

Relational Database Model Design for Logistics and Shipment Tracking Systems

Erza Hanif Pramudita Hanggara¹, Ezekiel Besarianta Prabaswara Sitorus², Aditya Wahyu Wiranatha³, Fathan Rafif Ryansyah⁴, Raditya Ramadhan Eka Widhianto⁵,
Diva Kurnianingtyas^{6*}

^{1,2,3,4,5,6*}Fakultas Ilmu Komputer, Universitas Brawijaya, Malang, 65154, Indonesia
erzahanif11@student.ub.ac.id¹; ezezielrian@student.ub.ac.id²; aditwiranata123@student.ub.ac.id³;
fathan78@student.ub.ac.id⁴; radityarew@student.ub.ac.id⁵; divaku@ub.ac.id^{6*}

Article Information

Article History:

Received : 13 May 2026
Revised : 3 October 2026
Accepted : 20 October 2026
Published : 30 October 2026

*Correspondences:

divaku@ub.ac.id

Keywords:

Relational Database, Shipment Tracking, Entity Relationship Diagram, SQL, GPS Ping

Copyright © 2026 by Author.
Published by Universitas Dinamika.



This is an open access article under the CC BY-SA license.

 10.37802/joti.v8i2.1414

Journal of Technology and Informatics (JoTI)

P-ISSN 2721-4842

E-ISSN 2686-6102

<https://ejournals.dinamika.ac.id/index.php/joti>

Abstract:

Efficient and reliable shipment tracking is essential for modern logistics, where managing large volumes of real-time transportation data remains a significant challenge. This study proposes a normalized relational database model to support structured logistics data management and shipment tracking using Entity Relationship Diagrams (ERDs) and Structured Query Language (SQL). The methodology involved analyzing a Transportation and Logistics dataset comprising 6,880 records and 32 attributes, identifying entity relationships, and applying normalization up to the Third Normal Form (3NF). The resulting database was implemented in Microsoft SQL Server and evaluated using simulated datasets containing 10,000, 100,000, and 1 million records. Experimental results demonstrated that complex queries could be executed in approximately 6 seconds for 1 million records. Furthermore, integrating a dedicated GPS_Ping table provides a structured mechanism for storing time-stamped shipment location observations, supporting location-based tracking data management. The proposed model reduces data redundancy, strengthens relational data integrity, and supports scalable query processing. These findings demonstrate the potential of a normalized relational database architecture as a foundation for developing scalable logistics information systems and reliable shipment tracking applications.

INTRODUCTION

Over the past few years, shipping and logistics systems have developed rapidly, becoming more rationalized and serving as one of the key competitive advantages for companies across various sectors, including manufacturing, trade, construction, agriculture, and services. Historically, logistics companies in the twentieth century still relied heavily on manual records and basic computer systems to manage transportation and inventory. As a result, information transactions were slower and more prone to errors, leading to operational inefficiencies. Entering the twenty-first century, the emergence of the smart logistics era has

encouraged further advancements in logistics intelligence. The implementation of advanced technologies such as artificial intelligence, the Internet of Things (IoT), big data, and blockchain has become a significant direction in logistics technology innovation [1]-[3],[10]-[12]. Intelligent decision-making and automation have become core characteristics of modern logistics systems.

Databases represent one of the most essential components in implementing information technology for logistics and shipment tracking systems. Databases allow structured and efficient information management, ranging from customer records, couriers, vehicles, and shipment histories. Through databases, interconnected data can be manipulated, searched, and displayed accurately [4]-[6]. The relational database model has become the most widely used approach because it effectively represents relationships among entities through interconnected table structures utilizing primary keys and foreign keys [8]-[9]. In shipment tracking systems, relational models enable the integration of shipment data, courier information, transportation routes, and delivery status. Well-designed database systems also allow shipment status reporting and delay monitoring to be performed automatically and in real-time [7], [13]-[15]. Furthermore, modern data analysis approaches enable the identification of more complex patterns through methods such as clustering and topic modeling, which can support the development of tracking systems and logistics process optimization [16]-[19]. Therefore, appropriate database design serves as an important foundation for building adaptive and modern logistics tracking systems.

Database design has also been examined in transportation and logistics-related studies. Steffan and Mumm (2011) [20] proposed a relational database model for vehicle routing problems, including the development of an E/R model and its relational implementation. Li and Wu (2021) [21] examined database design within a logistics information management system through conceptual, logical, and physical database design. These studies demonstrate the importance of database structure in supporting transportation and logistics information management. However, their database design is addressed in the context of vehicle routing and broader logistics information management, rather than specifically examining the relational database structure required for shipment tracking. Based on these previous studies, there is a need to examine a relational database structure specifically designed for logistics and shipment tracking data. This study focuses on organizing logistics entities and their relationships through relational database modeling, entity decomposition, normalization, SQL implementation, and performance evaluation.

