Payroll Management System Database Design

Nicolás Guevara Herrán
Systems Engineering School
Universidad Distrital Francisco José de Caldas
Bogotá D.C, Colombia
nguevarah@udistrital.edu.co

***Abstract*—This document will give shape to the formulation and development of a project, which aims to solve a legal problem presented by the Trimeca company, industrial and mechanical works, where the main object is the administrative management of both consultations. as an edition of the payroll and everything that it brings with it. Its purpose is to propose and execute the theoretical and practical solution to the problem presented, through specific objectives such as the creation of relational models and everything that this entails, as well as the use of relational algebra to raise possible queries that may arise, whether for consultation or editing, and so on until reaching a concrete solution.**

***Keywords*— Payroll Management System, Database Design, Compliance**

PROJECT DESCRIPTION

For this project, we are presented with a problem related to the mismanagement of information adjacent to the hiring, as well as what this brings with it for each individual such as the position, salary allocation, type of contract, contract time, as well Underlying the mishandling of this, the government has made a legal request towards the company Trimeca SA in the name of the company's lack of compensation to its employees for the hours required by Law 50, which states:

> "ARTICLE 21. Regulated by Decree 1127 of 1991. Add the following article to Chapter II of Title VI, Part One of the Substantive Labor Code:
> Exclusive dedication to certain activities. In companies with more than fifty (50) workers who work forty-eight (48) hours a week, they will have the right to have two (2) hours of said day, at the expense of the employer, be dedicated exclusively to recreational activities, cultural, sports or training."

being clear about article 21 of Law 50, the Trimeca company has raised the need for monetary compensation for the hours not given to employees through monetary payments corresponding to individual salary assignments, taking into account that in a month there are approximately 8 hours, which is equivalent to a working day, the accounts are made on this and in compensation the units will be days, that is, the total number of law 50 hours that must be divided into 8 multiplied by the equivalent daily salary allocation of each employee respectively, they will give us the corresponding monetary compensation.

To carry out this task and the entire general operation, data compendiums managed by the company in Excel will be used, making available the existing contracting data through our program that manages all the data from the entry and exit of a employee, the position, the salary assignment, giving medina formulation outputs so that he has the legal benefits, in addition to providing the monetary compensation for the 50 hours of law not given to the employee through the contracted time and the salary adjustment

User Stories

- As an HR manager, I want to see a list of all employees with their position and department, so that I can better understand the organizational structure.
- As a payroll manager, I want to see all allowances for a specific employee, so that I can ensure they are being paid correctly.
- As a payroll manager, I want to see all deductions for a specific employee, so that I can ensure they are being applied correctly.
- As a payroll administrator, I want to see all payrolls created within a date range, so that I can review the payment history.
- As an HR manager, I want to see a summary of an employee's payroll, including wages, allowances, and deductions, so that I can evaluate their total compensation.
- As the HR director, I want to see a report of departments with the number of employees and their average salary, so that I can identify areas with high salary costs.
- As a supervisor, I want to see a list of employees with their overtime hours and the total amount paid for these hours within a specific date range, so that I can evaluate performance and additional costs.
- As a payroll analyst, I want to see a report of compensations and deductions by payroll type for all employees, so that I can perform financial analysis.
- As an employee, I want to see my payment history, including attendance, absences, wages, allowances, and deductions, so that I can review my payroll records.
- As a system administrator, I want to see a list of all users with their respective roles, so that I can manage permissions and access.
- As a system administrator, I want to see the number of users per role, so that I can ensure the role distribution is appropriate.
- As an HR manager, I want to see the number of employees by each status, so that I can understand the distribution of statuses and manage the workforce efficiently.

Updates

- As an HR manager, I want to update an employee's position, so that I can reflect their promotion in the database.
- As a payroll manager, I want to update an employee's salary, so that their new compensation is accurately recorded.
- As an administrator, I want to change an employee's status to 'on leave', so that their leave period is documented correctly.
- As an employee, I want to update my email address, so that I can receive important notifications and information.
- As an administrator, I want to update a user's role from 'user' to

‘admin’, so that they have the necessary permissions to manage the system.

- As a payroll manager, I want to update the number of extra hours an employee has worked, so that their overtime pay is calculated correctly.
- As an employee, I want to change my phone number in the system, so that my contact information is current.
- As HR manager, I want to correct a typo in the job title of a position, so that all records are accurate and professional.

