E-Commerce Data Warehouse using PostgreSQL

Sanbeer Shafin
University of Calgary
Calgary, Canada
sanbeer.shafin@ucalgary.ca

Jon Choi
University of Calgary
Calgary, Canada
jongeak.choi@ucalgary.ca

Zuhaer Rahman
University of Calgary
Calgary, Canada
shadeen.rahman@ucalgary.ca

Reese Sanchez
University of Calgary
Calgary, Canada
jaiden.sanchez@ucalgary.ca

***Abstract*—**This paper presents the design, implementation, and evaluation of a PostgreSQL-based analytical data warehouse for e-commerce style retail data using the Tableau Superstore dataset(provided by Salesforce). A full dimensional architecture was constructed, consisting of a central fact table capturing sales, quantity, discount, and profit metric being supported by customer, product, date, geography and ship mode dimension tables. The data processing workflow follows an ELT pattern in which raw CSV data is first staged and then transformed into a star schema warehouse through a structured series of cleaning, standardization, and surrogate key generation steps. Analytical methods include a suite of OLAP queries involving multi attribute aggregations, time based aggregations, profitability analysis, and category level comparisons. To assess analytical performance, identical OLAP workloads were executed on both the original non star-schema dataset and the dimensional warehouse across macOS, Windows, and Linux systems. Quantitative results consistently show that the star schema reduces runtime for most aggregation intensive queries due to improved join structure, reduced redundancy and enhanced data locality. This case study demonstrates that, even within a traditional row store engine such as PostgreSQL, dimensional modeling can meaningfully improve analytical clarity and performance, highlighting its real world relevance for organizations seeking scalable, SQL-based analytical systems.

Keywords*—Datawarehouse, PostgreSQL, E-Commerce, Analytics, Star Schema, OLAP, OLTP, ETL/ELT,.*

I. Introduction

The relentless expansion of electronic commerce (e-commerce) has brought both challenges and opportunities to the table. On one hand, the digitalization of the storefront led to streamlined logistics between the consumer and the manufacturer, massive scalability due to freedom from physical presence, access to a broader consumer base, and greater access to data for business insights. On the other hand, these advantages assume the flawless operation of complex digital infrastructure. Services may overload and fail entirely, as seen with Amazon in 2004 [7]; data may be breached through insecure APIs or malicious cyber attacks [8]; or incorrect data handling may result in disastrous operational mistakes [9]. As such, it is crucial that the system architecture serving as the backbone of an e-commerce business be scalable, reliable and robust.

Within this landscape, the Data Warehouse (DWH) serves as a critical component, integrating various data sources into a repository optimized for analysis. The data warehouse ingests data from operational systems handling real-time transactions, performing necessary cleansing and transformations (Extraction, Transformation, Loading - ETL) to prepare the data for analytical reporting. This insight is vital for reliability; even if an operational system is technically scalable, scaling decisions made only after a system fails due to overload lack true reliability. Conversely, scaling up massively without demand is highly inefficient.

In this paper, we utilize Tableau's Superstore Dataset to construct a data warehouse system using PostgreSQL - an open and well understood database system that is cost effective. We demonstrate a warehouse architecture that addresses three key aspects - scalability, reliability and robustness - essential for a robust e-commerce analytical backbone.

II. Background & Theory

Traditional Online Transaction Processing (OLTP) systems, which strictly prioritize principles of atomicity, consistency, insolation and durability to support transactional integrity and update performance, struggle in analytical workloads such as profitability measurement, customer segmentation, and operational forecasting. Early relational database research, including foundational work on POSTGRES, emphasized extensibility and transactional guarantees but acknowledged the need for specialized mechanisms to support emerging analytical requirements at scale [1] . Contemporary studies confirm that OLTP engines face significant performance degradation when executing large aggregations or full table scans, motivating the use of data warehouses and dimensional modeling in modern data architectures [3] .

A. Relational SQL, NoSQL, Distributed Database Foundations and Data Lakehouse

Relational databases such as PostgreSQL remain central to transactional e-commerce systems due to their ACID properties, structured schemas, and consistency guarantees. However, comparative analyses of SQL, NoSQL, and NewSQL systems highlight that relational engines exhibit limited scalability for analytics once data volumes extend into multi gigabyte or terabyte ranges i.e Big Data.[3] .

NoSQL(Not Only SQL) and distributed systems offer horizontal scaling, flexible schemas, and high throughput thus making them suitable for semi structured data such as product metadata or log streams which are generally stored

in JSON(Java Script Object Notation) format. Yet these systems often lack the strong consistency required for accurate analytics. As a result, hybrid architectures where OLAP relational systems in data warehouses coupled with the flexibility of NoSQL are slowly becoming widely adopted in the format of Data Lakehouse.

B. Data Warehousing Principles: OLTP vs OLAP

Data warehouses(DW) are optimized for Online Analytical Processing (OLAP), supporting large scale queries, aggregations, and historical analysis. While OLTP(Online Transactional Processing) databases generally store data in highly normalized structures to preserve integrity, DWs employ denormalized schemas, pre aggregations, and optimized indexing strategies to accelerate analytical performance. Empirical evaluations of PostgreSQL as a warehouse engine demonstrate that OLTP systems suffer substantial slowdowns when used for analytical reporting, whereas warehouse oriented designs reduce query latency and enhance throughput [2] .

Although Data warehouses are generally denormalized in order to prioritize analytics over the accuracy of the data - we will be using the Star Schema with its fact tables and dimensions tables in order to have both normalized accurate data and apply large scale analytics to our chosen dataset.

Typical OLAP workloads in e-commerce scenarios can include revenue analysis, shipping performance, product profitability, customer segmentation, and time-series demand forecasting.