METHOD

This study designed and implemented a relational database using a quantitative approach with descriptive-exploratory techniques. The research process was divided into several major stages as follows.

Literature Review and Data Analysis

The research began with a literature review to understand database design concepts and shipment tracking systems [4]-[6]. The dataset used in this study was obtained from the public Kaggle dataset entitled "Transportation and Logistics Tracking Dataset," published by nicolemachado [19]. The primary data used in this study were collected from March 2019 to December 2020. The dataset contains 6,880 records and 32 attributes related to transportation and logistics tracking were subsequently analyzed. Initial analysis was conducted using Microsoft Excel to identify patterns and relationships among key variables forming entities and

attributes, such as BookingID, vehicle_no, customerID, supplierID, from_location, to_location, and delivery_time.

The 32 attributes in the original dataset were analyzed and grouped according to the type of information represented by each attribute. Customer-related attributes, namely customerID and customerNameCode, were grouped into the Customer entity. Supplier-related attributes, supplierID and supplierNameCode, were grouped into Supplier. Vehicle-related attributes, vehicle_no and vehicleType, were grouped into Vehicle, while Driver_Name and Driver_MobileNo were grouped into Driver. Origin and destination information, including location names, coordinates, and location codes, was represented through the Location entity. Attributes describing an individual shipment transaction, including BookingID, booking dates, ETA information, transportation distance, material, and references to customer, supplier, vehicle, driver, and locations, were retained in Booking. GPS-related attributes, including GpsProvider, Data_Ping_time, Current_Location, Curr_lat, and Curr_lon, were represented in GPS_Ping. This grouping was used as the basis for constructing the relational model and its primary and foreign key relationships.

Database Design

This stage focused on database preparation through the following steps:

a. Conceptual Data Modeling

An Entity Relationship Diagram (ERD) was created to represent entities, attributes, and relationships among data elements [4]-[6],[9]. The ERD was designed using dbdiagram.io while following the standards proposed by Elmasri and Navathe (2016) regarding the selection of primary keys, foreign keys, and cardinality relationships.

b. Data Normalization

To ensure data integrity, reduce redundancy, and prevent anomalies according to database design principles [4],[6], normalization was applied up to the Third Normal Form (3NF).

1. First Normal Form (1NF): Eliminating redundant data.
2. Second Normal Form (2NF): Eliminating partial dependency, where non-key attributes depend only on part of the primary key.
3. Third Normal Form (3NF): Eliminating transitive dependency, where non-key attributes depend on other non-key attributes.

This approach aligns with recent research trends emphasizing the importance of efficient database integration to support large-scale data analysis [16], [18].

c. Physical Data Modeling

The ERD and normalization results were then transformed into a Physical Data Model (PDM).

Database Implementation and Testing

The database was implemented using Microsoft SQL Server 2022. Performance testing was conducted by measuring response time using the SET STATISTICS TIME ON command in SQL Server under varying data loads. The evaluation focused on CPU time and elapsed time for workloads containing 10,000, 100,000, and 1,000,000 records [12].

Result Analysis

The implementation and testing results were analyzed descriptively to evaluate the resulting database structure and query performance under different data volumes.

RESULTS AND DISCUSSION

Entities and Relationships

An entity is a real-world object that can be uniquely identified and distinguished from other objects stored in a database, as shown in Table 1. In database systems, entities represent concepts or objects to be managed and tracked, such as employees, products, or customers. The Entity Relationship Model shown in Table 2 is a data modeling technique used to describe relationships among entities within a database through key components such as entities, attributes, and relationships. Relationships among entities are categorized based on cardinality. The entities identified in this study include Booking, Vehicle, Location, Driver, Customer, and Supplier.

