**TT HOLDING DATABASE DESIGN**

# TABLE OF CONTENTS

List of Figures ………………………………………………………………………………………………………2

List of Tables ……………………………………………………………………………………………………….3

List of Abbreviations ……………………………………………………………………………………………….3

Abstract ……………………………………………………………………………………………………………..4

Chapter 1: Introduction ……………………………………………………………………………………………5

Problem Statement Problem Solving Objective Scope & Constraint

Chapter 2: Literature Review …………………………………………………………………………………….6

Introduction

Reviewing the literature (Last papers and publications)

Findings & Discussion

Chapter 3: Methodology ………………………………………………………………………………………….7

Requirement Analysis System Design

Architecture design

System Implementation

Testing

Chapter 4: System Initiation and Planning ………………………………………………………………………14

Assessing Project feasibility Project Plan

Chapter 5: System Analysis ………………………………………………………………………………………15

Determining System Requirement Structuring System Requirements

Chapter 6: Conclusion ……………………………………………………………………………………………16

Advantages of the System Future Enhancement of the System Potential Benefits Conclusion

References …………………………………………………………………………………………………………17

Appendices …………………………………………………………………………………………………………18

# **LIST OF FIGURES**

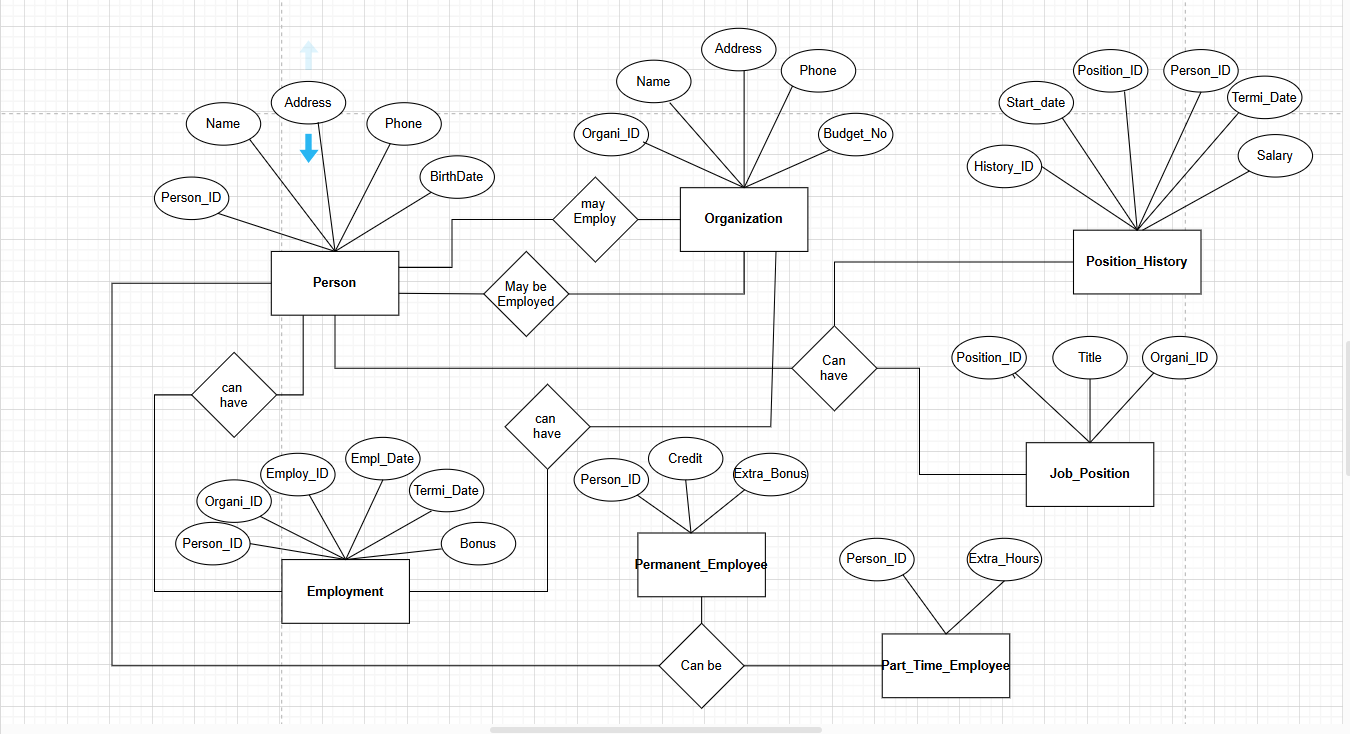
****

Figure 1 Shows TT\_Holding design using Chen

# **LIST OF TABLES**

1. **Person Table:** (Person\_ID **PK**, Name, Address, Phone, Birth\_Date)

2. **Organization Table:** (Organization\_ID **PK**, Name, Address, Phone, Budget\_Number)

3. **Job\_Position Table**:(Position\_ID **PK**, Title, Organization\_ID **FK**)

4. **Employment Table:** (Employment\_ID **PK**, Person\_ID **FK**, Organization\_ID **FK**, Employment\_Date, Termination\_Date, Bonus)

5. **Permanent\_Employee Table:** (Person\_ID **PK, FK**, Extra\_Bonus, Credit)

6. **Part\_Time\_Employee Table:** (Person\_ID **PK, FK**, Extra\_Hours)

7.**Position\_History Table:** (History\_ID **PK**, Person\_ID **FK**, Position\_ID **FK**, Start\_Date, Termination\_Date, Salary)

**List of Abbreviations**

**DBMS (Database Management System)**: An DBMS helps end users in creating, maintaining, reading, modifying and deleting data in a database. It also controls concurrency, data integrity and security in databases. (Yasar, 2010)

**SQL (Structured Query Language)**: A language used for managing databases. Also (Amazon, 2023) says that is a programming language for storing and processing information in a relational database

**UML (Unified Modeling Language)**: is a general-purpose visual modeling language that is intended to provide a standard way to visualize the design of a system, a diagramming language used for system design. (wikipedia, 2022)

**PK (Primary Key)**: A unique identifier for records in a table.

**FK (Foreign Key)**: A key that links records from one table to another.

**ER (Entity-Relationship)**: A diagram showing how things and the manner in which they relate to one another within a database. Specifies entities and things that go together. An ER model is a minimal basic ER model that can hold distinct relationships between entities. (Wikipedia, 2023)

**Abstract**

This report demonstrates the design and development of a database system for TT Holdings focusing on organizational and employee management. The project uses Entity-Relationship (ER) diagrams and UML diagrams in illustrating the structure of the database and relationship. The project involves the development of basic database tables and SQL commands for table definitions, insertion of sample data, and advanced database functions like views, triggers, functions, and stored procedures.

The system can efficiently manage employee records, employment history, job titles, and organizational information, with data integrity, consistency, and scalability. The architectural design also dictates how components interact with each other in the database system. The database is structured for efficient data retrieval, which enhances the performance of the entire system. The project provides a full database solution, where TT Holdings can automate employee management and organizational tracking, while enjoying a secure and well-structured data environment

**Chapter 1: Introduction**

## **Problem Statement**

Managing the employee records, job titles, and organizational details manually is time-consuming, laborious, and error-prone. TT Holdings requires a database management system to look after employee data, organizations, work history, and job titles effectively to store, retrieve, and update the information. There exists data redundancy, trace problems for employment records, and payroll calculation inefficiency by adhering to the current practice. It must be an in-house database system that not only supports data integrity and security but also provides quick access to critical information.

# **Problem Solving**

As a move towards solving these problems, this project establishes a relational database system that systematically organizes and manages organizational and employee information efficiently. The system uses:

* Entity-Relationship (ER) diagrams.
* Structured tables to store the employee, organization, work history, and job posts.
* SQL queries to insert, retrieve, and modify data.
* Triggers, stored procedures, and views in a bid to streamline managing data and improve efficiency.

# **Objective**

The main aim of this project is to create and implement a well-designed database system for TT Holdings that guarantees:

* Efficient storage and retrieval of employee and organizational data.
* Accurate record keeping of job positions and employment history.
* Reduced data redundancy and enhanced data integrity.
* Automatic management of employment records through views, triggers, and stored procedures.

# **Scope & Constraint**

The system will be employment history, job titles, and organization management based, with the exception of payroll and HR-specific activity beyond employee tracking.