Deletes

- As an HR manager, I want to delete an employee who has left the company, so that the records are up-to-date.
- As a payroll manager, I want to delete an outdated allowance entry for an employee, so that the payroll calculations are accurate.
- As an HR manager, I want to remove a department that has been dissolved, so that the organizational structure reflects current operations.
- As a payroll manager, I want to delete incorrect deduction entries for an employee, so that their payroll is corrected.
- As an HR manager, I want to delete extra hours record that was mistakenly added, so that the payroll is accurate.
- As a payroll manager, I want to remove old payroll records that are past the retention period, so that the database remains efficient.
- As a payroll manager, I want to delete payroll records for a specific date range that were entered incorrectly, so that the payroll data is accurate.

Views

- As a payroll manager, I want to have a consolidated view that shows the payroll summary for each employee, including their base salary, total allowances, total deductions, and net salary, so that I can quickly review payroll information.
- As an HR manager, I want a view that shows the number of employees and the average salary for each department, so that I can assess the distribution and cost of human resources across departments.
- As a system administrator, I want a view that logs user activities, including login times, actions performed, and the roles of users, so that I can monitor system usage and detect any unusual activities.
- As a finance manager, I want to have a view that shows the total salary budget for each department, including base salaries and allowances, so that I can manage departmental budgets effectively.
- As a payroll manager, I want to have a view that shows the attendance summary for each employee, including the number of days present and absent, so that I can analyze attendance patterns and make necessary adjustments.

Triggers

- As a payroll manager, I want the status of payroll records to automatically change from “new” to “computed” after a certain period, so that the payroll processing state is accurately reflected.
- As a system administrator, I want to log all changes (insert, update, delete) made to the ‘employee’ table, so that I can monitor and audit modifications to employee records.
- As a payroll manager, I want to prevent any updates or inserts that would set an employee’s salary to a negative value, so that the data integrity is maintained.
- As an HR manager, I want all related records in ‘employee_allowances’, ‘employee_deductions’, and ‘employee_extra_hours’ to be automatically deleted when an employee record is deleted, so that there are no orphan records left in the database.

METHODS AND MATERIALS

Step 0. Components

Role, Employee, Status, Allowance, Payroll, Deductions, Position, Department.

Step 1. Entities

Employee, Payroll, Department, Position, Allowances, Deductions, Employee Allowances, Employee Deductions, User, Role, Pay Slip, Status, Payroll Type, Employee_Extra_Hours

Step 2. Attributes

E1. Employee: UUID, employee_code, name, department_FK, position_FK, salary, email, status_FK.

E2. Payroll: ID, reference_number, date_from, date_to, payroll_type_FK, status, date_created.

E3. Department: ID, name, description.

E4. Position: ID, department_FK, name, description.

E5. Allowance: ID, deduction_name, description.

E6. Deduction: ID, deduction_name, description.

E7. Employee_Allowances: ID, employee_FK, Allowance_FK, Payroll_Type_FK, amount, effective_date, date_created.

E8. Employee_Deduction: ID, employee_FK, deduction_FK, payroll_type_FK, amount, effective_date, date_created.

E9. UserActivityLog: ID, action, action_timestamp, employee_fk

E10. Role: ID, name, permissions.

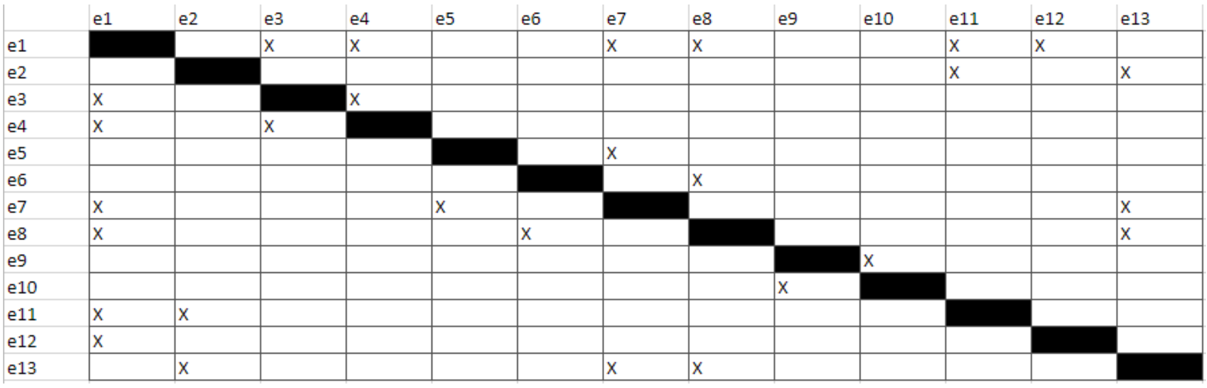
E11. PaySlip: ID, payroll_FK, employee_FK, present, absent, salary, allowance_amount, deduction, deduction_amount, net, date_created.