C. Dimensional Modeling and the Star Schema

Dimensional modeling provides a structured framework for designing analytical databases, and the star schema is one the most widely adopted architecture for data warehouse systems alongside snowflake schema. In a star schema, analytical data is organized around a central append only fact table that stores quantitative metrics such as sales, profit, discount, and quantity. This fact table is linked to multiple descriptive dimension tables that capture deeper context. Prior research shows that the star schema simplifies query execution, reduces join complexity, and significantly improves aggregation performance in OLAP systems [2] .

The star schema is particularly effective in e-commerce analytics, where business questions require slicing and aggregating data across multiple dimensions. Domain entities such as customers, products, dates, shipping categories, and geographic regions naturally map into dimension tables that support hierarchical analysis and drill down operations.

D. Normalization, Denormalization and Redundancy Control in Date Warehouses

A core design principle of data warehousing is achieving the right balance between normalization and denormalization. In OLTP systems, schemas are typically normalized to eliminate redundancy, ensure data integrity, and minimize update anomalies. However, normalized schemas introduce numerous tables and complex joins, making them inefficient for analytical workloads that require large aggregations and historical trend analysis. As such, Data warehouses on PostgreSQL can intentionally adopt a hybrid approach to ensure data accuracy and analytics since:

1. Dimensions are normalized to reduce redundancy

Dimensions such as Customer, Product, Geography, and Date are stored once in dedicated tables eliminating repeated descriptive information across millions of transactions. For example:

- Customer names, segments, and regions are stored once in DimCustomer.
- Product categories and descriptions reside in DimProduct.
- Calendar attributes are stored in DimDate.

This removes redundancy, lowers storage cost and ensures that updates (e.g., product category changes) occur in a single location.

2. Fact tables are denormalized to improve analytical performance

The central fact table contains foreign keys pointing to each dimension but stores all numerical metrics directly such as Sales, Profit, Quantity and Discount. Denormalization here avoids repeated multi table joins required in fully normalized OLTP schemas. Research demonstrates that this design significantly accelerates OLAP workloads, especially in PostgreSQL based warehouses where reduced join depth leads to faster scans and aggregations [2] .

Thus, the star schema achieves an optimal balance between full normalization and full denormalization by combining the strengths of both approaches into a single analytical model. Dimension tables remain normalized to avoid redundancy, ensure consistency and maintain clean attribute hierarchies allowing customer, product, geographic and time related information to exist in a single authoritative place. At the same time, the fact table is intentionally denormalized, storing all quantitative metrics and foreign keys together so that analytical queries can be executed with minimal join depth and maximum scan efficiency. This hybrid design is particularly advantageous for e-commerce analytics, where large volumes of historical transactions must be aggregated across multiple perspectives such as product category, customer segment and region. By reducing repeated storage of descriptive attributes while still enabling high performance analytical queries, the star schema unifies the benefits of normalized data integrity with the computational efficiency of denormalized fact structures, making it one of the most effective and widely adopted models for data warehousing.

E. ETL/ELT and Dataflow for Upstream and Downstream Purposes

A data warehouse sits between upstream OLTP systems and downstream BI tools, acting as the central hub for clean, structured data. In the upstream flow, data is extracted from transactional databases and moved into the warehouse

using either ETL or ELT pipelines. ETL is preferred when data requires significant cleaning, normalization, and restructuring before analysis, especially when the transformed form will not need to be reverted later, which is common in e-commerce environments where OLTP schemas differ heavily from analytical ones. Prior work shows that performing these transformations before loading greatly simplifies warehouse queries and improves performance in PostgreSQL based systems [2] . ELT, on the other hand, is useful when data may undergo multiple transformations or iterative refinement, allowing raw data to be loaded first and transformed inside the warehouse as analytical needs evolve.

Downstream, the data warehouse provides a consistent and analytics ready foundation for dashboards, reporting tools and visualization systems. Research on e-commerce warehouse monitoring demonstrates that dimensional schemas directly enhance dashboard responsiveness and operational insight generation [4]. Through this dual role; upstream ingestion and downstream data visualization, the data warehouse enables efficient, reliable, and scalable e-commerce analytics.

III. Methods and Pipelines

The group collaboratively explored the dataset from multiple analytical perspectives, with each member experimenting independently with different OLAP queries to validate the flexibility of the star schema. One member focused on time-series and regional sales trends, another examined product level profitability and category based aggregation, while a third analyzed customer segmentation and discount behavior. Additional exploration included shipping performance and fulfillment patterns using the Ship Mode and Geography dimensions. This distributed approach allowed the team to test the warehouse design from diverse analytical angles, confirming that the dimensional model supported efficient slicing, dicing, drill down, and roll up operations across various business domains and correlating questions that may arise.

We attempted to implement an end-to-end data engineering pipeline for both the non-star-schema database and the star-schema dimensional warehouse, enabling a controlled comparison of their analytical behavior. The process began by loading the raw transactional dataset into a database structured in the original OLTP style. This involved constructing a single wide table (orders_no_star_schema) containing all order, customer, product, temporal, and geographic attributes. During ingestion, we resolved data type inconsistencies, standardized date formats, and ensured that all relevant transactional fields were preserved for downstream analysis.

A parallel warehouse was then built using dimensional modeling principles. In a separate PostgreSQL database, we designed the complete star schema, consisting of a central line item fact table (fact_sales) and multiple normalized dimension tables, including dim_customer, dim_product, dim_date, dim_geography and dim_ship_mode. The ELT process developed for this warehouse extracted records from the flat OLTP style table, generated surrogate keys, normalized categorical attributes, populated dimension tables and loaded the fact table with the appropriate foreign-key relationships to maintain referential integrity and analytical consistency.

After both systems were fully populated, we executed a suite of OLAP analytical queries on each table. These queries evaluated time-series sales trends, category-level performance, regional revenue patterns, customer-segment profitability and shipping-mode behavior. Equivalent queries were run across the two designs to compare aggregation accuracy, query complexity, result clarity, and processing behavior. This comparative analysis enabled a clear assessment of how dimensional modeling affects analytical efficiency relative to a flat, non-star-schema structure.