**Chapter 2: Literature Review**

# **Introduction**

Several research studies have analyzed the importance of relational databases, normalization techniques, and advanced SQL features to maintain data integrity and reduce redundancy. (Connolly, 2021) said that Databases play a critical role in modern organizations, facilitating data storage, retrieval, and manipulation. Research indicates that relational database management systems (RDBMS) such as MySQL, PostgreSQL, and SQL Server are widely used due to their structured approach to data management. A well-designed database gives you access to current, correct information. Since a good design is the secret to success in using a database, taking the time needed to learn about good design is worth it. Ultimately, you are much more likely to have a database that will serve your needs and be easily adaptable to change. (Support, 2023).This chapter presents past research on relational database design, employee tracking systems, and automation using triggers, functions, and stored procedures. The findings from these studies provide insights on best practices for designing an effective database system for TT Holdings

### **Relational Database Design**

A well-structured database begins with careful planning and normalization to eliminate data redundancy and improve data integrity. According to (Elmasri, 2020) Normalization is the process of organizing data into multiple related tables to minimize redundancy while preserving data consistency. The authors further discuss the role of entity-relationship (ER) modeling in defining the structure of a database before implementation. Studies emphasize the need for **properly structured database tables** to maintain **data consistency and avoid anomalies**. A research paper by (Ramez Elmasri, August 26, 2016)discusses **Entity-Relationship (ER) modeling** and how **normalization** minimizes data duplication. This approach ensures that the **TT Holdings database** remains **efficient and scalable.**

# **Findings & Discussion**

As evident from the literature reviewed, database design, data manipulation and testing are essential for constructing stable information systems. The use of normalization and indexing significantly improves performance while ACID properties ensure data consistency. Query optimization techniques such as indexing and plan execution analysis improve data retrieval performance. Furthermore, good database testing practices such as stress and regression testing ensure that the database is stable before large-scale implementation.

Researchers highlight that data insertion follows the principles of ensuring constraints like primary keys and foreign keys to maintain data integrity (Silberschatz.A, 2020) Inserting records efficiently involves using batch insert statements and indexing strategies to enhance performance.

Efficient data retrieval requires optimized queries to minimize execution time and resource usage. According to a study by (García-Molina.H, 2021) query optimization techniques include indexing, query rewriting, and execution plan analysis. These techniques significantly improve the speed of SELECT statements, especially when dealing with large datasets.

**Chapter 3: Methodology**

# **Requirement Analysis**

The process of developing the TT Holdings database system started with the analysis of detailed requirements to possess a system aligned with business requirements. The key requirements are the following:

**Functional Requirements**:

* Employee Data Management – The system will hold employee data, employment history, and working positions.
* Organization Management – There should be an association to an organization for a single employee to maintain employment details.
* Temporary Features – Triggers and stored functions to be utilized by the system in carrying out automatic updates and computation.

# **System Implementation / Prototyping**

The database was implemented in MySQL with table structures, constraints, views, and procedures. The primary steps followed in the implementation were:

***Creating database tables with proper relationship and constraints.***

Only three screenshots have been included in this section to illustrate the creation of table process. However, all relevant screenshots can be found in the appendices for comprehensive reference.

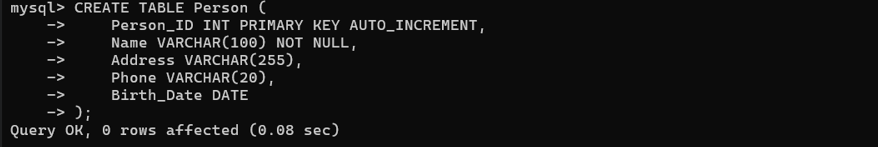


Figure 2 Shows creation of table Person

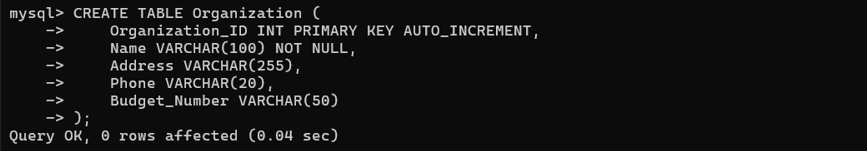


Figure 3 Shows creation of table Organization

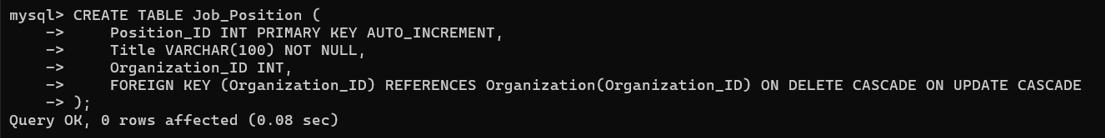


Figure 4 Shows creation of table Job\_Position

***Designing SQL statements for inserting, updating, and fetching data.***

***statements for inserting***

Below, only two tables are shown to illustrate the data insertion process, while the insertion for the remaining tables is provided in the Appendices

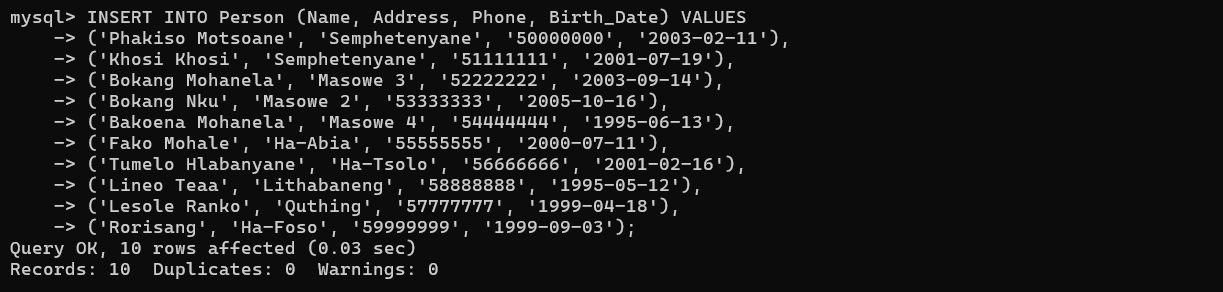


Figure 5 Shows Person table where i am inserting the values to it.

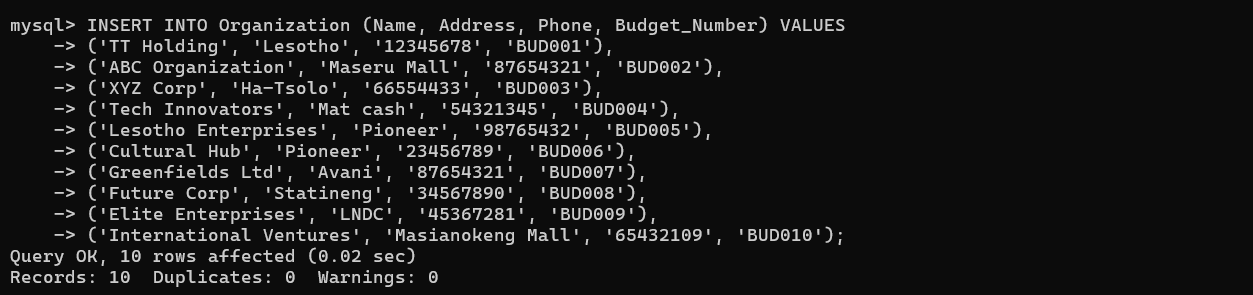


Figure 6 Shows Organization table where i am inserting the values.

***statements for updating***

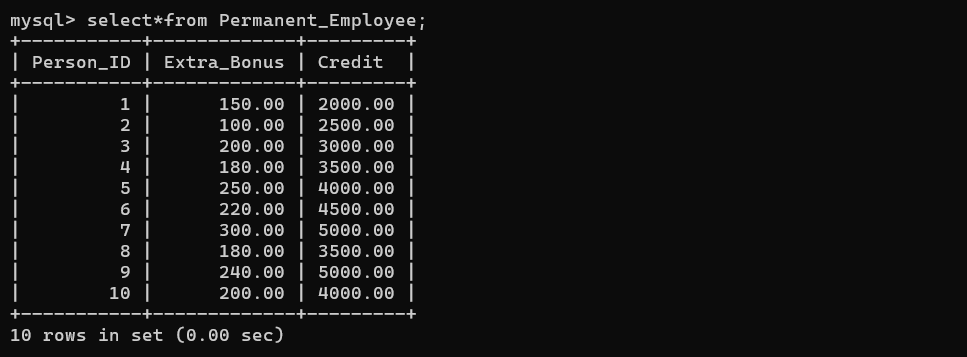


Figure 7 Shows Permanent Employee table before Updating it

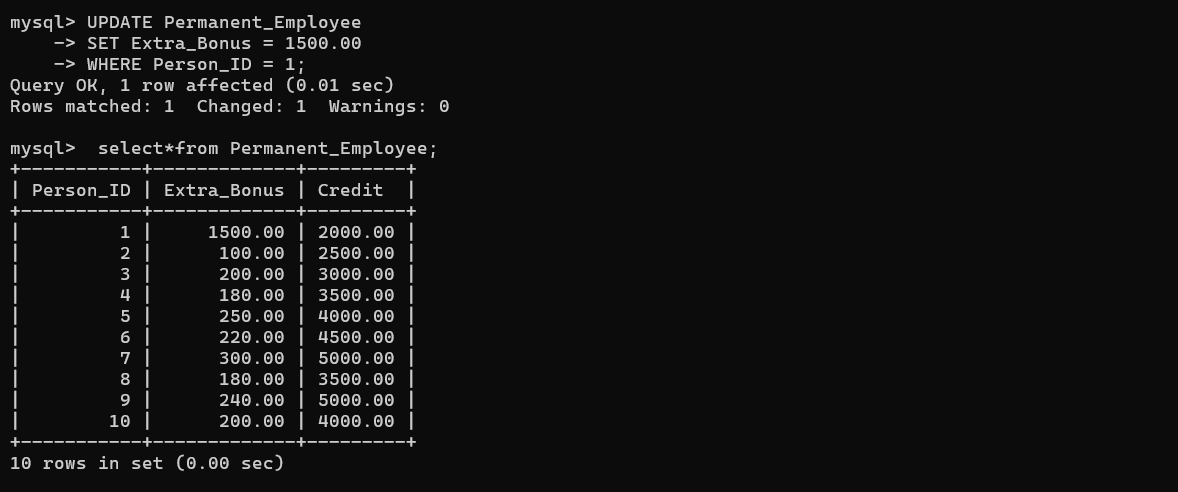


Figure 8 Shows Permanent Employee table after Updating Person with ID 1 to new extra bonus of M 1500.00

