus on HR and Sales can bring a wealth of knowledge and experience in using tools like Power BI and Microsoft Azure, but technology is always updating and innovating, so in the future and in the futures we need to access more tools that can solve problems more quickly and powerfully.
- Diversify data sources: Diversifying data sources can involve collecting additional data from market information, customer feedback, and supplier data to significantly enhance the understanding of the research. This diversification can not only enrich the data but also help uncover potential factors related to Human Resource Management and Sales that may have been overlooked in this study.

- Interdisciplinary Research: Researching human resource management and Sales will become even more wonderful if it can connect with other fields such as economics, sociology and environmental science. This can help uncover how external factors influence Employees and Sales, leading to more comprehensive and sustainable solutions.
- Add trigger features to tracking incremental data: Adding trigger features to track incremental data involves implementing mechanisms that detect changes or updates in data, allowing for real-time or near-real-time processing. These triggers can be set to monitor specific conditions, such as new entries, updates to existing records, or deletions, ensuring that any data alterations are promptly captured and processed. This approach enhances data accuracy and timeliness, facilitating more responsive and dynamic data-driven applications. By leveraging these triggers, systems can efficiently manage and utilize growing datasets, providing up-to-date insights and enabling proactive decision-making.

6.3. Recommendations

6.3.1. Expand analysis of interdepartmental interactions

Sales and materials management: Strengthen the collaboration between the sales and materials management departments. Conduct a detailed analysis to understand how the availability and management of materials impact sales performance. This can lead to better inventory management, reduce stockouts or overstock situations, and improve customer satisfaction.

Sales and procurement: Enhance the interaction between the sales and procurement teams. By aligning procurement strategies with sales forecasts, the company can ensure timely availability of products and materials, which in turn can improve sales outcomes. Implementing regular meetings and integrated planning sessions can help synchronize efforts and improve overall efficiency.

6.3.2. Recruitment of younger employee

Given the high average age of the current sales team, consider recruiting younger employees to bring fresh perspectives and innovative ideas to the department. Younger employees may also be more adaptable to new technologies and methodologies, which

is crucial in a rapidly evolving business environment. Additionally, a diverse age group can lead to a more dynamic and resilient team.

6.3.3. Invest in advanced training programs:

Provide continuous training and development opportunities for the sales team to keep them updated with the latest sales techniques, market trends, and technological advancements. Advanced training programs can include workshops on data analytics, customer relationship management (CRM) systems, and digital marketing strategies. This will not only improve individual performance but also contribute to the overall growth of the department.

References

[1] Athukorala, C., Kumarasinghe, H., Dabare, K., Ujithangana, P., Thelijjagoda, S., & Liyanage, P. (2020, November). Business intelligence assistant for human resource management for IT companies. In *2020 20th International Conference on Advances in ICT for Emerging Regions (ICTer)* (pp. 220-225). IEEE.

[2] Sherman, R. (2014). *Business intelligence guidebook: From data integration to analytics*. Newnes.

[3] L'Esteve, R. C. (2023). Designing a Secure Data Lake. In The Cloud Leader's Handbook: Strategically Innovate, Transform, and Scale Organizations (pp. 183-201). Berkeley, CA: Apress.

[4] Knight, D., Pearson, M., Schacht, B., & Ostrowsky, E. (2020). Microsoft Power BI Quick Start Guide: Bring your data to life through data modeling, visualization, digital storytelling, and more. Packt Publishing Ltd.

[5] Ribeiro, A., Amaral, A., & Barros, T. (2021). Project Manager Competencies in the context of the Industry 4.0. Procedia computer science, 181, 803-810.

[6] Microsoft Azure. Introduction to Azure Data Factory. Retrieved May 7, 2024 from: https://learn.microsoft.com/en-us/azure/data-factory/introduction

[7] Izhar, A. (2024). Azure Synapse Analytics: A Complete Overview. Retrieved May 5, 2024 from: https://k21academy.com/microsoft-azure/data-engineer/azure-synapse-analytics/

[8] Richart, P. (2024). Microsoft Azure Tutorial for Beginners: Learn Basics in 1 day. Retrieved May 7, 2024 from: https://www.guru99.com/microsoft-azure-tutorial.html