Table 1. Entity-Relationship Schema for the Logistics Booking System

Entity	Key Attribute	Other Attributes	Relationship
Booking	BookingID	BookingID_Date, Market, Data_Ping_time	One record for each shipment booking
Vehicle	vehicle_no	vehicleType	One vehicle can handle multiple bookings
Location	LocationID	Origin_Location, Destination_Location	Origin and destination locations exist in each booking
Driver	Driver_Name	Driver_MobileNo	One driver can handle multiple bookings
Customer	customerID	customerNameCode	One customer can create multiple bookings
Supplier	supplierID	supplierNameCode	One supplier can handle multiple bookings

Table 2. Entity Relationships, Foreign Keys, and Cardinality Rules

Entity	Relation	Cardinality
Customer	Customer.customerID → Booking.customerID	1 ; N
Supplier	Supplier.supplierID → Booking.supplierID	1 : N
Vehicle	Vehicle.vehicle_no → Booking.vehicle_no	1 : N
Driver	Driver.DriverID → Booking.DriverID	1 : N
Location (Origin)	Location.LocationID → Booking.OriginLocationID	1 : N
Location (Destination)	Location.LocationID → Booking.DestinationLocationID	1 : N
Booking	Booking.BookingID → GPS_Ping.BookingID	1 : N

Entity Relationship Diagram (ERD)

The Entity Relationship Diagram (ERD) in Figure 1 is a conceptual representation of database structures used to model relationships among entities in a system. The ERD visually illustrates key elements such as entities, attributes, and relationships, making it easier to understand system data requirements before implementation. The ERD design stage serves as the initial phase in database design. Its purpose is to comprehensively and consistently capture

information requirements based on system analysis. Once completed and validated, the ERD becomes the basis for designing the logical model that can later be implemented in a Relational Database Management System (RDBMS).

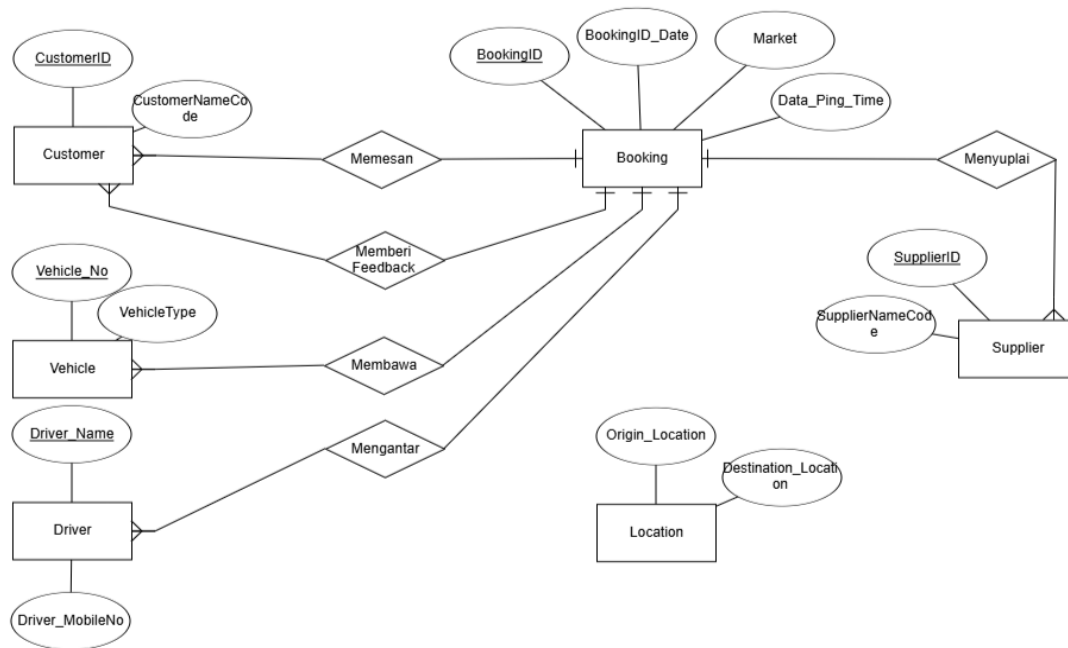


Figure 1. Entity Relationship Diagram (ERD)

Data Normalization

Data normalization was performed by separating data initially stored in a single Unnormalized Form (UNF) table containing multiple attributes, including:

DestinationLocation, actual_eta, Curr_lat, Curr_lon, ontime, delay, OriginLocation_Code, DestinationLocation_Code, trip_start_date, trip_end_date, TRANSPORTATION_DISTANCE_IN_KM, vehicleType, Minimum_kms_to_be_covered_in_a_day, Driver_Name, Driver_MobileNo, customerID, customerNameCode, supplierID, supplierNameCode, Material Shipped

The original structure combined multiple entities into a single table, making it vulnerable to data anomalies. Therefore, normalization was performed through the following stages:

1. First Normal Form (1NF)

Each column contains atomic values, and every row is unique without repeated data.