E12. Status: ID, name, comments.

E13. Payroll_type: ID, name.

E14. Employee_Extra_Hours: ID, Employee_FK, hours, Payroll_Type_FK, amount, Effective_Date, Date_Created

Step 3. Relations



	e1	e2	e3	e4	e5	e6	e7	e8	e9	e10	e11	e12	e13
e1	■		X	X			X	X			X	X	
e2		■									X		X
e3	X		■	X									
e4	X		X	■									
e5					■		X						
e6						■		X					
e7	X				X		■						X
e8	X					X		■					X
e9									■	X			
e10									X	■			
e11	X	X									■		
e12	X											■	
e13		X					X	X					■

Step 4. Relationships Types

E1	nx-x1	E3	E2	nx-x1	E13
E1	nx-x1	E4	E3	1x-xn	E4
E1	1x-xn	E7	E5	1x-xn	E7
E1	1x-xn	E8	E6	1x-xn	E8
E1	1x-xn	E11	E7	nx-x1	E13
E1	nx-x1	E12	E8	nx-x1	E13
E2	1x-xn	E11	E9	nx-x1	E10
E1	1x-xn	E14	E13	1x-xn	E14

Results

The payroll management system for Trimeca S.A. was successfully developed and implemented. Key results include:

Database Design: A comprehensive database structure was created to manage employees, departments, positions, allowances, deductions, payrolls, payslips, and user activities. The Entity-Relationship (ER) diagram illustrates the robust relationships between these entities, ensuring efficient data management and retrieval.

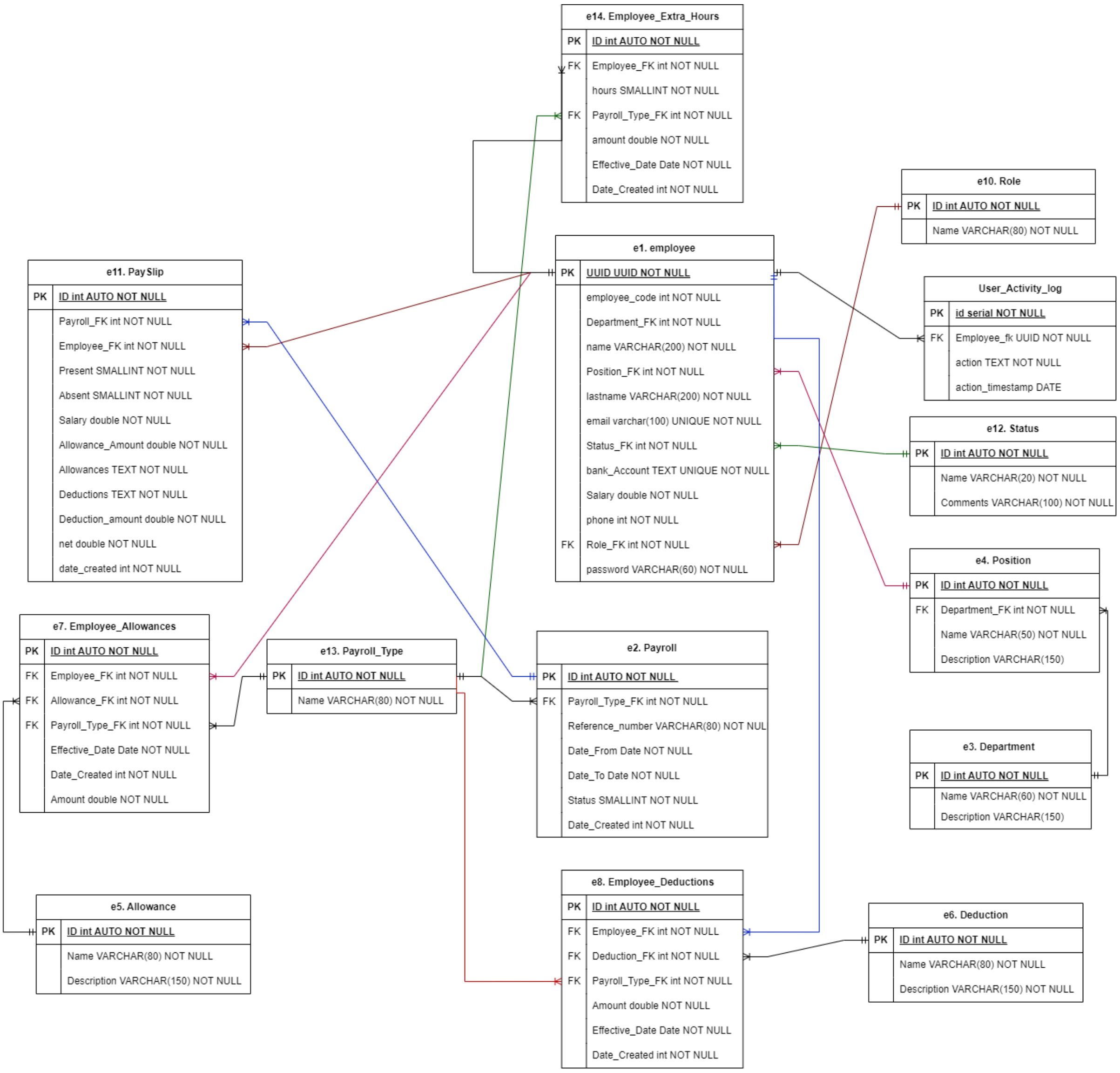
Data Population: The database was populated with realistic dummy data using the Faker library and custom scripts. This data includes employees, departments, positions, payroll records, allowances, deductions, extra hours, and user activities.