A. Data Processing and ELT Workflow

The data processing workflow adopted in this project follows an Extract-Load-Transform (ELT) model. All raw transactional records were first extracted from the operational database and loaded directly into a staging table within the analytical PostgreSQL environment. This staging table served as an immutable landing zone that preserved the original structure and content of the source data. Using ELT ensured that the system maintained a complete and reproducible snapshot of the operational dataset, allowing all transformation logic to occur inside the warehouse without requiring repeated data exports or external preprocessing. This process is visually represented in the ELT pipeline architecture diagram (see Fig. 1), which shows the movement of data from the staging layer into the dimensional structures.

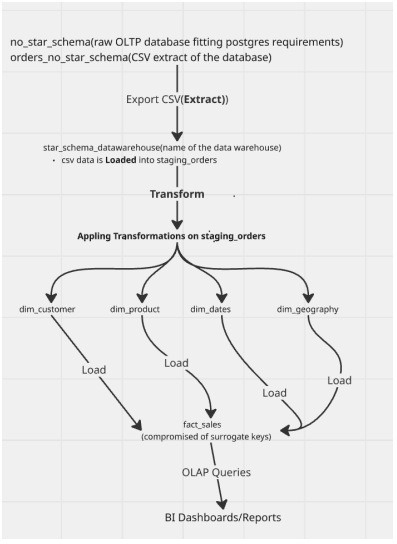

Figure 1: Dataflow

After loading the raw data into staging, the transformation phase generated the dimensional structures required by the star schema. This involved selecting distinct business entities to populate the customer, product, date, geography, and shipping-mode dimensions; standardizing attribute values; and assigning surrogate keys to support referential integrity. The fact table was then produced by joining staging records against these dimensions and

replacing natural keys with their corresponding surrogate identifiers. The resulting star schema structure is illustrated in Fig. 2, highlighting the central fact table and its relationships to the surrounding dimensions. This approach was well suited to the dataset and analytical objectives, as the source data was already clean and structured, reducing the need for complex pre load processing. Additionally, performing all transformations after loading provides flexibility: the warehouse can be rebuilt, extended, or refined entirely within PostgreSQL. Overall, the ELT approach offered a transparent, efficient, and repeatable mechanism for constructing the analytical warehouse used to compare the star-schema and non-star-schema designs.

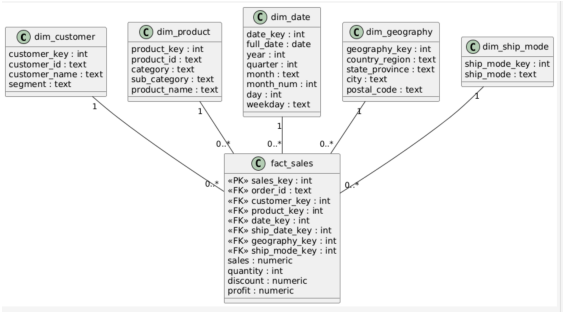

Figure 2: Star schema

B. *Possibly Using a Python Based ETL/ELT Pipeline*

Although the data warehouse in this project was populated manually through SQL based ELT operations, it is entirely feasible to automate the workflow using a Python based ETL pipeline. In a production setting, Python scripts using libraries such as pandas and psycopg2 or orchestration tools like Airflow, would typically extract data directly from a live transactional system, perform the required transformations and load the results into the dimensional warehouse on a scheduled basis. In our case, a Python ETL pipeline was not implemented because the dataset was not available as a live operational stream; instead, all records were contained in a static CSV file of roughly ten thousand rows. Given the small dataset size and lack of real time data, directly loading the CSV into a staging table and performing the transformations inside PostgreSQL was more efficient and easier to manage. However, the structure of the warehouse and the staged ELT workflow could easily be automated through Python if continuous ingestion or larger scale data processing were required.

IV. OLAP Queries Performed: Motivations and Explanations

The OLAP queries selected for this study were designed to reflect analytical tasks that are routinely required in modern e-commerce environments. These tasks span temporal analysis, multidimensional profitability analysis, customer valuation, product performance ranking, and trend detection. By executing these analytical workloads on both the non–star-schema database and the star-schema warehouse, the study evaluates how each database design supports common decision support operations. The goal is not merely to compute numerical results, but to assess the structural advantages of dimensional modeling particularly with respect to query clarity, computational efficiency, and suitability for high performance OLAP processing. The following subsections outline the motivation behind each set OLAP query used in the evaluation.

A. *Monthly Sales and Profit Analysis (Query Set 1)*

Time series analysis is foundational in e-commerce analytics, as organizations rely on monthly sales and profit trends to understand seasonality, detect anomalies, and support operational forecasting. Computing these summaries on the non-star-schema table requires applying repeated date functions across a large transactional dataset, which increases computational cost and query complexity. In contrast, the star schema structures temporal data within the date dimension, enabling clean key based joins and efficient roll ups across year and month hierarchies. This query set was chosen to evaluate how effectively each schema supports time-series OLAP operations.

B. *Multidimensional Profit Aggregation Across Region, Category and Shipping Mode (Query Set 2)*

Understanding how profit varies across multiple business dimensions simultaneously is essential for supply chain optimization and strategic planning. Analyzing profitability across regions, product categories and shipping modes provides insight into market performance, logistical efficiency, and product mix alignment. When performed on the non-star-schema database, this multidimensional aggregation requires grouping on repeated textual attributes, leading to increased I/O cost and slower execution. The star schema consolidates each of these attributes into dedicated dimension tables, supporting efficient slicing and dicing using surrogate keys. This query set evaluates the benefits of dimensional organization for multidimensional OLAP workloads.