***statements for fetching data***

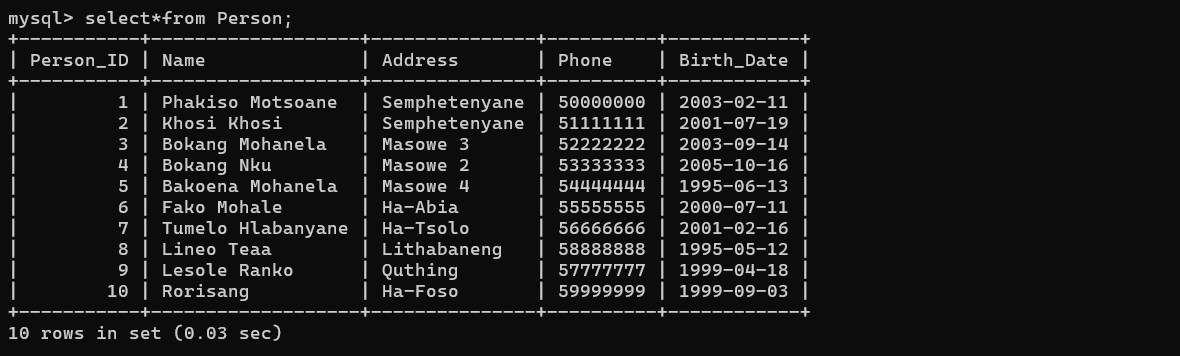


Figure 9 Shows Person Table after selecting or fetching all data from it

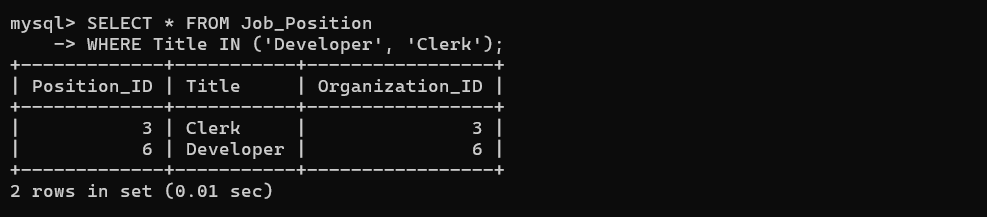


Figure 10 Show table Job\_Position only fetching data where Title is **Clerk** and **Developer** not selecting all from that table

1. Adding triggers, stored procedures, and functions for automated tasks.

***Triggers***

For brevity, only three screenshots have been included in this section to illustrate the process. However, all relevant screenshots can be found in the appendices for comprehensive reference.

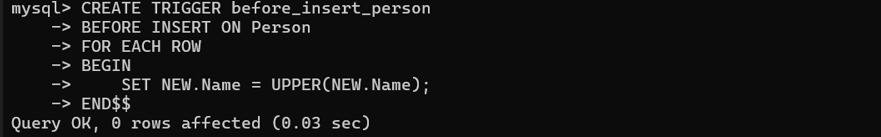


Figure 11 show the INSERT TRIGGER which automatically sets every new Name to Upper case every time a new name is inserted

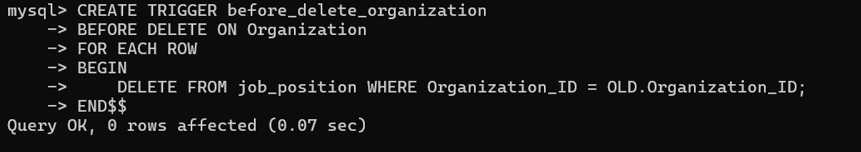


Figure 12 Ensures that when an organization is deleted, related job positions are removed.

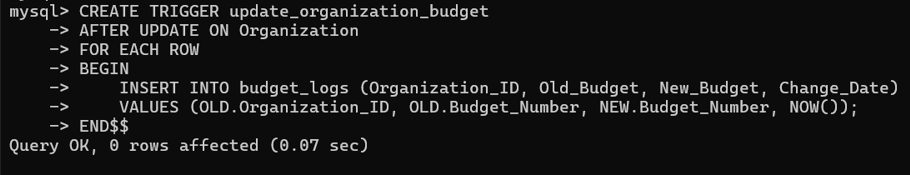


Figure 13 When an organization updates its budget, a log is recorded.

**Testing**

Testing was carried out to make data accurate, secure, and the system reliable. The testing done is given below:

***Unit Testing***

Verified independent database objects (tables, views, triggers).

**Testing Tables (Checking Schema & Integrity)**

To verify that the tables were created correctly, three screenshots have been included below. These demonstrate the schema structure, column data types, and constraints applied to each table.

The remaining table verification screenshots have been included in the Appendices for reference.

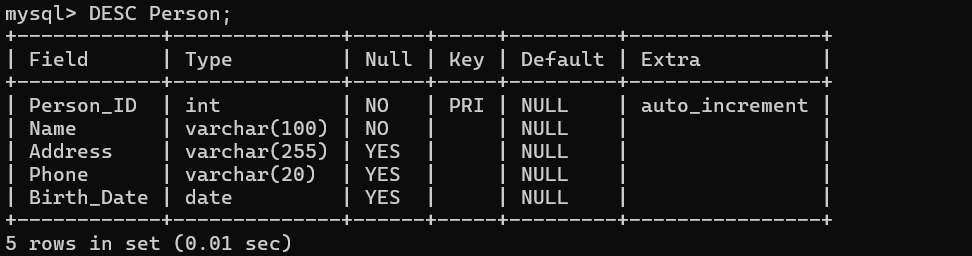


Figure 14 Shows person Table when i describe it.

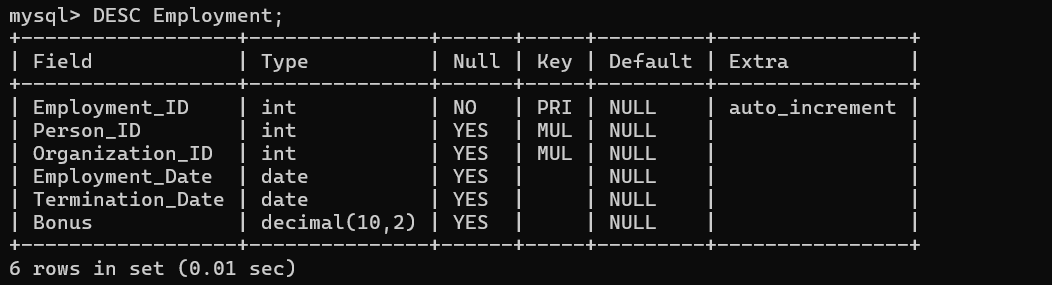


Figure 15 Shows Employment table when i describe it.

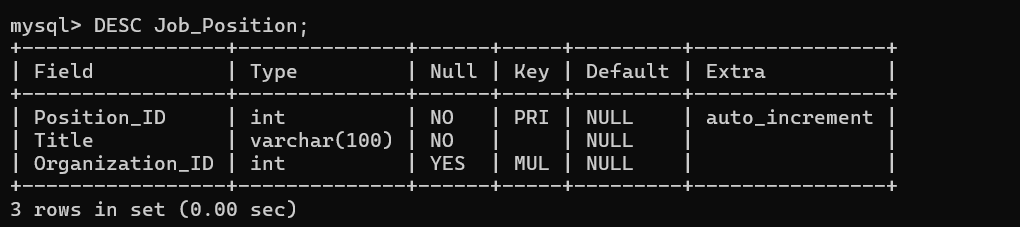


Figure 16 Shows Job\_Position table when i describe it.

**Testing Views (Checking Data Displayed by Views)**

To ensure that views return the expected data,three screenshots are included below. These screenshots show the execution of SELECT statements on views and verify that the correct data is retrieved.

The remaining view verification screenshots can be found in the Appendices.