Web Services: A suite of RESTful web services was developed using FastAPI, SQLAlchemy, and PostgreSQL. These services provide endpoints for retrieving, adding, updating, and deleting data across various entities, facilitating seamless interaction between the front-end and the database.

Views and Reports: Several database views were created to provide consolidated reports, such as payroll summaries for employees, department employee summaries, department salary budgets, and employee attendance summaries. These views enable efficient data analysis and decision-making.

Triggers and Functions: PostgreSQL triggers and functions were implemented to automate status updates for payroll records, log changes to the employee table, prevent negative salary values, and handle cascading deletes for related records. These ensure data integrity and automate routine processes.

The resulting ER diagram is the following.


e14. Employee_Extra_Hours
PK ID int AUTO NOT NULL
FK Employee_FK int NOT NULL
hours SMALLINT NOT NULL
FK Payroll_Type_FK int NOT NULL
amount double NOT NULL
Effective_Date Date NOT NULL
Date_Created int NOT NULL
e10. Role
PK ID int AUTO NOT NULL
Name VARCHAR(80) NOT NULL
User_Activity_log
PK id serial NOT NULL
FK Employee_fk UUID NOT NULL
action TEXT NOT NULL
action_timestamp DATE
e11. PaySlip
PK ID int AUTO NOT NULL
Payroll_FK int NOT NULL
Employee_FK int NOT NULL
Present SMALLINT NOT NULL
Absent SMALLINT NOT NULL
Salary double NOT NULL
Allowance_Amount double NOT NULL
Allowances TEXT NOT NULL
Deductions TEXT NOT NULL
Deduction_amount double NOT NULL
net double NOT NULL
date_created int NOT NULL
e1. employee
PK UUID UUID NOT NULL
employee_code int NOT NULL
Department_FK int NOT NULL
name VARCHAR(200) NOT NULL
Position_FK int NOT NULL
lastname VARCHAR(200) NOT NULL
email varchar(100) UNIQUE NOT NULL
Status_FK int NOT NULL
bank_Account TEXT UNIQUE NOT NULL
Salary double NOT NULL
phone int NOT NULL
FK Role_FK int NOT NULL
password VARCHAR(60) NOT NULL
e12. Status
PK ID int AUTO NOT NULL
Name VARCHAR(20) NOT NULL
Comments VARCHAR(100) NOT NULL
e4. Position
PK ID int AUTO NOT NULL
FK Department_FK int NOT NULL
Name VARCHAR(50) NOT NULL
Description VARCHAR(150)
e7. Employee_Allowances
PK ID int AUTO NOT NULL
FK Employee_FK int NOT NULL
FK Allowance_FK int NOT NULL
FK Payroll_Type_FK int NOT NULL
Effective_Date Date NOT NULL
Date_Created int NOT NULL
Amount double NOT NULL
e13. Payroll_Type
PK ID int AUTO NOT NULL
Name VARCHAR(80) NOT NULL
e2. Payroll
PK ID int AUTO NOT NULL
FK Payroll_Type_FK int NOT NULL
Reference_number VARCHAR(80) NOT NUL
Date_From Date NOT NULL
Date_To Date NOT NULL
Status SMALLINT NOT NULL
Date_Created int NOT NULL
e3. Department
PK ID int AUTO NOT NULL
Name VARCHAR(60) NOT NULL
Description VARCHAR(150)
e5. Allowance
PK ID int AUTO NOT NULL
Name VARCHAR(80) NOT NULL
Description VARCHAR(150) NOT NULL
e8. Employee_Deductions
PK ID int AUTO NOT NULL
FK Employee_FK int NOT NULL
FK Deduction_FK int NOT NULL
FK Payroll_Type_FK int NOT NULL
Amount double NOT NULL
Effective_Date Date NOT NULL
Date_Created int NOT NULL
e6. Deduction
PK ID int AUTO NOT NULL
Name VARCHAR(80) NOT NULL
Description VARCHAR(150) NOT NULL

