Data warehouse implementation

Neha

BTech, Computer Science Engineering

Dronacharya College of Engineering, MDU Rohtak, Haryana

Abstract

In the modern digital landscape, data serves as the backbone of strategic business decisions. This paper presents the design, implementation, and analysis of a data warehouse built using the Medallion Architecture to support business intelligence and analytics. The project consolidates and transforms raw enterprise data into actionable insights through a structured ETL pipeline across Bronze, Silver, and Gold layers. PostgreSQL was chosen as the core database engine, and the project incorporated tools such as pgAdmin, GitHub, Draw.io, and Notion. Analytical outcomes included customer behavior segmentation, product performance insights, and sales trend analyses. The system was designed for scalability, performance, and future integration with visualization tools and cloud platforms.

Keywords: Advanced SQL, Query Optimization, ETL Pipelines, Dimensional Modeling, Star Schema, Data Visualization Planning, Medallion Architecture

Introduction

The exponential growth of digital data across industries has created unprecedented opportunities for leveraging data-driven strategies in decision-making, innovation, and customer engagement. From e-commerce to healthcare, financial services to manufacturing, organizations are generating and collecting massive volumes of data on a daily basis. However, the ability to extract meaningful insights from this data is often constrained by the limitations of traditional transactional databases. These systems, while optimized for handling operational transactions, fall short when subjected to complex, cross-temporal analytical queries that require high levels of aggregation, filtering, and dimensional slicing.

To address these limitations, data warehouses (DWs) have emerged as foundational components of modern analytics infrastructure. A data warehouse is a centralized repository designed to store integrated, historical, and subject-oriented data that is structured specifically to support business analysis and reporting. Unlike operational databases, data warehouses are optimized for read-heavy workloads, enabling analysts and decision-makers to run ad-hoc queries, generate dashboards, and perform trend analyses with greater efficiency and consistency.

In the contemporary data ecosystem, businesses engage with a broad array of data sources. These include internal enterprise systems like Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM) systems, as well as external sources such as web analytics platforms, Internet of Things (IoT) sensors, mobile applications, and third-party APIs. Each of these data streams brings its own data structure, update frequency, and quality characteristics. Managing such heterogeneous and high-velocity data sources requires a robust, scalable, and modular architectural approach to data engineering.

The Medallion Architecture, pioneered by Databricks, offers a highly effective methodology for organizing data pipelines into

distinct layers based on the data's quality and transformation stage. This architecture divides the data flow into three logical layers:

- Bronze Layer: This layer stores raw, unprocessed data ingested directly from source systems. It serves as a persistent landing zone that retains the original data in its native form for traceability and audit purposes.
- Silver Layer: In this stage, the raw data is cleaned, standardized, enriched, and validated to create a consistent and reliable version suitable for downstream use. Business rules and transformation logic are applied here to enhance data quality and usability.
- Gold Layer: This is the business-ready data layer, where refined datasets are organized into dimensional models such as star schemas or snowflake schemas. It supports high-performance analytics and reporting and forms the basis for business intelligence tools and machine learning models.

By separating concerns across these layers, the Medallion Architecture promotes reusability, maintainability, and scalability of data pipelines. It also ensures that each stage of data processing is well-defined and independently verifiable, enabling better governance and lineage tracking.

This research paper aims to provide a comprehensive demonstration of the design and implementation of a scalable data warehouse system following the Medallion Architecture. The implementation leverages open-source technologies, including PostgreSQL for database management, and focuses on achieving the following key objectives:

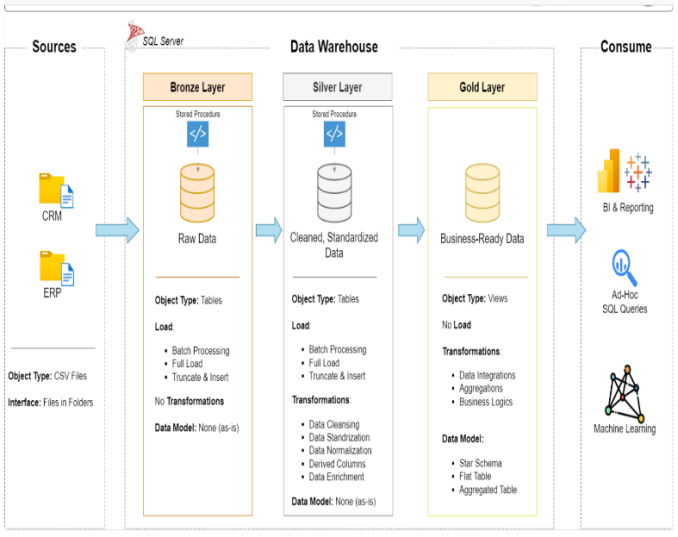
- Ensuring high data quality through structured ETL (Extract, Transform, Load) pipelines.
- Designing reusable and modular SQL transformation scripts.
- Optimizing system performance through indexing, partitioning, and query tuning.
- Preparing the system for seamless integration with business intelligence platforms such as Power BI and Tableau.
- Enabling future extensibility for advanced analytics and predictive modeling.