C. *Customer Profitability Ranking (Query Set 3)*

Customer profitability ranking is widely used in customer segmentation, lifetime value estimation and marketing strategy. Identifying high-value customers requires grouping and ordering by total profit, which can be computationally demanding in the non–star-schema dataset due to repeated customer attributes appearing across all order rows. The star schema eliminates this redundancy by storing customer metadata in a central dimension table, enabling ranking operations to rely on stable surrogate keys rather than repeated strings. This query set was included to assess how well each schema supports customer-centric analytical tasks.

D. *Product and Subcategory Ranking Using Window Functions (Query Set 4)*

Product performance analysis often requires comparing items within the same subcategory or category using advanced OLAP operations such as window functions. Executing partitioned ranking on the non–star-schema table involves grouping on large textual product descriptors, which increases processing time and

memory usage. The star schema reduces this overhead by enabling partitioning based on surrogate keys, with product names and categories stored only once in the product dimension. This query set evaluates how dimensional modeling enhances efficiency and interpretability for product-level OLAP computations.

E. Month-over-Month Profit Growth Analysis (Query Set 5)

Short term trend detection, such as month-over-month (MoM) profit growth, is critical for operational monitoring and strategic decision making. On the non–star-schema database, MoM calculations require repeated date truncation, sorting, and window operations over the full transactional table. In the star schema, these computations are simplified through the structured temporal hierarchy stored in the date dimension, enabling efficient sequential comparisons across months and years. This query set was selected to examine how effectively each schema handles time-windowed OLAP analysis.

Collectively, the OLAP query sets used in this study represent analytical patterns commonly employed in e-commerce decision support environments. Their selection provides a comprehensive basis for comparing the non–star-schema database with the star-schema data warehouse. Each query category isolates a specific class of analytical operation from temporal analysis, multidimensional aggregation, customer ranking, product performance evaluation and trend detection thereby demonstrating how dimensional modeling can improve query expressiveness, reduces redundancy, and enhances computational performance. This structured evaluation motivates the use of the star-schema design for scalable and maintainable OLAP workloads.

V. Results & Evaluation

To evaluate the analytical performance of the star schema data warehouse relative to the non-star-schema database, a series of OLAP queries were executed across multiple computing environments. Tests were conducted on macOS, Windows, and Linux systems featuring a wide range of hardware specifications, including configurations with varying RAM capacities, SSD sizes, CPU architectures, and cache characteristics. Each OLAP query set was executed multiple times per system, and average runtimes were computed to ensure consistency. The results, summarized from the execution logs in the test document , indicate that the star-schema data warehouse generally achieved lower execution times across most platforms, reflecting the efficiency gains associated with surrogate-key joins, reduced data redundancy and the use of compact dimension tables. However, one high-end Linux system exhibited atypical behavior in which the non–star-schema table performed faster, a case examined in the subsequent discussion. Overall, the aggregated results demonstrate that dimensional modeling provides measurable performance benefits for OLAP workloads under typical hardware conditions.

A. Results on Mac OS

The performance evaluation was first conducted on a macOS 26.1 system equipped with an Apple M2 chip, 16 GB of unified memory and 256 GB of SSD storage. The OLAP test queries were executed repeatedly on both the non–star-schema database and the star-schema data warehouse to obtain stable averaged runtimes. The numerical results obtained from these trials are summarized in **Table 1** and a visual representation of the averaged runtimes for all five query experiments is provided in **Fig. 3**, which compares the non–star-schema and star-schema execution times on this hardware platform.

As shown in both the table and the accompanying chart, the star-schema warehouse demonstrated consistently faster execution for most of the performed OLAP aggregations. For query sets 1, 2, 3, and 5 - the star schema outperformed the non–star-schema model by reducing scan overhead, minimizing redundant text-based grouping and relying on surrogate-key joins across compact dimension tables. This behavior is consistent with expected OLAP performance characteristics, as dimensional modeling reduces I/O pressure and improves cache locality during analytical scans.

One exception was observed in Test 4, where the non -star-schema results showed a noticeably higher execution time. This anomaly is visible in **Fig. 3**, where query set 4 exhibits a substantial performance deviation relative to the other test cases. The elevated runtime is likely attributable to the nature of the queryset, as repeated trials on other operating systems also reproduced this behavior. Even with this outlier, the aggregated results clearly show that the star-schema warehouse is generally more efficient for OLAP workloads on this hardware.

Query Set	Run for No-Star (ms)				Run for Star (ms)			
	Run 1	Run 2	Run 3	Avg	Run 1	Run 2	Run 3	Avg
1	101	96	68	89	72	77	63	71
2	80	67	70	72	62	62	62	62
3	76	69	81	75	70	63	73	69
4	74	74	79	76	120	125	75	107
5	56	106	80	81	71	72	67	70

Table 1

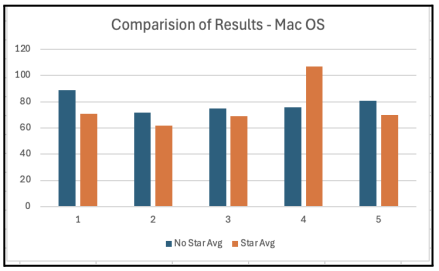

Figure 3

B. *Results on Windows OS*

The second set of performance evaluations was conducted on an Intel Chip Windows 10 system equipped with 16 GB of RAM and 1.14 TB of SSD storage. As with the macOS experiment, each OLAP query set was executed multiple times on both the non–star-schema database and the star-schema data warehouse to obtain stable averaged runtimes. The measured values are summarized in **Table 2**, while **Fig. 4** presents a visual comparison of the averaged runtimes for all five query sets.