Figure 17 Shows the creation of table person view

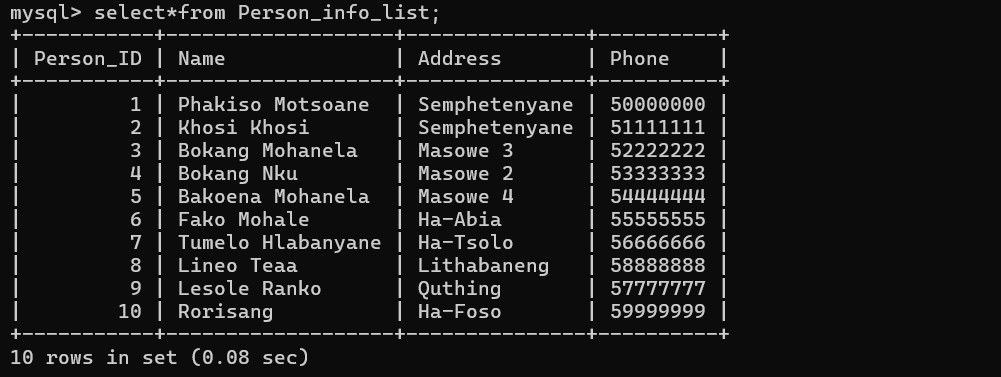


Figure 18 Show the results of the view when retrieving data

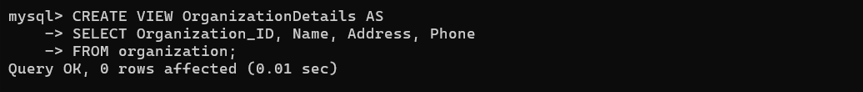


Figure 19 Shows the creation of table Organization view

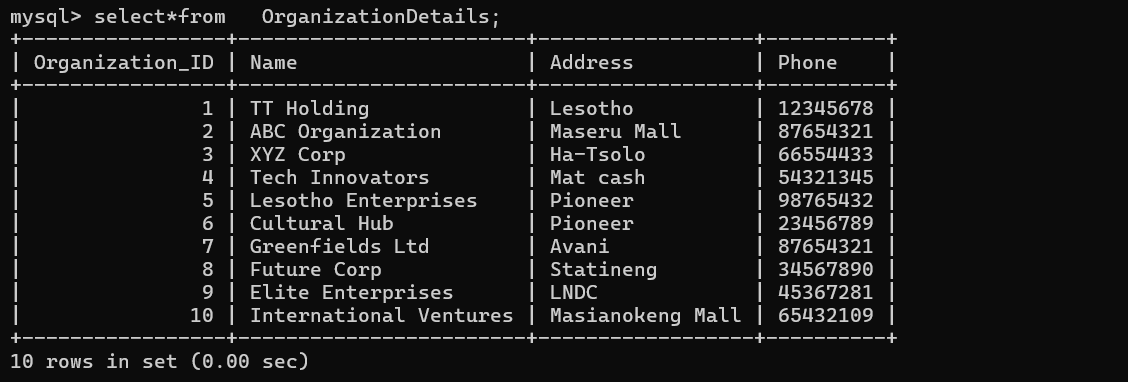


Figure 20 Show the results of the view when retrieving data

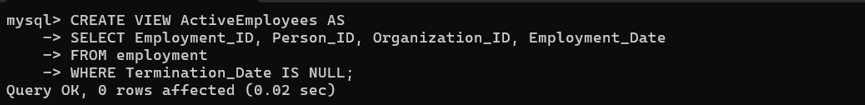


Figure 21 Shows the creation of table Employement view

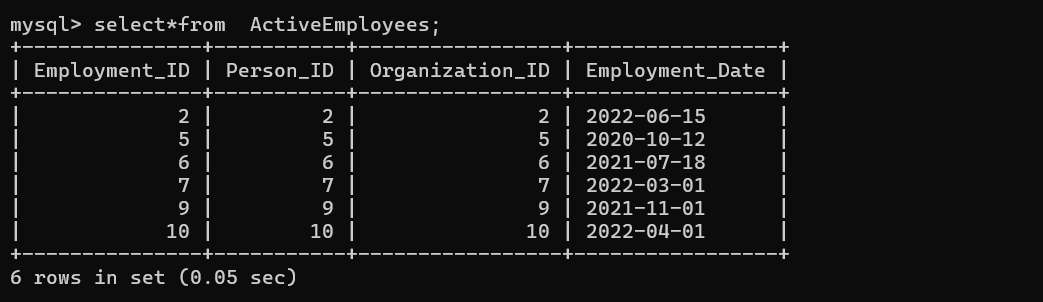


Figure 22 Show the results of the view when retrieving data

**Testing Triggers (Checking Automatic Execution)**

Triggers ensure that **automatic changes** occur when specific actions are performed in the database. To test the triggers, **one screenshot is** provided

The remaining triggers verification screenshots have been included in the **Appendices** for reference

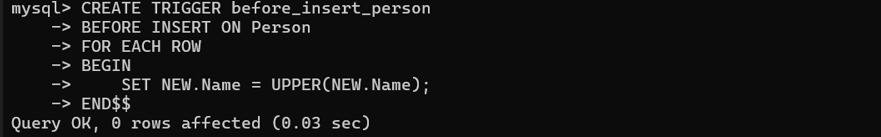
****

Figure 23 Shows creation of s trigger that set name to uppercase when i insert data

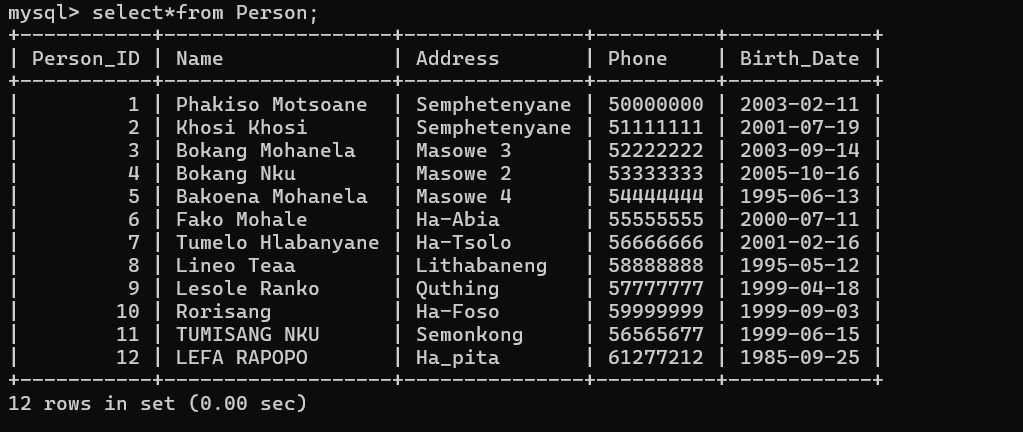


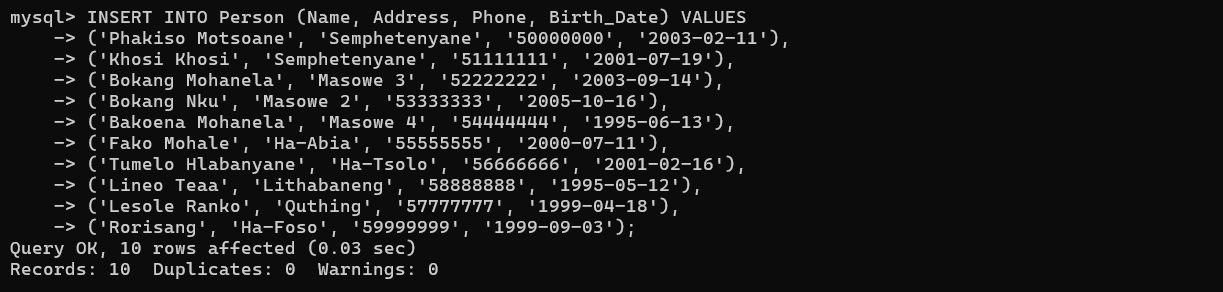
Figure Show the results of the triiger when i select all from Person table

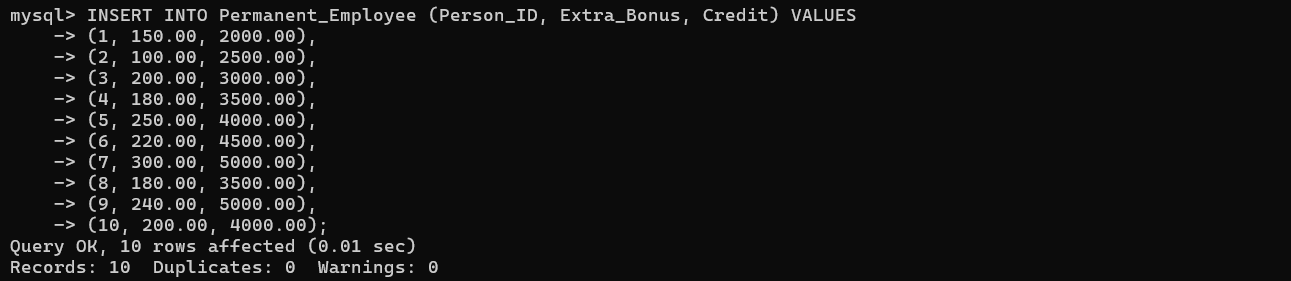
***Integration Testing***

Foreign key constraint testing and table integration are the process of checking whether table-to-table relationships are functional and data remains intact. Foreign key constraints are applied to attain referential integrity between two tables in the sense that foreign key column(s) values of a table should be present within referenced primary key column(s) of another table.