Methodology

In addition to the layered structure, the methodology included iterative testing and validation at each ETL phase. The Bronze Layer data was first profiled using descriptive statistics to understand patterns, anomalies, and null distributions. In the Silver Layer, business logic was encoded as SQL transformation scripts to ensure consistency across dimensions. Automated scripts were used for repeatable cleansing operations.

Special care was taken to define surrogate keys in the dimension tables to avoid direct dependency on operational identifiers, which can change over time. Star schema design also involved analyzing query patterns to optimize join paths and indexing strategies. Finally, the system was benchmarked using a sample dataset to evaluate performance and ensure scalability.

2.1 Medallion Architecture Overview



Medallion Architecture divides the data processing pipeline into three core layers:

- Bronze Layer (Raw Data): Contains unprocessed data ingested from CSV files simulating ERP and CRM systems.
- Silver Layer (Cleaned Data): Involves standardization, cleansing, and validation of data to ensure consistency and quality.
- Gold Layer (Business-Ready Data): Implements a Star Schema comprising fact and dimension tables for fast, business-centric queries.

This layered approach ensures data integrity, scalability, and optimized performance for reporting and analytics.

2.2 Tools and Technologies

The implementation of the scalable data warehouse system was greatly facilitated by a carefully curated set of tools and technologies. Each tool was selected based on its robustness, ease of use, and ability to support various phases of data engineering and project management. Below is a detailed description of the tools and their specific roles within the project:

- **PostgreSQL**: PostgreSQL served as the core relational database management system (RDBMS) for this project. It was selected due to its open-source nature, strong community support, ACID compliance, and extensive feature set, including advanced SQL functionalities, indexing options, and extensibility via custom functions. PostgreSQL hosted all three layers of the Medallion Architecture—Bronze, Silver, and Gold—offering a consistent environment for ETL operations, data modeling, and analytics. Its support for stored procedures, window functions, and JSON data types made it highly suitable for processing both structured and semi-structured data.
- **pgAdmin**: pgAdmin is a popular open-source graphical user interface (GUI) tool for managing PostgreSQL databases. It was used extensively for administrative tasks such as creating databases, defining schemas, writing and executing SQL queries, and managing user roles and permissions. The intuitive interface and built-in tools for query analysis and performance monitoring made pgAdmin an invaluable resource for both development and debugging. Additionally, its ability to visually inspect data models and browse database objects accelerated the implementation of the data warehouse.
- **GitHub**: GitHub played a central role in version control, code collaboration, and project documentation. All SQL scripts, schema definitions, and transformation logic were managed in Git repositories, allowing team members to collaborate efficiently and maintain a complete history of changes. Issues and pull requests were used to manage tasks and peer reviews, promoting best practices in collaborative software development. GitHub also served as a backup and central storage location for project artifacts, reducing the risk of data loss and improving traceability.
- **Draw.io (diagrams.net)**: Draw.io, also known as diagrams.net, was used for creating a variety of architectural and design diagrams. These included Entity-Relationship Diagrams (ERDs), Data Flow Diagrams (DFDs), and ETL workflow visualizations. Such diagrams were essential during the planning and design stages, as they provided a clear visual representation of data relationships, process flows, and transformation pipelines. The diagrams also served as reference documents during review meetings and documentation.
- **Notion**: Notion was adopted as the central platform for project planning, team coordination, and documentation. It supported task tracking through kanban boards and to-do lists, while

also allowing for collaborative note-taking, meeting minutes, and knowledge sharing. By maintaining a living document of the project's progress, Notion facilitated transparency and alignment among team members. Its integration with GitHub and ability to embed diagrams and tables made it a versatile tool for managing all non-code aspects of the project lifecycle.

Implementation

Each layer was implemented through a combination of SQL scripts and stored procedures. The ETL pipeline was developed to run in a scheduled batch mode, ensuring daily updates to the Gold Layer. Triggers and logging mechanisms were implemented to ensure traceability of data lineage.

Visualization diagrams created using Draw.io served as blueprints for team collaboration. These included Data Flow Diagrams (DFDs), Entity-Relationship Diagrams (ERDs), and transformation workflows. These diagrams were essential during design reviews and iterative improvement cycles.