2. Second Normal Form (2NF)

Relations were decomposed based on functional dependencies from the primary key. Attributes not directly dependent on the primary key were moved into separate tables.

- a. Customer information was separated into the Customer table.
- b. Supplier information was separated into the Supplier table.
- c. Vehicle information was separated into the Vehicle table.
- d. Driver information was separated into the Driver table.
- e. Location information was separated into the Location table.

3. Third Normal Form (3NF)

All non-key attributes with transitive dependencies were eliminated, resulting in the final normalized tables: Customer, Supplier, Vehicle, Driver, Location, Booking, GPS_Ping

The normalization process was performed by decomposing the original RawData relation into several relational entities based on the functional dependencies of the attributes. The original dataset contains 6,880 records with 6,875 distinct BookingID values. Therefore, BookingID is not unique in the original raw relation and cannot be directly considered a candidate key of RawData. During transformation, duplicate BookingID records were handled using ROW_NUMBER() and only one booking-level record was retained for each BookingID in the Booking relation. Consequently, BookingID functions as the primary key of the resulting Booking relation. The functional dependencies used in the decomposition are shown in Table 3.

Table 3. Primary Keys and Functional Dependencies for Database Normalization

Relation	Key	Functional Dependency
Customer	customerID	customerID → customerNameCode
Supplier	supplierID	supplierID → supplierNameCode
Vehicle	vehicle_no	vehicle_no → vehicleType
Driver	DriverID	DriverID → Driver_Name, Driver_MobileNo
Location	LocationID	LocationID → LocationName, Latitude, Longitude, LocationCode
Booking	BookingID	BookingID → Market_Regular, BookingID_Date, trip_start_date, trip_end_date, Planned_ETA, actual_eta, ontime, delay, TRANSPORTATION_DISTANCE_IN_KM, Minimum_kms_to_be_covered_in_a_day, Material_Shipped, vehicle_no, DriverID, OriginLocationID, DestinationLocationID, customerID, supplierID
GPS_Ping	PingID	PingID → BookingID, Data_Ping_time, Current_Location, Curr_lat, Curr_lon, GpsProvider

The decomposition was performed according to these dependencies. Customer-related attributes were separated into the Customer relation because customerNameCode depends on customerID. Similarly, supplierNameCode depends on supplierID, and vehicleType depends on vehicle_no; therefore, these attributes were separated into the Supplier and Vehicle relations, respectively. Driver attributes were separated into the Driver relation and referenced from Booking through DriverID. Location attributes were separated into the Location relation and referenced by Booking through OriginLocationID and DestinationLocationID. GPS-related observations were separated into GPS_Ping because a single booking may be associated with multiple GPS observations.

Regarding Second Normal Form (2NF), the resulting relations use single-attribute keys (customerID, supplierID, vehicle_no, DriverID, LocationID, BookingID, and PingID). Because each key consists of only one attribute, partial dependency on a subset of a composite key cannot occur in these resulting relations. Thus, the resulting relations satisfy the 2NF requirement with respect to partial dependency.

For Third Normal Form (3NF), the decomposition removes dependencies in which non-key descriptive attributes would depend on another non-key attribute through the booking identifier. For example, if customer information were retained directly in Booking, the dependency could be represented as BookingID → customerID → customerNameCode. By separating customerID → customerNameCode into the Customer relation, the customer descriptive attribute is determined directly by its own key. The same principle is applied to Supplier, Vehicle, Driver, and Location. Consequently, the decomposition separates entity-specific attributes from the Booking relation and removes the identified transitive dependencies, resulting in relations that satisfy the 3NF structure.

Relational Model / Physical Data Model (PDM)

The relational model represents a logical database structure in which all data is stored in the form of tables (relations). Each table consists of rows (tuples) and columns (attributes). In this study, the relational model design served as the continuation of the conceptual design stage after normalization and adjustment for implementation in an RDBMS.