Across the Windows platform, the results show a clear performance advantage for the star-schema warehouse in most cases. For Query Sets 1, 2, 3, and the star schema executed relatively & marginally faster than the non-star-schema model, reflecting the expected benefits of dimensional modeling. By reducing redundant text-based grouping operations and relying on compact dimension tables with surrogate-key joins, the star-schema design significantly lowered execution times, particularly in Query Set 2, where the non–star-schema average was **252 ms**, compared to **97 ms** for the star schema. This represents the most pronounced improvement observed on Windows.

As such, the overall results from the Windows system indicate that the star-schema warehouse provides marginally efficient OLAP query execution across the majority of analytical workloads. The improvements observed in this environment further reinforce the advantages of dimensional modeling, especially on hardware configurations where memory constraints and disk I/O have greater impact on query performance.

Query Set	Run for No-Star (ms)				Run for Star (ms)			
	Run 1	Run 2	Run 3	Avg	Run 1	Run 2	Run 3	Avg
1	125	101	112	113	106	111	102	107
2	508	140	106	252	95	94	101	97
3	114	108	110	111	126	111	102	114
4	425	415	461	434	427	430	424	427
5	93	94	91	93	112	107	119	113

Table 2

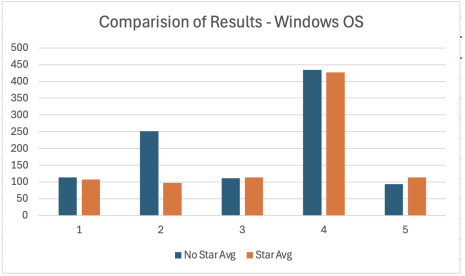

Figure 4

C. *Results on Linux OS - Manjaro*

A third set of performance evaluations was conducted on a Linux 6.6.107-1 Manjaro system equipped with an Intel i5-1034G4 processor, approximately 7.7 GB of usable RAM and a 467 GB SSD. Unlike the macOS and Windows tests, each OLAP query set was executed **30 times** on this platform. This decision reflects common industry practice, as production grade analytical databases and data warehouses are frequently deployed on Linux systems, making a more extensive trial set appropriate for obtaining stable measurements. The averaged runtimes for both schemas are reported in **Table 3**, and the corresponding performance comparison is shown in **Fig. 5**.

The Linux results further reinforce the performance advantages of the star-schema warehouse. For Query Sets 1, 2, 3, and 5 the star schema significantly outperformed the non–star-schema model. The improvements were especially pronounced in Query Sets 1 and 5, where the non–star-schema table produced averages of 535 ms and 510 ms, compared to 162 ms and 144 ms for the star schema. These results reflect the impact of dimensional modeling in reducing scan size, grouping complexity, and memory pressure - factors that are particularly relevant on Linux systems optimized for efficiency and concurrency.

As with the macOS and Windows results, Query Set 4 again showed atypical behavior. On this platform, the non–star-schema execution averaged 106 ms, compared to 149 ms for the star schema. This marks the third instance in which Query Set 4 deviates from the general trend, indicating that this particular query structure may benefit from the flat-table layout when executed under certain grouping and filtering conditions. Despite this isolated anomaly, the performance gap in all other query sets is substantial and strongly favors the star schema.

Overall, the Linux evaluation supported by 30 trial runs per query set provides the most robust evidence of the benefits of dimensional modeling. The star-schema warehouse consistently delivered lower execution times across nearly all OLAP workloads, confirming its suitability for analytical processing in environments that mirror real-world industry deployments.

Query Set	Run for No-Star (ms), 3 out of 30				Run for Star (ms), 3 out of 30			
	Run 1	Run 2	Run 3	Avg	Run 1	Run 2	Run 3	Avg
1	612	525	491	535	155	168	118	162
2	359	381	437	390	174	150	148	205
3	445	363	401	392	129	112	121	141
4	101	103	105	106	148	139	156	149
5	523	516	513	510	159	161	135	144

Table 3

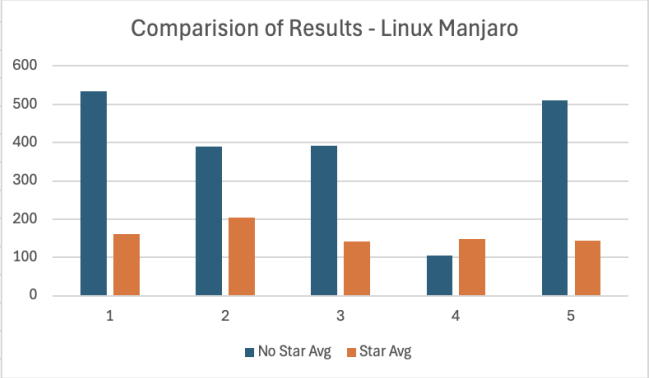

Figure 5

D. *Results on Hyper-Specialized High Memory Linux OS*

A final experiment was performed on a specialized Linux 6.17.8-zen1-1-zen system equipped with exceptionally high memory capacity (approximately 125 GB of RAM), a 4 TB SSD, and an AMD 5700X3D processor. As with the previous Linux test, each OLAP query set was executed **30 times** on both schemas to obtain stable averages. The measured results are summarized in **Table 4**, and the performance comparison is presented in **Fig. 6**.

Unlike the other operating environments, this system consistently showed faster runtimes for the non–star-schema table across all five query sets. For example, in Query Set 1 the non–star-schema average was 83 ms, compared to 150 ms for the star schema, and a similar pattern was observed across the remaining queries. This reversal of the expected behavior appears to be attributable to the hardware characteristics of the system. With more than 120 GB of RAM and a large CPU cache, the entire dataset, containing only ~10 columns can fit comfortably into memory, enabling near-instantaneous access to rows without the overhead of joining multiple tables. In this scenario, the star schema introduces additional join operations without providing meaningful reductions in scan size, resulting in slower performance.