3.1 Data Ingestion and Bronze Layer

Raw ERP and CRM datasets in CSV format were imported into PostgreSQL using SQL scripts. These datasets included sales, customer information, and product inventories. The Bronze layer stored this data without transformation.

3.2 Silver Layer Transformations

In this layer, key data quality tasks were performed:

- Cleansing: Removal of duplicates, handling null values, and correcting inconsistent data types.
- Standardization: Uniform formats for dates, currencies, and units.
- Normalization: Adhered to business rules and ensured referential integrity.

3.3 Gold Layer and Star Schema Design

The Gold layer implemented a star schema composed of:

- FactSales: Containing transactional sales data.
- Dimension Tables: Including DimCustomer, DimProduct, DimTime, and DimRegion.

Results and Discussion

The project also revealed opportunities for A/B testing and product bundling strategies. By identifying customers who frequently purchased related items, recommendations could be personalized based on past buying behavior. These insights can be further enhanced by integrating predictive analytics models.

Advanced queries were written using window functions to rank customers and segment them dynamically. Heatmaps of product sales across regions could be derived from the geographic data, assisting in territory planning and localized advertising campaigns.

4.1 Customer Behavior Analysis

- Identified top 10% of customers responsible for ~50% of sales.
- Segmented buyers based on frequency and value.
- Uncovered regional and seasonal purchasing patterns.

4.2 Product Performance Evaluation

- Highlighted top-performing products based on revenue and units sold.
- Identified geographic demand variations.

- Analyzed underperforming products for discontinuation or strategic promotions.

4.3 Sales Trend Analysis

- Detected seasonal peaks (e.g., holidays) and sales dips.
- Monthly analysis revealed underperforming periods, aiding promotional planning.
- Longitudinal data helped track overall business growth trajectory.

Conclusion

In summary, this data warehouse system is more than just a repository—it acts as a data ecosystem that can power future machine learning initiatives by serving as a trusted source of truth. The lessons learned from this implementation provide a strong foundation for building enterprise-grade analytics systems.

This project successfully demonstrated the end-to-end development of a data warehouse using the Medallion Architecture, integrating various tools and applying best practices in data engineering. The structured ETL process ensured data accuracy and accessibility, while the analytical models provided actionable insights into customer behavior, product performance, and sales trends.

The Gold Layer's star schema enabled fast querying for business reporting, laying the groundwork for advanced analytics and business intelligence.

Future Scope

An additional enhancement involves incorporating real-time data streaming capabilities using tools like Apache Kafka or AWS Kinesis. This would make the warehouse suitable for operational intelligence use cases such as fraud detection or supply chain monitoring.

Implementing DataOps practices such as CI/CD pipelines for data scripts, automated testing, and anomaly detection can further increase the reliability and maintainability of the data warehouse.

Several enhancements are planned to expand the warehouse's utility:

- Integration with BI Tools: Connecting Power BI or Tableau for interactive dashboards. - Incremental Data Loads: Enable near-real-time data updates instead of batch refreshes. - Cloud Deployment: Migrate to AWS RDS or Azure SQL for scalability and enterprisegrade availability.

References

1. Microsoft SQL Server Documentation – https://learn.microsoft.com/en-us/sql/sqlserver/
2. Draw.io (diagrams.net) – https://app.diagrams.net
3. dbdiagram.io – https://dbdiagram.io
4. Kaggle Dataset (ERP + CRM Simulation) – https://www.kaggle.com
5. W3Schools SQL Tutorial – https://www.w3schools.com/sql/
6. PostgreSQL Official Documentation – https://www.postgresql.org/docs/
7. Medallion Architecture by Databricks – https://www.databricks.com/glossary/medallion-architecture
8. pgAdmin Documentation – https://www.pgadmin.org/docs/
9. GitHub Docs – https://docs.github.com/en
10. Notion Documentation – https://www.notion.so/help
11. Kimball, R., & Ross, M. (2013). The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling. Wiley.
12. Inmon, W. H. (2005). Building the Data Warehouse. Wiley.

13. ETL Best Practices by Microsoft Azure – https://learn.microsoft.com/en-us/azure/architecture/data-guide/relational-data/etl
14. Data Warehouse Design Principles – https://docs.oracle.com/cd/E11882_01/server.112/e25554/tdpdw_design.htm
15. Stack Overflow Community Discussions on SQL and Data Warehousing – https://stackoverflow.com/
16. Apache Kafka Documentation – https://kafka.apache.org/documentation/
17. Tableau Official Help – https://help.tableau.com/
18. Power BI Documentation – https://learn.microsoft.com/en-us/power-bi/