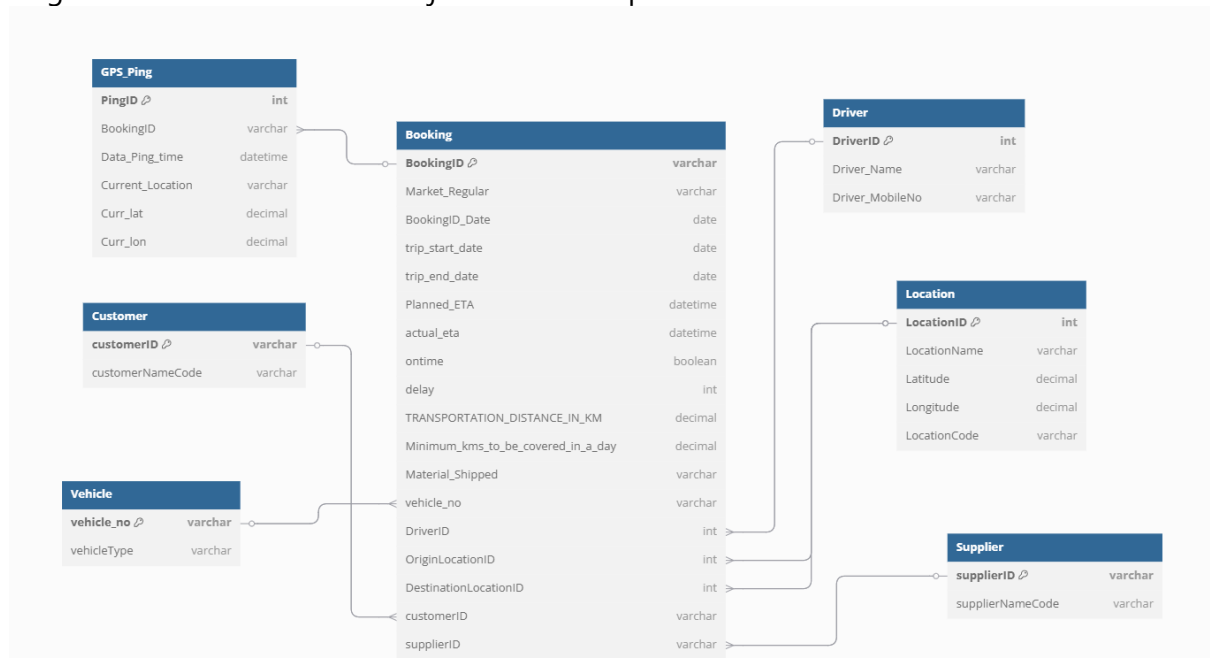


Figure 2. Physical Data Model (PDM)

The original dataset does not contain an explicit shipment-status attribute. Shipment progress is instead represented through several temporal, delivery, and location-related attributes, including BookingID_Date, trip_start_date, trip_end_date, Planned_ETA, actual_eta, ontime, and delay, as well as time-stamped GPS observations through Data_Ping_time, Current_Location, Curr_lat, and Curr_lon. Therefore, shipment lifecycle stages such as BOOKED, PICKED_UP, IN_TRANSIT, ARRIVED_AT_HUB, OUT_FOR_DELIVERY, and DELIVERED cannot be directly identified from the original dataset. For example, trip_start_date provides information about when a trip started, while trip_end_date indicates the recorded end of a trip, but these attributes do not explicitly define the complete shipment-status lifecycle.

Consequently, the proposed relational model retains the shipment progress information that is actually available in the dataset rather than introducing unsupported status values. The absence of an explicit shipment-status history is a limitation of the current model

and dataset, and a dedicated shipment-status or tracking-event relation could be incorporated in future work if event-level status data become available.

SQL Implementation

To manage and access data within the database system, Structured Query Language (SQL) was used. Before implementation, it was necessary to understand the structure of the database tables. Since the original dataset was still stored in a single large table, SQL commands were used to migrate the data into normalized tables. The database was implemented using seven relational tables: Customer, Supplier, Vehicle, Driver, Location, Booking, and GPS_Ping. The Booking table links shipment transactions to their associated entities through foreign keys, while GPS_Ping stores time-stamped location observations associated with bookings. The table definitions follow the relational structure presented in Figure 2. The following SQL statements illustrate the creation of the Booking table, which serves as the main shipment transaction table.

```
CREATE TABLE Booking (
    BookingID VARCHAR(50) PRIMARY KEY,
    Market_Regular VARCHAR(20) NULL,
    BookingID_Date DATE NULL,
    trip_start_date DATE NULL,
    trip_end_date DATE NULL,
    Planned_ETA DATETIME NULL,
    actual_eta DATETIME NULL,
    ontime BIT NULL,
    delay INT NULL,
    TRANSPORTATION_DISTANCE_IN_KM DECIMAL(10,2) NULL,
    Minimum_kms_to_be_covered_in_a_day DECIMAL(10,2) NULL,
    Material_Shipped VARCHAR(100) NULL,
    vehicle_no VARCHAR(50) NULL,
    DriverID INT NULL,
    OriginLocationID INT NULL,
    DestinationLocationID INT NULL,
    customerID VARCHAR(50) NULL,
    supplierID VARCHAR(50) NULL,
    FOREIGN KEY (vehicle_no) REFERENCES Vehicle(vehicle_no),
    FOREIGN KEY (DriverID) REFERENCES Driver(DriverID),
    FOREIGN KEY (OriginLocationID) REFERENCES Location(LocationID),
    FOREIGN KEY (DestinationLocationID) REFERENCES Location(LocationID),
    FOREIGN KEY (customerID) REFERENCES Customer(customerID),
    FOREIGN KEY (supplierID) REFERENCES Supplier(supplierID));
```