This outcome illustrates an important practical consideration: hardware architecture can influence the relative performance of OLAP designs and extreme high-memory configurations may reduce the typical advantages of dimensional modeling. Even so, these results emphasize the importance of conducting performance testing in the specific deployment environment, as unconventional hardware configurations may yield unexpected behaviors in real-world analytic workloads.

Query Set	Run for No-Star (ms), 3 out of 30				Run for Star (ms), 3 out of 30			
	Run 1	Run 2	Run 3	Avg	Run 1	Run 2	Run 3	Avg
1	81	71	69	83	149	126	118	150
2	89	82	70	82	249	165	167	218
3	73	71	71	74	169	121	151	148
4	95	92	93	74	175	143	145	151
5	102	99	109	90	165	138	143	150

Table 4

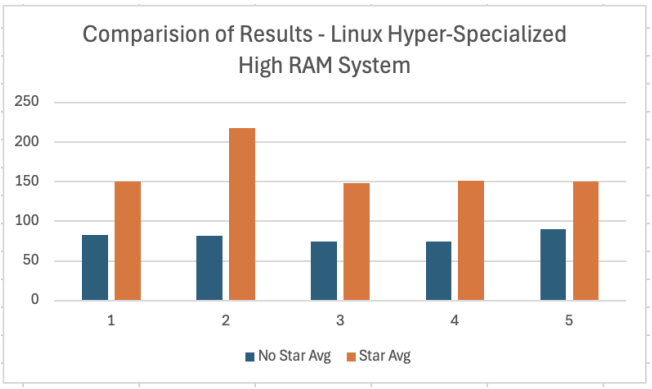

Figure 6

VI. Discussion

The experimental results collected across four different operating environments demonstrate that the star-schema data warehouse consistently improves OLAP query performance under typical hardware conditions. On macOS, Windows, and standard Linux-Manjaro systems, the star-schema design outperformed the non–star-schema dataset in the majority of the test cases, particularly for multidimensional aggregations and time-series analyses. These improvements were attributable to reduced data redundancy, smaller dimension tables, and more efficient grouping operations enabled by surrogate key joins. The quantitative differences observed in these environments reinforce the established advantages of dimensional modeling for analytical workloads.

A notable exception occurred in Query Set 4, where all systems(regardless of operating systems)showed faster performance on the non–star-schema table. This suggests that the computational pattern of this query benefits from a flat row-store structure when the grouping keys align closely with existing physical data layout. The anomaly was further amplified on the high memory Linux system, where the non–star-schema dataset significantly outperformed the star schema across all query sets. In this environment, the combination of more than 120 GB of RAM and a large CPU cache allowed the entire dataset to reside fully in memory, eliminating the usual advantages of dimensional denormalization. With sufficient memory bandwidth and cache locality, row-store access became near-instantaneous, while the star-schema model incurred additional join overhead that outweighed its structural benefits.

In terms of storage efficiency, the star-schema warehouse demonstrated measurable reductions in redundancy by isolating descriptive attributes into dimension tables. Although the fact table remained the largest structure, the separation of repeated product, customer, and geographic

fields into dimensions reduced total storage and improved data maintainability. This stands in contrast to the non–star-schema table, where textual attributes are duplicated across every row, increasing storage overhead and complicating updates or corrections. Even on high-memory systems where performance is not affected by redundancy, the storage benefits of dimensional modeling remain relevant for long term maintainability.

The warehouse also provided improved analytical consistency, as surrogate keys ensured referential integrity and eliminated issues arising from inconsistent text fields or missing values. Accuracy was preserved across all environments, with both schemas producing identical numerical results. However, the star schema's structured temporal hierarchy and explicit relationships reduced the risk of user error when writing complex analytical queries, which contributes to better analytical reliability in organizational settings.

From a scalability perspective, the star schema offers clear advantages for large-scale analytical workloads. As datasets grow and begin to exceed cache and memory capacities, the benefits of dimensional modeling become increasingly pronounced. Fact tables can be partitioned by date or region, and dimension tables remain small and indexable even as the transactional volume grows. In contrast, fully denormalized tables scale poorly: repeated textual fields produce larger I/O scans, reduced cache efficiency, and significantly slower group-by operations. The star schema is therefore better suited for cloud or distributed warehouse platforms such as PostgreSQL on managed services, Spark SQL, or columnar engines like Amazon Redshift.

Regarding fault tolerance, relational data warehouses provide durable ACID guarantees but do not inherently support distributed failover unless paired with replication or clustering mechanisms. NoSQL systems such as MongoDB or Cassandra offer more native horizontal scaling and fault-tolerance capabilities but typically sacrifice strong relational semantics, complex joins, and cross-dimensional consistency. For analytical workloads where dimensional relationships must be preserved, SQL-based warehouses remain the preferred design. NoSQL warehouses may provide superior resilience in distributed environments but require additional effort to model multi-dimensional data without native join support.

Overall, the experiments highlight both the strengths and the limitations of the star schema approach. Under conventional hardware conditions representative of most enterprise environments, the star-schema data warehouse delivers superior OLAP performance, improved storage efficiency, and more consistent analytical modeling. The outlier behavior observed on a highly specialized, high memory Linux system underscores that data architectures must be evaluated within the context of the target deployment environment. While dimensional modeling offers clear structural advantages, extreme hardware configurations can alter performance trade offs, reinforcing the importance of environment specific testing when designing real world analytical systems.

VII. Limitations

The findings presented in this study are subject to several limitations related to the dataset, the experimental methodology and the broader applicability of the results. These constraints should be considered when interpreting the performance outcomes and when generalizing them to real world analytical environments.

A. Dataset Related Limitations

The first limitation concerns the nature of the dataset used to construct the data warehouse. The source data was obtained from a static CSV extract rather than a live operational system. As a result, the experiment does not reflect the complexities associated with high velocity data ingestion, late-arriving facts, incremental loads, or real time update patterns commonly encountered in production environments. Additionally, the dataset consists of approximately 10,000 rows, which is significantly smaller than the scale typically observed in enterprise analytical warehouses, where fact tables often contain millions or billions of records. Although useful for controlled experimentation, a dataset of this size cannot fully demonstrate the performance behavior, indexing strategies, or partitioning benefits that emerge at larger scales.