Since we have Person\_ID as Foreign key Permanent table that reference Person this show both tables integrate.

Below its just a sample of the two tables.





**Chapter 4: System Planning and Initiation**

# **Project Feasibility Assessment**

The feasibility of TT Holdings' database system is assessed on the following parameters:

**Technical Feasibility**

The system will be implemented using MySQL for storing data with high performance and reliability. The system architecture will be sustainable and scalable.

**Operational Feasibility**

The system supports TT Holdings' business objectives through improved data administration, monitoring of employees, and automation of HR functions.

# **Project Plan**

Phase 1: System Analysis

* Determine system requirements
* Define major functionalities
* Collect input from stakeholders

Phase 2: System Design

* Define database schema
* Create architectural design

Phase 3: System Implementation

* Create MySQL database and tables
* Create stored procedures, triggers, and views

**Chapter 5: System Analysis**

## **System Requirements Determination**

To develop an effective and reliable database system for TT Holdings, the following requirements have been determined:

**Functional Requirements**

* Store and retain employee information
* Maintain employment history
* Automatically delete related records when an employee is deleted

**Non-Functional Requirements**

* Data consistency and integrity
* Secure access and login controls

## **System Requirements Structuring**

**User Permissions & Roles**

* Admin: Creates employee records, job positions, and system configuration.

**Database Schema Overview**

* Person Table: Saves employee details
* Employment Table: Saves employment history
* Job\_Position Table: Saves job positions

Such system requirements structured above ensure the TT Holdings database system is most effectively dealing with all the operational and business needs

**Chapter 6: Conclusion**

# **Advantages of the System**

Advantages of the TT Holdings database management system include:

* Ease of Handling Data: The system enables easy storage, retrieval, and upkeep of employee and organizational data.
* Accuracy: Automating data entry and retrieval eliminates errors and maintains data integrity.
* Enhanced Security: Access controls and authentication measures put in place protect sensitive data.
* Expandability: The system is capable of accommodating the growing needs of TT Holdings as it grows.

# **Future System Improvements**

Certain of the system improvements are possible:

* Cloud Deployment: Moving the database to the cloud to improve accessibility and scalability.
* Improved User Interface: Developing an improved interactive and user-friendly front-end for improved user experience.
* Mobile Application Support: Having a mobile-optimized version for convenience on the go.

# **Conclusion**

The TT Holdings database management system is robust and effective to handle organizational and employee data. Robust management of structured data, security, and usage are its cornerstones that enable it to do what the organization requires. Future enhancements proposed will enhance its ability even more so that it will develop at a rapid rate with evolving business situations. Implementation of the system in TT Holdings provides it with better data management, operational effectiveness, and long-term viability.

# **References**

Amazon, 2023. *Amazon Web Service (AWS).* [Online]   
Available at: https://aws.amazon.com  
[Accessed 08 March 2025].

Connolly, T. &. B. C., 2021. *Database Systems: A Practical Approach to Design, Implementation, and Management.* 7th edn ed. Harlow: Pearson Education.

Elmasri, R. N. S., 2020. *Fundamentals of Database Systems.* 7th edn ed. Harlow: Pearson Education.

García-Molina.H, U. J. &. W., 2021. *Database Systems: The Complete Book.* 2nd edn ed. Upper Saddle River: Pearson.

Ramez Elmasri, S. N., August 26, 2016. *Fundamentals of Database Systems.* 7th ed. ed. New York: Pearson.

Silberschatz.A, K. &. S., 2020. *Database System Concepts.* 7th edn ed. New York: McGraw-Hill.

Support, M., 2023. *Microsof.* [Online]   
Available at: https://support.microsoft.com  
[Accessed 25 March 2025].

wikipedia, 2022. *wikipedia.* [Online]   
Available at: https://en.wikipedia.org  
[Accessed 08 March 2025].

Wikipedia, 2023. *Wikipedia.* [Online]   
Available at: https://en.wikipedia.org  
[Accessed 08 March 2025].

Yasar, K., 2010. *Tech Target.* [Online]   
Available at: https://www.techtarget.com  
[Accessed 08 March 2025].

**Appendices**

**Appendix A:** **SQL Database Setup & Schema 1 Entity-Relationship Diagram (ERD)**

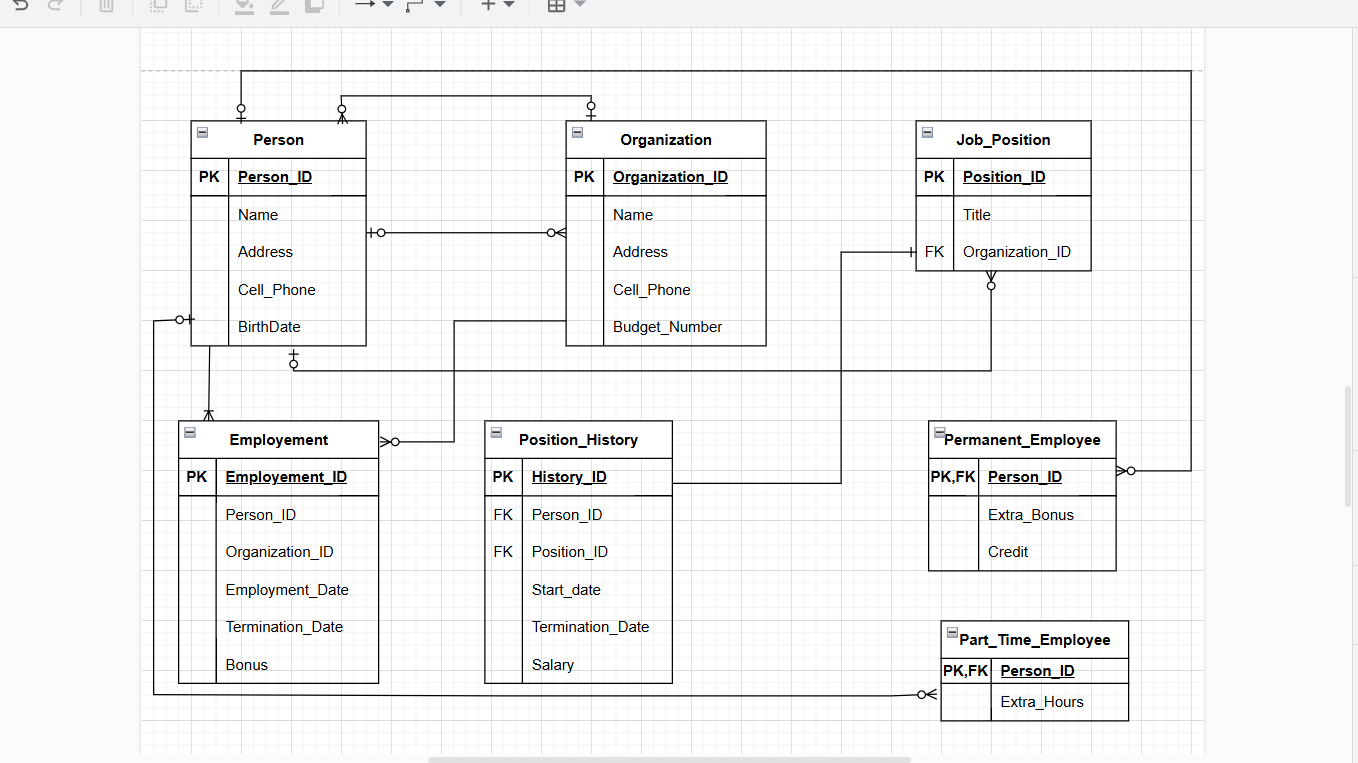
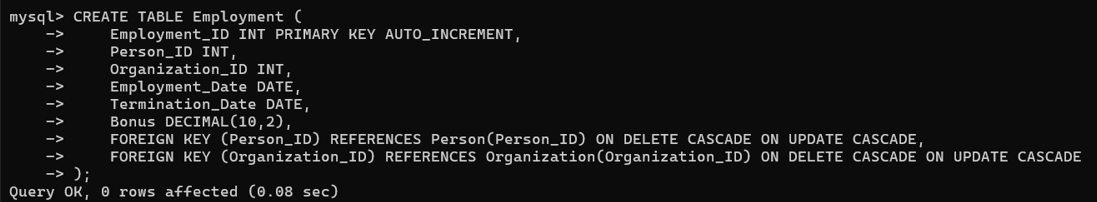
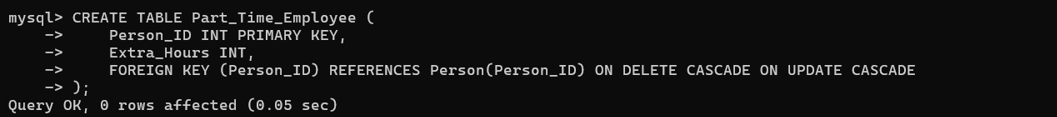