The Booking table uses BookingID as its primary key and foreign keys to reference the related customer, supplier, vehicle, driver, and location records. The GPS_Ping table uses PingID as its primary key and BookingID as a foreign key, allowing multiple location observations to be associated with a single booking. Following table creation, data migration was performed from the original RawData table into the seven normalized relations. Entity-specific records were extracted into the Customer, Supplier, Vehicle, Driver, and Location tables. Origin and destination coordinates were parsed into latitude and longitude values, while booking attributes were converted to the corresponding SQL data types. The resulting entity records were then mapped to Booking through their primary and foreign keys. To handle repeated BookingID values in the source dataset, ROW_NUMBER() was applied to retain one record per BookingID in the Booking relation. GPS-related observations were transferred to GPS_Ping to support the storage of multiple time-stamped location records for a booking. This process implemented the relational structure defined in the PDM.

System Performance Testing

Testing was conducted using dummy datasets containing 10 thousand, 100 thousand, and 1 million records shown in Figures 3, 4, and 5. Response times were measured using the SET STATISTICS TIME ON query in Microsoft SQL Server. Testing was performed on a laptop equipped with a dual-core processor, 4 threads, and 12 GB RAM. Because the evaluation was conducted in a single local hardware environment, the results are interpreted as workload-specific performance measurements rather than evidence of production-scale scalability.

Testing Procedure

1. Dummy data was inserted incrementally after each testing phase.

The dummy test data were generated using a SQL `WHILE` loop and `INSERT` statements to populate the `Booking` table. Each record was assigned a `BookingID` generated by concatenating the prefix 'BKG' with the loop counter.

2. The performance query was executed for each workload size, and the resulting CPU time and elapsed time were recorded.

```
SET STATISTICS TIME ON;

SELECT
    s.supplierNameCode,
    COUNT(b.BookingID) AS total_booking,
    AVG(CAST(b.delay AS FLOAT)) AS delay_rate
FROM Booking b
JOIN Supplier s ON b.supplierID = s.supplierID
JOIN Customer c ON b.customerID = c.customerID
JOIN Vehicle v ON b.vehicle_no = v.vehicle_no
JOIN Driver d ON b.DriverID = d.DriverID
JOIN Location o ON b.OriginLocationID = o.LocationID
JOIN Location dest ON b.DestinationLocationID = dest.LocationID
LEFT JOIN GPS_Ping g ON b.BookingID = g.BookingID
GROUP BY s.supplierNameCode
ORDER BY delay_rate DESC;

SET STATISTICS TIME OFF;
```

Testing Results

SQL Server Execution Times:
CPU time = 47 ms, elapsed time = 130 ms.

Figure 3. Performance Testing with 10,000 Records

SQL Server Execution Times:
CPU time = 469 ms, elapsed time = 504 ms.

Figure 4. Performance Testing with 100,000 Records

SQL Server Execution Times:
CPU time = 4641 ms, elapsed time = 6110 ms.

Figure 5. Performance Testing with 1,000,000 Records

Table 4. Query Performance Benchmarks across Varying Dataset Sizes

Number of Records	CPU Time (ms)	Elapsed Time (ms)
10,000	47	130
100,000	469	504

Number of Records	CPU Time (ms)	Elapsed Time (ms)
1,000,000	4.64	6.11

The performance measurements were obtained using Microsoft SQL Server 2022 in a single local hardware environment shown in Table 4. No additional indexes were explicitly configured. The test data were synthetically generated, with unique BookingID values and several attributes assigned fixed values. Each workload size (10,000, 100,000, and 1,000,000 records) was tested once. Therefore, the reported execution times represent individual measurements rather than averages across repeated trials. With a dataset containing 1 million records, complex queries could still be executed in approximately 6.11 seconds. This value represents the execution time of the tested SQL query under the experimental database environment. The 6.11-second execution time at the 1-million-record workload indicates that the proposed model remains executable under a substantially larger workload than the original dataset. However, this result represents database-level query execution and does not establish production-level interactive response time or multi-user scalability.