A further limitation is that the data warehouse designed in this study is tailored specifically to the structure and semantics of the Tableau Superstore dataset. While the star schema approach is broadly applicable, the exact schema, dimensions, and transformation logic used here may not generalize to other organizations with different business processes, granularity requirements, or data models. In real-world environments, data engineers and analysts must develop domain-specific schemas that align with the company's operational workflows, KPIs, and data governance requirements. The warehouse implemented in this project should therefore be viewed as an instructional example rather than a universal design.

B. Experiment Related Limitations

Experimentally, the study is limited by the number and variety of hardware configurations used for performance testing. Although tests were conducted across macOS, Windows, and two Linux systems - the sample remains insufficient to draw definitive conclusions about performance across all computing environments. Additional testing on cloud-based platforms, distributed warehouse engines, and columnar storage systems would be necessary to fully validate the observed trends.

Furthermore, some trials were inadvertently affected by caching effects or residual memory states from previously executed queries. Because the datasets were small enough to fit entirely into memory, repeated execution of similar queries may have benefited from warm caches, reducing the

accuracy of cold start performance measurements. Without explicit control of caching, disk flushing, and query planning states, the measured runtimes may not perfectly reflect real world conditions where query workloads vary significantly over time.

C. General Structural Limitations

Finally, the study does not replicate the complexities of an industry level deployment. The dataset used is Salesforce's Superstore dataset prepared for Tableau and it's a high-quality, curated instructional dataset designed for demonstration and visualization purposes of Tableau. It lacks the irregularities, data quality issues, schema drift, source system inconsistencies, and operational constraints present in real enterprise environments. Moreover, the warehouse and queries evaluated here do not incorporate production considerations such as concurrency handling, workload management, fault tolerance mechanisms, or incremental refresh strategies.

Given these limitations, the results of this study should be interpreted as indicative of general trends rather than definitive performance guarantees. While the star schema demonstrated significant performance advantages under most tested conditions, further experimentation with larger, more diverse, and operationally realistic datasets is required before drawing broader and absolute conclusions applicable to industry scale analytical systems.

VIII. Future Work

Future work in this area and for this project would benefit from expanding the scope, scale, and architectural diversity of the analytical environments tested in this study. With additional time and financial resources, a primary direction for extension would be to evaluate the dimensional data warehouse using significantly larger datasets, ideally ranging from tens of millions to billions of rows. Larger datasets would enable more realistic testing of indexing strategies, partition pruning, memory utilization, columnar compression, and long running OLAP workloads that our current 10,000 row CSV extract cannot meaningfully stress. Such experimentation would provide a clearer understanding of how star schema performance scales under conditions that resemble industry grade analytical workloads.

Another direction for future exploration involves benchmarking the warehouse against modern cloud native analytics systems. Platforms such as AWS Redshift, Google BigQuery, Snowflake, and Azure Synapse Analytics offer cost effective, columnar storage, and distributed query execution, all of which are optimized for large scale OLAP queries. Comparing PostgreSQL's row store performance on both star-schema and non-star-schema models with these high performance albeit costly cloud systems would allow a more comprehensive evaluation of cost/performance trade offs, scalability, elastic provisioning, and resilience under concurrent workloads.

Finally, an important area for extension would be the development of a fully automated end-to-end ETL/ELT pipeline. Such a pipeline using this particular dataset would ingest data from operational sources, transform and load it into the dimensional warehouse, execute scheduled OLAP queries, and push aggregated outputs directly into visualization tools such as Tableau or Power BI. Implementing a production grade workflow using technologies such as Airflow, dbt, Kafka, or cloud-native orchestration services would more closely mirror real-world industrial analytical workflow systems. This automation pipeline would demonstrate how dimensional modeling interacts with continuous data ingestion, dashboard refresh cycles, and downstream BI systems, thereby providing a holistic view of warehouse operations in practice.

IX. Conclusion

This project implemented and evaluated a PostgreSQL based analytical data warehouse using a dimensional star schema and compared its performance against a non-star-schema relational model. By constructing a complete ELT workflow, designing all fact and dimension tables, and executing multiple OLAP query sets across macOS, Windows and Linux environments we observed that the star schema warehouse consistently improved analytical performance for the majority of tested workloads. These improvements were most evident in aggregation heavy and multi dimensional queries, where reduced redundancy and structured key based joins enabled more efficient execution than the flat transactional table of our dataset. Although certain hardware configurations produced unexpected behavior thus highlighting how caching, CPU architecture, and row-store characteristics can influence query execution. However, the overall results reinforce core data warehousing principles: dimensional modeling offers clearer organization, better analytical efficiency, and more scalable query behavior than operating directly on a denormalized transactional extract. The study demonstrates that even within the constraints of a traditional row-based relational engine like PostgreSQL, a well-designed star schema can meaningfully enhance OLAP processing and serve as a practical blueprint for organizations intending to efficiently perform analysis of their data.

X. References

[1] M. Stonebraker and G. Kemnitz, "The POSTGRES next generation database management system," Commun. ACM, vol. 34, no. 10, pp. 78–92, Oct. 1991, doi:10.1145/125223.125262.

[2] D. Joiner et al., "PostgreSQL Data Warehouse Implementation and Performance Optimization For Energy Companies," in 2024 IEEE International Systems Conferen -ce (SysCon), 2024, pp. 1–8. doi:10.1109/SysCon61195.2024.10553614.

[3] L. N. Nalla and V. M. Reddy, "Comparative Analysis of Modern Database Technologies in Ecommerce Applications," International Journal of Advanced Engineer -ing Technologies and Innovations, vol. 1, no. 2, pp. 21–39, 2020.