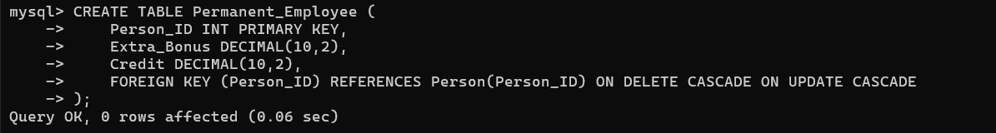
Figure 14 Shows TT\_Holding design using Crows Foot. The ERD above illustrates the relationships between tables in our database. It shows how Person, Job\_Position, Permanent\_Employee, Part\_Time\_Employee, and Permanent\_Employee table are connected, enforcing referential integrity

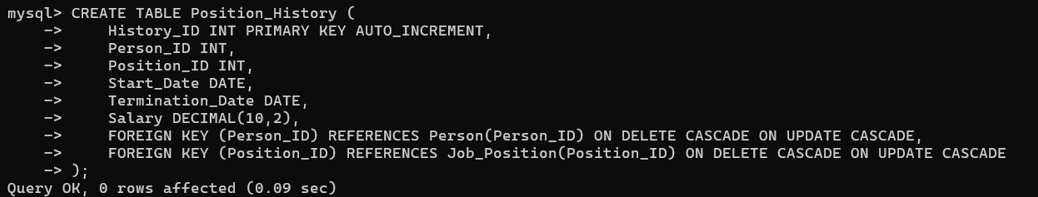
**Appendix B: Database Schema & Table Structures**

In the main report, only selected tables were included to illustrate data insertion and database interactions. To maintain clarity and avoid excessive length, the full set of database tables and SQL statements used for creating are below:





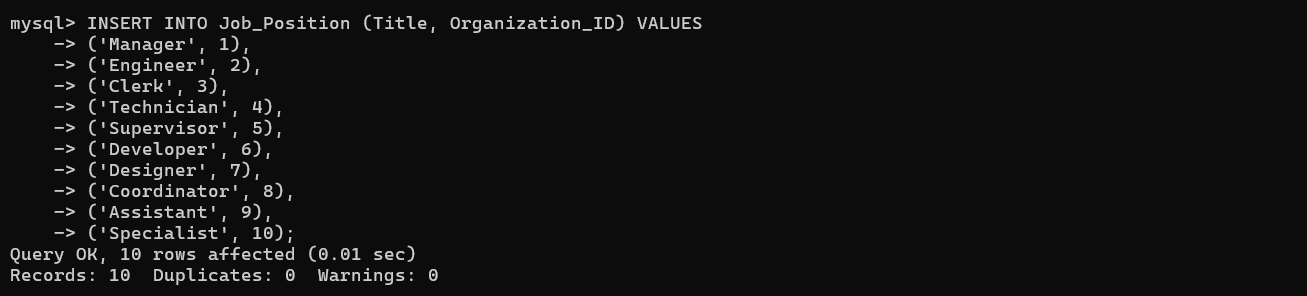


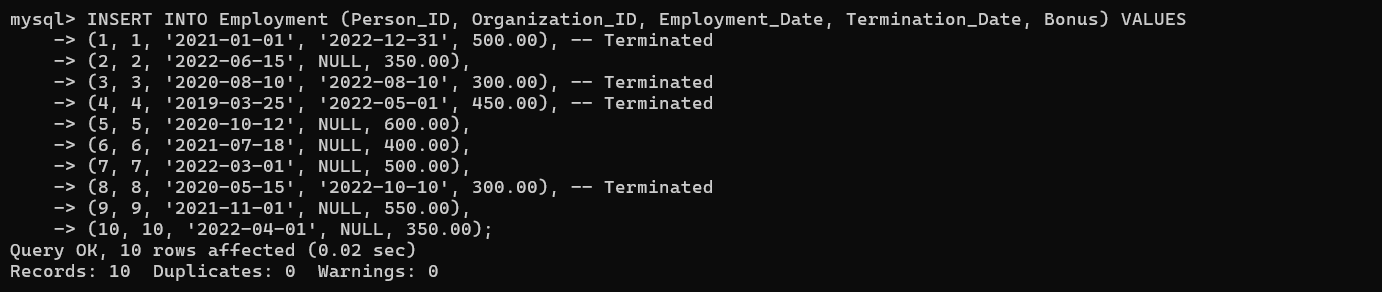


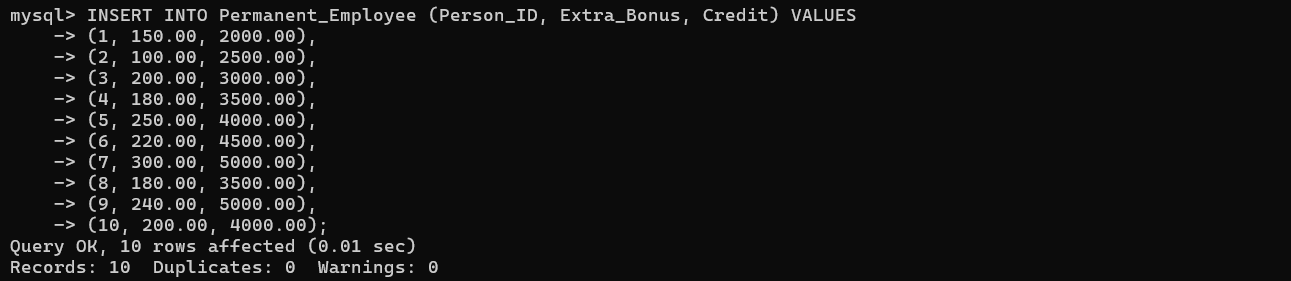
**Appendix C: *Data Insertion Screenshots***

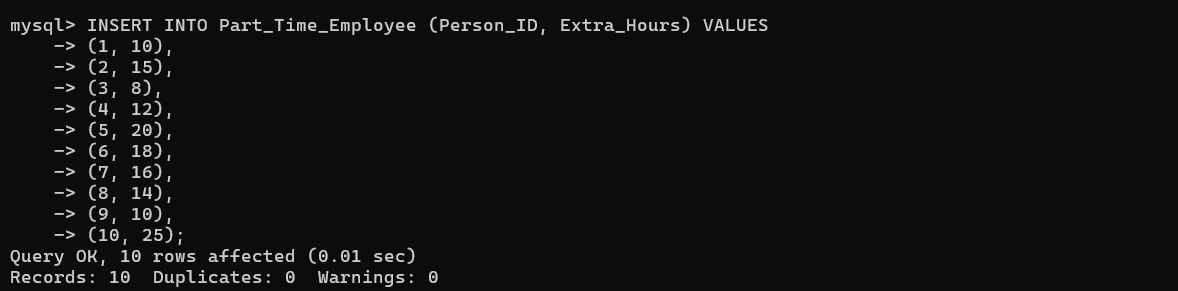
This section provides screenshots of data insertion for all database tables. These images serve as visual proof of successful data entry into the system. The tables include Job Position, Permanent Employee, Part Time Employee, Employment and Position History.

Each screenshot captures the results after executing the INSERT statements, ensuring that the records were correctly added to the database. The data shown aligns with the structured tables defined in the main report.









**Appendix D: *Advanced SQL Implementation***

This section presents **Advanced SQL techniques,** including **Views, Triggers, Functions, and Stored Procedures** used in the database. These implementations ensure data integrity, automate operations, and improve query efficiency.