The findings of this study complement previous logistics tracking research employing various approaches. Dong (2023) [1] and Gunawan et al. (2023) [2] emphasized algorithms and IoT integration but did not focus extensively on relational database design. Li et al. (2024) [3] utilized blockchain for transparency and security, whereas this study focused on efficient relational database structures that remain compatible with such technologies. Furthermore, this study evaluated query execution time using synthetic workloads of up to 1 million records. The results provide an initial assessment of query execution under the tested database environment. This differs from the studies conducted by Haryanto [5] and Barlian & Susanti [13], which primarily focused on web-based applications without performance evaluation.

CONCLUSION AND SUGGESTIONS

This study successfully designed and implemented a relational database model for logistics and shipment tracking using an Entity Relationship Diagram (ERD) and normalization up to the Third Normal Form (3NF). The proposed model organizes logistics information into seven interconnected tables: Customer, Supplier, Vehicle, Driver, Location, Booking, and GPS_Ping, with primary and foreign keys ensuring structured relationships and data integrity. The normalization process reduces data redundancy and minimizes potential data anomalies by separating entity-specific attributes from shipment transactions. Furthermore, the GPS_Ping table provides a structured mechanism for storing multiple time-stamped location observations associated with individual bookings. Performance evaluation using synthetic datasets of 10,000, 100,000, and 1 million records demonstrated that the implemented database could execute the tested complex SQL query in approximately 6.11 seconds for the largest workload. These findings establish the proposed relational model as a structured foundation for logistics information management and further shipment tracking system development. Nevertheless, real-time GPS updates, complete shipment-status history, repeated performance trials, and concurrent user requests require further investigation. Future work should focus on integrating real-time data streaming through MQTT or Apache Kafka, implementing shipment-status event management, and evaluating cloud-based deployment to improve scalability, reliability, and operational efficiency.

REFERENCES

- [1] D. G. Dong, "Design and development of intelligent logistics tracking system based on computer algorithm," *Int. J. Appl. Inf. Manag.*, vol. 3, no. 2, pp. 58–69, 2023, doi:

- 10.47738/ijaim.v3i2.56.
- [2] W. Gunawan, M. Listya, H. Alianto, and D. A. Nugroho, "Real time tracking system integrated with IoT for package delivery in Indonesia's logistic company with TOGAF implementation," in Proc. IEEE ICISS, 2023, doi: 10.1109/ICISS59129.2023.10291211.
- [3] Z. Z. Li, J. Zhao, H. Cui, E. X. Zheng, X. Ma, D. Y. W. Liu, and M. H. Au, "Blockchain for smart logistics: Enhancing identity security, bidding transparency and goods tracking," in Proc. 6th ACM Int. Symp. Blockchain and Secure Critical Infrastructure (BSCI), 2024, pp. 1–9, doi: 10.1145/3659463.3660028.
- [4] D. M. Kroenke, D. J. Auer, R. C. Yoder, and S. L. Vandenberg, Database Processing: Fundamentals, Design, and Implementation, 16th ed. Pearson Education, 2024. [Online]. Available: <https://studentebookhub.com/wp-content/uploads/2024/preview/9780136930174.pdf>
- [5] J. T. Haryanto, "Sistem logistik berbasis web untuk usaha mikro, kecil, dan menengah," J. Edukasi Teknol. Inf. (EDUTIK), 2022. [Online]. Available: https://www.researchgate.net/publication/373377084_Sistem_Logistik_Berbasis_Web_untuk_Usaha_Mikro_Kecil_dan_Menengah
- [6] R. Elmasri and S. B. Navathe, Fundamentals of Database Systems, 7th ed. Pearson Education, 2016. [Online]. Available: <https://www.auhd.edu.ye/upfiles/elibrary/Azal2020-01-22-12-28-11-76901.pdf>
- [7] D. Effendi, "Perancangan sistem pelacakan pengiriman barang (tracking system for freight forwarding) berbasis SMS (short message service)," Wawasan Tridharma, vol. 11, pp. 1–10, 2017. [Online]. Available: https://www.researchgate.net/publication/320099494_Perancangan_Sistem_Pelacakan_Pengiriman_Barang_Tracking_System_For_Freight_Forwading_Berbasis_SMS_Short_Message_Service
- [8] A. Zulfikar and T. Hidayat, "Analisis dan perancangan database menggunakan model konseptual data warehouse pada toko online," J. EDUKATIF: J. Ilmu Pendidikan, vol. 3, no. 6, pp. 4622–4630, 2021. [Online]. Available: <https://edukatif.org/edukatif/article/download/1477/pdf>
- [9] Y. Y. Joeffie and P. P. Kalatiku, "Desain basis data sistem informasi akademik di Fakultas Teknik Universitas Tadulako," J. Ilm. Foristek, vol. 2, pp. 190–194, 2013. [Online]. Available: <https://foristek.fatek.untad.ac.id/index.php/foristek/article/view/101/76>
- [10] Y. Jing and P. Yang, "Research on the risk prevention of cross-border e-commerce logistics in China by applying big data technology," Procedia Comput. Sci., vol. 208, pp. 407–413, 2022. [Online]. Available: Research on the Risk Prevention of Cross-Border E-Commerce Logistics in China by Applying Big Data Technology - ScienceDirect
- [11] R. A. Mohammed, A. Nadi, L. Tavasszy, and M. de Bok, "Data fusion approach to identify distribution chain segments in freight shipment databases," Transp. Res. Rec., vol. 2677, no. 6, pp. 310–323, 2023. [Online]. Available: Data Fusion Approach to Identify Distribution Chain Segments in Freight Shipment Databases - Raed Ali Mohammed, Ali Nadi, Lórant Tavasszy, Michiel de Bok, 2023
- [12] B. R. Prakash, K. V. Nagaraja, and H. P. M. Kumar, "Streaming analytics for shipment monitoring system," in Proc. Int. Conf. Futuristic Technol. (INCOFT), Belgaum, India, 2022, pp. 1–4. [Online]. Available: Streaming Analytics for Shipment Monitoring System | IEEE Conference Publication | IEEE Xplore
- [13] G. Barlian and S. Susanti, "Sistem informasi tracking logistik berbasis website pada distribution center PT Akur Pratama (YOGYA GROUP)," INTEK: J. Inform. Teknol. Inf., vol.