[4] J. Tang et al., "A Visualization Approach for Monitoring Order Processing in E-Commerce Warehouse," IEEE

Transactions on Visualization and Computer Graphics, vol. 28, no. 1, pp. 857–867, 2022, doi: 10.1109/TVCG.2021.3114878.
[5] S. Bose, “eCommerce Analytics 101 | What is eCommerce Analytics.” Accessed: Nov. 16, 2025. [Online]. Available: https://www.sarasanalytics.com/blog/ecommerce-analytics
[6] T. N. Hewage, M. N. Halgamuge, A. Syed, and G. Ekici, “Big data techniques of Google, Amazon, Facebook and Twitter.,” J. Commun., vol. 13, no. 2, pp. 94–100, 2018.
References
[7] “How Amazon's DynamoDB helped reinvent data-bases.” Accessed: Nov. 26, 2025. [Online]. Available: https://www.networkworld.com/article/939814/how-amazon-s-dynamodb-helped-reinvent-databases.html
[8] CBC, “Canadian Tire says customer info caught in data breach on e-commerce platform,” CBC, Oct. 2025, Accessed: Nov. 26, 2025. [Online]. Available: https://www.cbc.ca/news/business/canadian-tire-breach-customer-data-9.6937748
[9] BBC News, “'London whale' traders charged in US over 6.2bn loss,” BBC News, Aug. 2013, Accessed: Nov. 26, 2025. [Online]. Available: https://www.bbc.com/news/business-23692109

Appendix

A. *Sanbeer Shafin*

Pipeline Components Implemented:

I served as the project lead and helped implement all major technical components of the project. This included designing the full dimensional star-schema model (fact and all dimension tables), constructing the non-star-schema baseline database, and building the complete ELT workflow. From ingesting the raw CSV into a staging table to transforming and loading it into the dimensional warehouse, I helped with it all. I also coordinated meetings, assigned tasks, and oversaw integration of all group member contributions.

Validation and Evaluation:

I designed and executed the complete set of OLAP benchmark queries on my Mac OS Laptop used to compare both architectures(datawarehouse vs non-star database). I computed averaged runtimes, produced figures using Excel and performed the comparative analysis used in the Results and Discussion section.

Writing Contributions:

I authored the majority of the final report including the Abstract, Background and Theory, Methods & Pipeline, Results, Discussion, Limitations, Future Work and Conclusion.

Challenges and Lessons Learned:

The primary challenge was time limitations. Implementing multiple databases, conducting cross platform tests, and writing extensive technical sections required significant effort and time management. Due to time constraints, planned components such as a visualization module using Tableau and expanded ETL automation were not completed. This reinforces the importance of scoping and prioritization in data engineering projects.

Use of AI Tools:

ChatGPT was used for assistance in drafting OLAP query sets, refining sections of my initial written texts for the report and resolving minor PostgreSQL syntax issues. All conceptual design, schema development, experiments, and analysis were performed by me. AI tools were used for wording/grammar support and some coding support but not for generating project structure or approach.

B. *Jaiden Reese Sanchez*

Pipeline Evaluation and Validation

I conducted the testing of the queries made by the group using my Windows device. While testing, I ensured that all queries were returning the appropriate data that we requested. The data collected was the time it took to run all the queries then those times were calculated and shown on the paper.

Use of AI Tools:

No AI tools were used

Challenges and Lessons Learned:

The biggest lesson learned from this assignment was the management of several different projects and the time management needed to successfully achieve good results across all projects. The work of the project was hindered by the lack of time resources across the whole group but we were able to complete the assignment through careful planning and good communication throughout the group to get as much done as we could. By managing all of our priorities, our group was able to complete this assignment effectively.

C. *Jonathan Choi*

General Implementation Contribution:

Experimented with different data with a pipeline to generate synthetic data to add columns due to lack of data in said dataset. This effort was scrapped before proper implementation due to less complication with well known and trusted data.

Writing Contributions:

I’ve contributed the introduction part of the paper. Other attempted contributions were too lacking to make it into the final paper.

Challenges and Lessons Learned:

Unexpected situations and lack of focus due to juggling different projects from different courses and hopping between those subjects due to lack of scope and inefficient do-over resulted in little to no final contribution despite substantial use of time. To avoid these circumstances in the future, improvement in how one can look at the situation at hand would be necessary instead of being tunnel visioned into solving only the problem right on front.

Use of AI Tools:

QuillBot and Gemini were used to check grammar.

Pipeline Analysis and Testing:

Using the OLAP benchmark query set created by Shafin, wrote a simple python based automated testing suite using psycopg2 and performed a test on the Linux environment with two different hardwares.

D. Zuhaer Rahman

Pipeline Components Implemented:

My role did not involve direct implementation of the ELT pipeline or database schema. Instead, I contributed by reviewing the correctness and clarity of the technical workflow descriptions and ensuring that the documented pipeline accurately reflected the implemented system.

Validation and Evaluation:

While I did not run the OLAP benchmarks myself, I assisted in validating the reported results by reviewing the tables, figures, and written interpretations for accuracy, consistency and coherence within the Results and Discussion sections.

Writing Contributions:

My primary contribution was proofreading and editing the full project report to ensure clarity, consistency, and readability. I corrected grammatical issues, improved flow between sections, and ensured that the technical narrative remained coherent. I also developed the slide deck for the final presentation, summarizing the project's key components in a clear and accessible format.

Challenges and Lessons Learned:

The main challenge was integrating writing from multiple contributors into a unified, coherent document. Ensuring consistent terminology and logical flow required careful attention to detail. Converting dense technical content into an effective presentation also emphasized the importance of clear communication in data engineering projects.

Use of AI Tools:

I didn't use AI Tools.