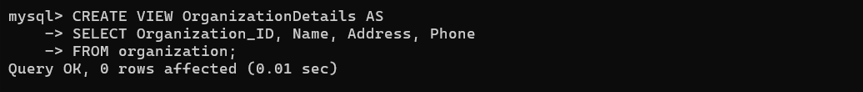
**1. Views**

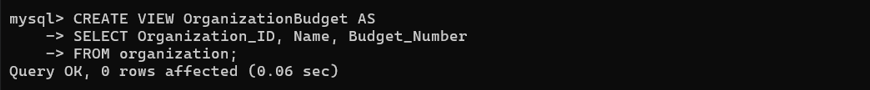
**Person**



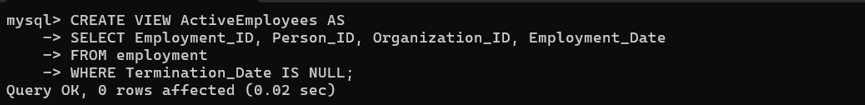


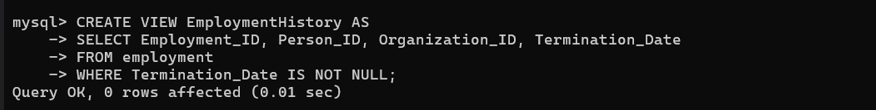
**Organization**



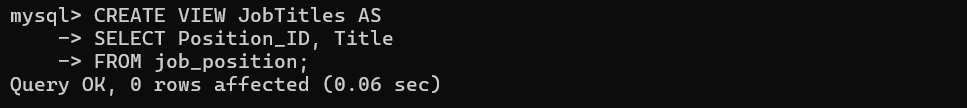


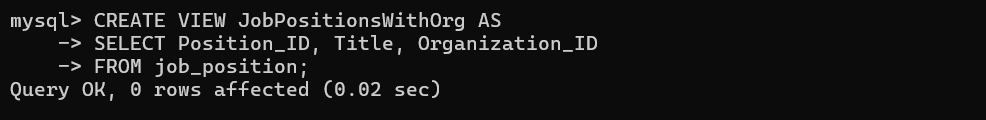
**Employment**



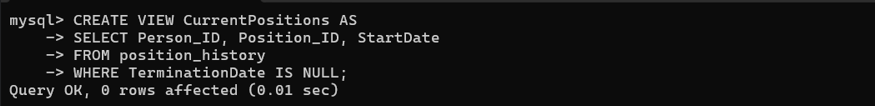


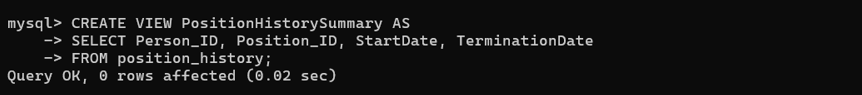
**Job\_Position**

****

****

**Position\_History**



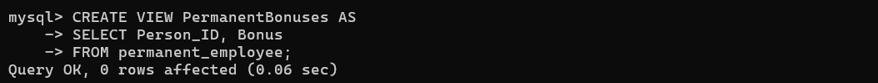


**Part\_Time Employee**





**Permanent Employee**

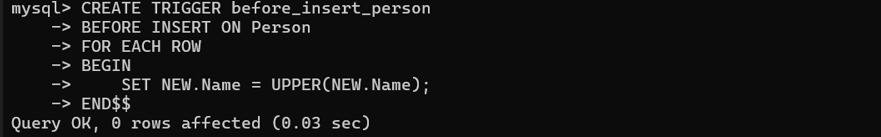


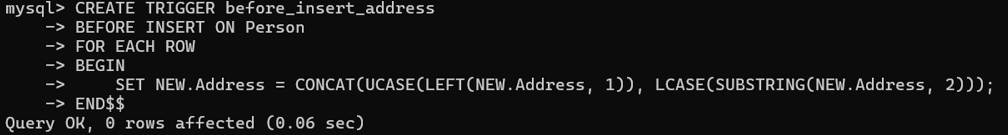


1. **Triggers**

Triggers automatically execute predefined actions when **data is inserted, updated, or deleted**. Below are **two Insert, two Update, and two Delete Triggers** to maintain data integrity.

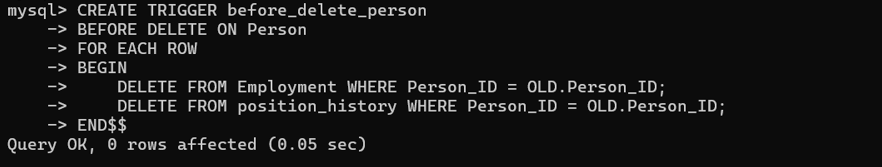
***Insert Triggers***

****

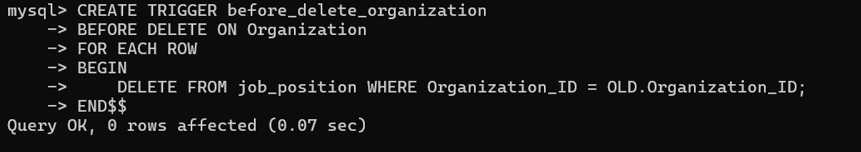
****

***DeleteTriggers***

Before deleting a person from the Person table, this trigger ensures that all associated records in the Employment and position\_history tables are deleted to maintain database integrity.

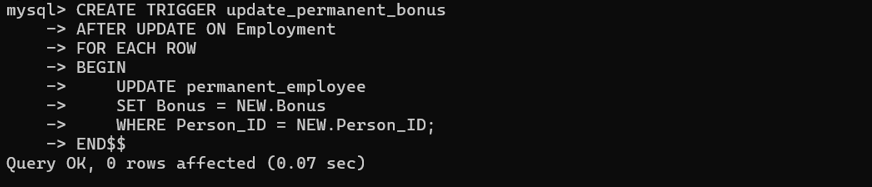
****

Before deleting an organization, this trigger ensures that all related job positions are removed. This prevents orphaned records in the job\_position table.

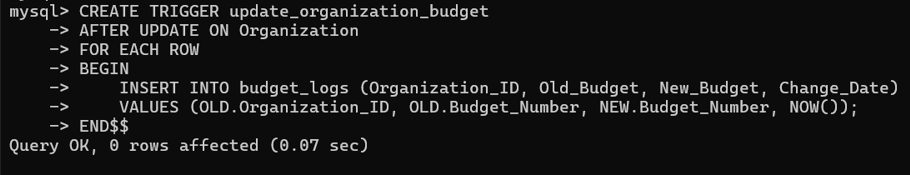
****

***Update Triggers***

When the Employment table is updated, this trigger updates the Bonus field in the permanent\_employee table for the same Person\_ID, ensuring data consistency.



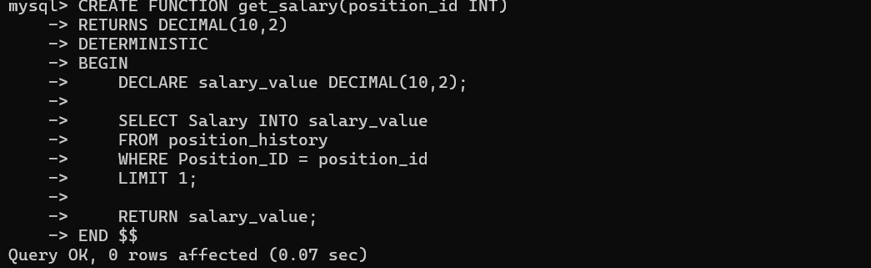
When an organization’s budget is updated, this trigger logs the changes in the budget\_logs table by storing the old and new budget values along with the update timestamp.



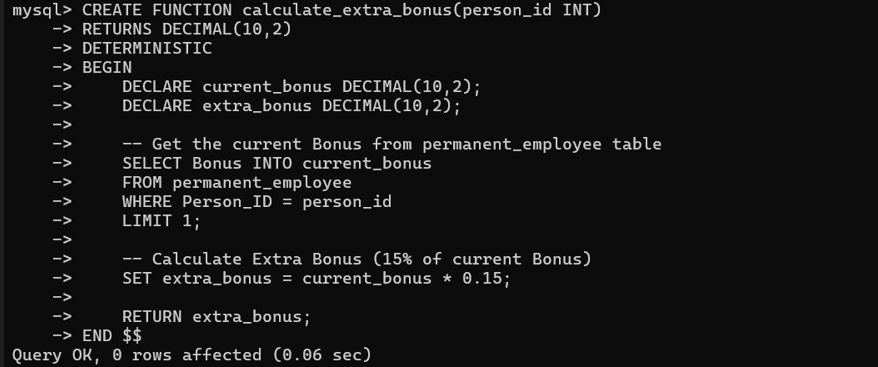
1. **Functions**

Functions return a value based on database input. Below are **two functions** for different tables.

This function below retrieves the salary of a specific job position based on Position\_ID.

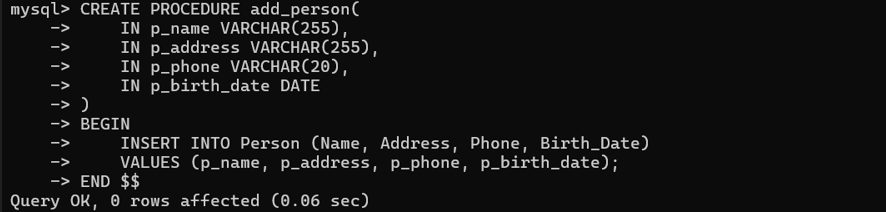
****

Retrieves a person's current bonus and calculates an additional 15% as extra bonus.

****

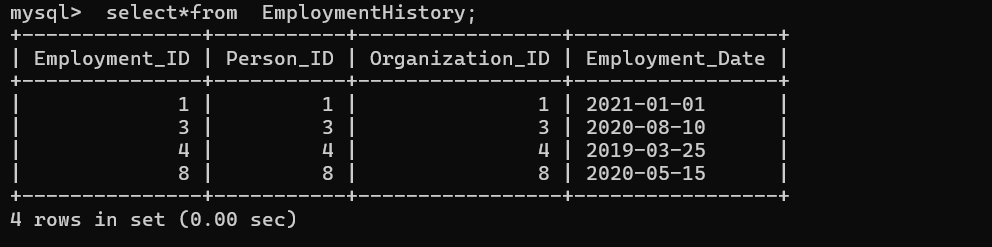
1. **Procedures**

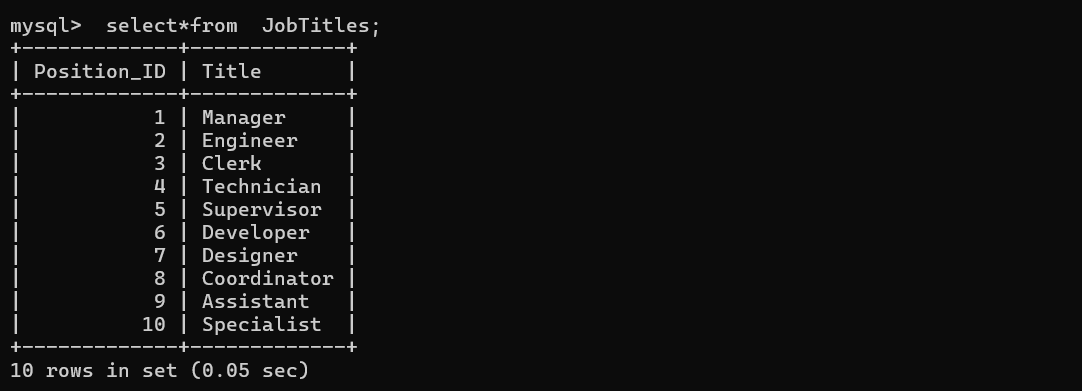
A **procedure** is a reusable block of SQL code that can perform multiple SQL statements.