- 8, no. 1, pp. 13–20, 2024. [Online]. Available: <https://jurnal.umpwr.ac.id/index.php/intek/article/view/1605>
- [14] M. F. A. Prasetyo, R. Widodo, and R. Maulana, "Rancang bangun tracking pengiriman berbasis website menggunakan metode systems development life cycle (SDLC) dengan model waterfall," J. Teknol. Inf. Komun. (JTik), vol. 8, no. 2, pp. 115–123, 2024. [Online]. Available: <https://journal.lembagakita.org/jtik/article/view/1387>
- [15] R. J. Pratama, "Perancangan sistem informasi laporan pengiriman barang berbasis web di inventory PT. Setiap Hari Dipakai," J. Indones. Sos. Teknol., vol. 2, no. 11, pp. 1719–1728, 2021. [Online]. Available: <https://jst.publikasiindonesia.id/index.php/jst/article/view/281>
- [16] L. Rosiana and I. Yuadi, "K-Means Clustering untuk Analisis Tren Peminjaman Buku di Perpustakaan," J. Technol. Inform. (JoTI), vol. 7, no. 1, pp. 1–10, Apr. 2025, doi: 10.37802/joti.v7i1.933.
- [17] J. Anggara et al., "Pengembangan Sistem Prediksi Harga dan Rekomendasi Mobil Bekas Berbasis Machine Learning," J. Technol. Inform. (JoTI), vol. 7, no. 1, pp. 11–23, Apr. 2025, doi: 10.37802/joti.v7i1.987.
- [18] P. Pavithra and S. Savitha, "Topic Modeling for Evolving Textual Data Using LDA, HDP, NMF, BERTopic and DTM With a Focus on Research Papers," J. Technol. Inform. (JoTI), vol. 5, no. 2, pp. 53–63, Apr. 2024, doi: 10.37802/joti.v5i2.618.
- [19] N. Machado, "Transportation and Logistics Tracking Dataset," Kaggle. [Online]. Available: <https://www.kaggle.com/datasets/nicolemachado/transportation-and-logistics-tracking-dataset>
- [20] R. Steffan and H. Mumm, "Database design for vehicle routing problems," in Perspectives in Business Informatics Research, Local Proceedings, Riga, Latvia, pp. 12–20, 2011. [Online]. Available: https://www.researchgate.net/publication/331398807_Database_Design_for_Vehicle_Routing_Problems
- [21] Q. Li and G. Wu, "ERP system in the logistics information management system of supply chain enterprises," Mobile Information Systems, vol. 2021, Art. no. 7423717, pp. 1–11, 2021, doi: 10.1155/2021/7423717. [Online]. Available: <https://onlinelibrary.wiley.com/doi/10.1155/2021/7423717>