CONCLUSIONS

The development of the payroll management system for Trimeca S.A. demonstrated the effective application of database design principles, web service implementation, and automated data management techniques. Key conclusions drawn from this project are:

Efficient Data Management: The well-structured and normalized database design facilitated efficient data management, ensuring data consistency, integrity, and quick retrieval. The comprehensive ER diagram and metadata documentation provided a clear blueprint for database construction and maintenance.

Seamless Integration: The RESTful web services enabled seamless integration between the front-end interface and the database. These services catered to various user needs, such as managing employee records, generating payrolls, and producing detailed reports, thereby enhancing user experience and operational efficiency.

Robust Reporting: The implementation of views and reports provided valuable insights into the organization's payroll data. These reports supported strategic decision-making by highlighting key metrics such as employee payroll summaries, department budgets, and attendance records.

Automated Processes: The use of triggers and functions automated several critical processes, such as updating payroll statuses, logging employee changes, and preventing invalid data entries. This automation reduced manual intervention, minimized errors, and ensured data integrity.

Scalability and Flexibility: The system's design allows for scalability and flexibility, making it adaptable to future requirements and changes. New features and data entities can be incorporated with minimal disruption to the existing system, ensuring long-term usability and relevance.

Overall, the payroll management system project successfully met its objectives, delivering a robust, efficient, and scalable solution for Trimeca S.A.'s payroll processing needs. The integration of advanced database management techniques and web services ensured a comprehensive and user-friendly system, significantly improving the organization's payroll operations.

BIBLIOGRAPHY

- Tiangolo, S. (n.d.). *FastAPI Documentation*. Retrieved from https://fastapi.tiangolo.com/

- Bayer, M. (n.d.). *SQLAlchemy Documentation*. Retrieved from https://www.sqlalchemy.org/

- PostgreSQL Global Development Group. (n.d.). *PostgreSQL Documentation*. Retrieved from https://www.postgresql.org/docs/

- Samuel, H. (n.d.). *Pydantic Documentation*. Retrieved from https://pydantic-docs.helpmanual.io/

- Uvicorn. (n.d.). *Uvicorn Documentation*. Retrieved from https://www.uvicorn.org/

- Revesz, J. (n.d.). *Faker Documentation*. Retrieved from https://faker.readthedocs.io/

- Varrazzo, D. (n.d.). *Psycopg2 Documentation*. Retrieved from https://www.psycopg.org/docs/

- DBeaver. (n.d.). *DBeaver Documentation*. Retrieved from https://dbeaver.io/docs/

- Bayer, M. (n.d.). *SQLAlchemy Core Tutorial*. Retrieved from https://docs.sqlalchemy.org/en/14/core/tutorial.html

- PostgreSQL Global Development Group. (n.d.). *PostgreSQL Triggers and Functions*. Retrieved from https://www.postgresql.org/docs/current/plpgsql-trigger.html