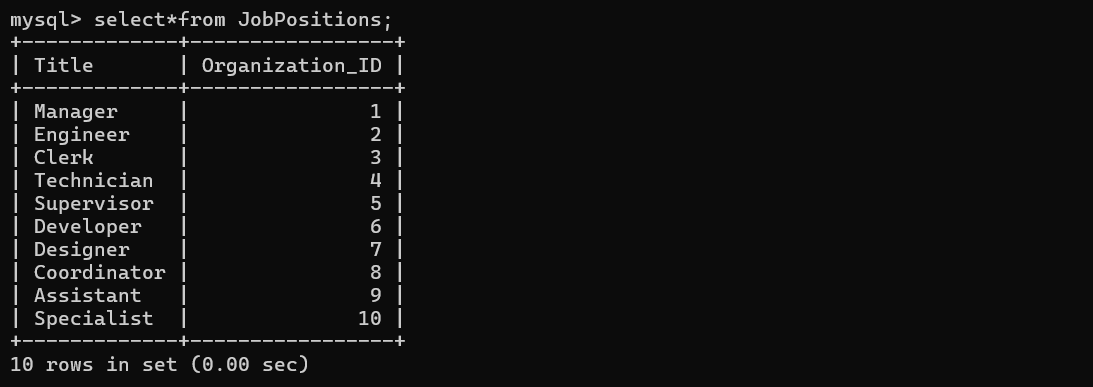
****

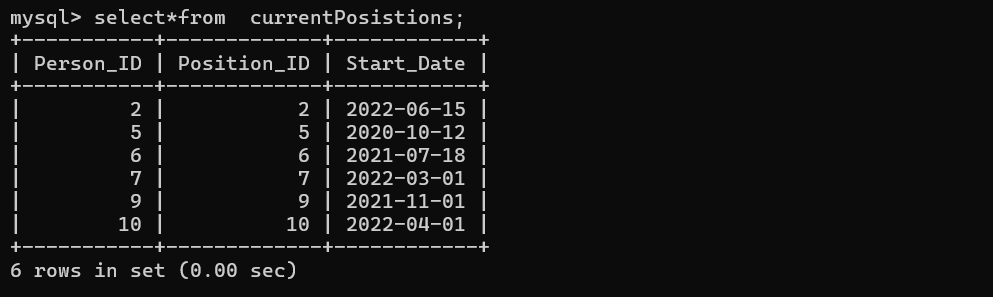
**Appendix E: Results of all table views that were not included on the report**

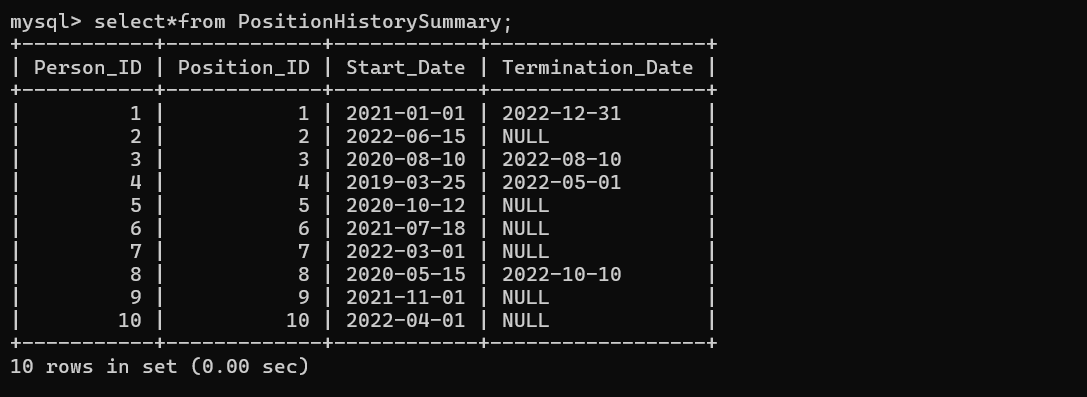
****

****

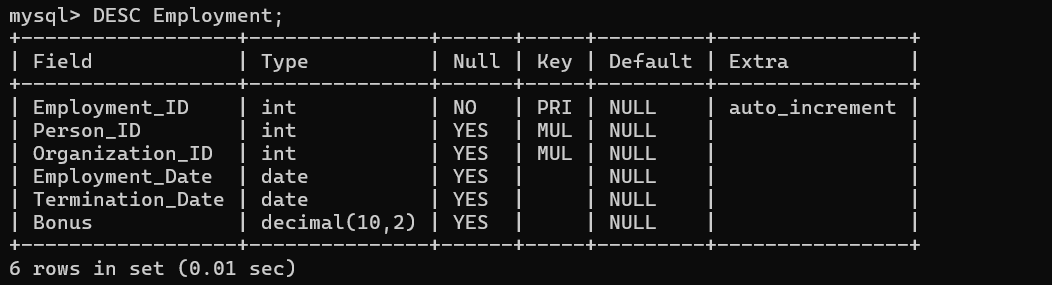
****

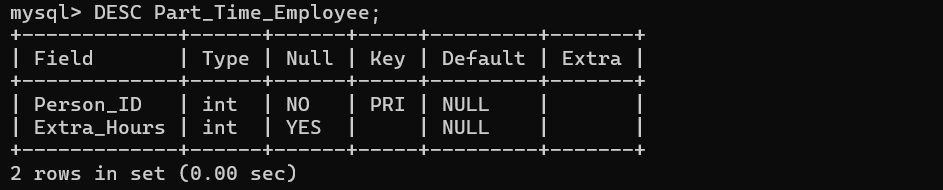
****

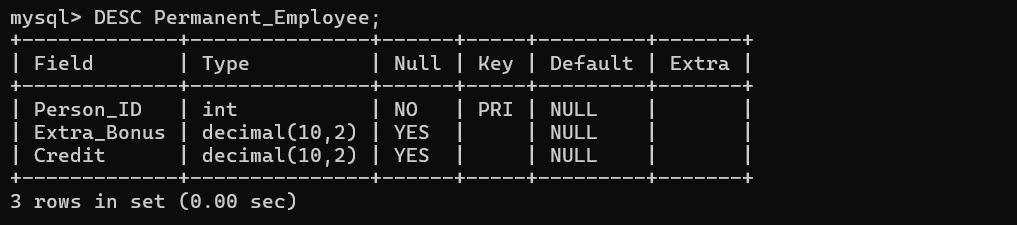
****

****

**Appendix F: Checking Schema & Integrity**

****

****

****

**Appendix G: User Management and Privileges**

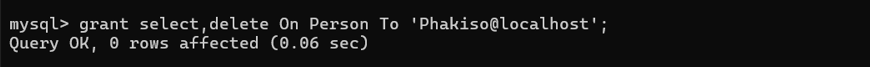
This section presents the **creation of two normal users** in the database and the **granting of two privileges** to each user. User management and privilege assignment are essential for ensuring secure and controlled access to the database system.

**User creation**

****

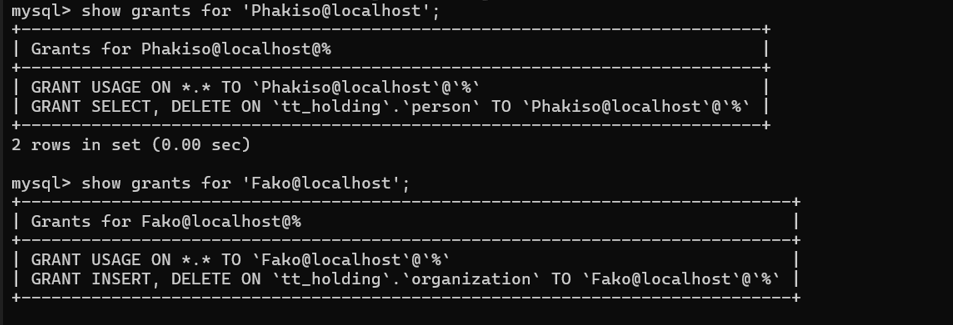
****

**Granting Privileges to users**

****

****

**Showing grants**

****