Names: Harolt Farinango, Mateo Pilaquinga

DESIGN AND IMPLEMENTATION OF A DATABASE FOR A CAR INSURANCE BUSINESS

1. Objectives

Understand the complete database design cycle, from conceptualization to implementation, applied to the context of an insurance agency.

Apply theoretical database knowledge in practical scenarios, managing customers, policies, claims, and payments.

Gain practical experience in the design, implementation, and administration of a relational database system for a real insurance business.

Develop critical thinking and problem-solving skills when addressing complex relationships, constraints, and reporting needs inherent in insurance operations.

2. Description

In a company focused on automobile insurance, the correct management and quality of information is essential. The database must handle customers, vehicles, policies, premium payments, and claims efficiently. Without a structured system, critical processes are affected, such as knowing policy expiration dates, calculating premium balances, monitoring collections, and registering claims in a traceable way. To address this need, the project proposes the design and implementation of a relational database in MySQL, deployed within a Docker container to guarantee portability and ease of administration. The system will maintain referential integrity between key entities and allow the execution of queries for essential reports, such as active policies, payment status, or claims history.

The implementation will also establish minimal business rules to ensure consistent and reliable data. The database will serve as the backbone for policy and claims management in the insurance agency, supporting decision-making and operational efficiency. Customers will be linked to vehicles and policies; policies will connect to coverages, agents, and claims; and claims will be associated with payments. This structure provides a solid foundation for accurate reporting, traceability of transactions, and monitoring of obligations and risks. Overall, this system will guarantee the reliability of the information, reduce operational errors, and improve the agency’s ability to serve clients and meet regulatory and business needs effectively.

Requirements

Functional (Operational) Requirements

1. Manage clients: Register/update unique client information.
2. Manage vehicles: Register/update information, linked to clients.
3. Manage agents: Register/update information, associated with policies.
4. Manage coverage: Register/update coverage details for claims and policies.
5. Policies: Create/modify policies linked with clients, vehicles, agents, and claims. Each policy must include ID, dates, status, premium, and deductible.
6. Associate coverages to policies with insured sums and premiums.
7. Validate policy dependencies before activation or updates.
8. Change policy states (active, expired, canceled, overdue).
9. Manage payments of premiums (amount, method, due and payment dates).
10. Query outstanding balances per policy.
11. Register claims with ID, occurrence/report date, state, reserve, description, and location.
12. Register claim payments (date, amount, beneficiary).
13. Query total payments by claim and by policy.

Non-Functional (Quality) Requirements

1. Portability: The system must run in a Dockerized MySQL environment.
2. Data Integrity: Enforce referential integrity and business rules for consistency.
3. Security: Ensure controlled access to data and secure operations.
4. Performance: Queries must return results in acceptable time frames for business use.
5. Reliability: Guarantee database availability and minimal downtime.
6. Maintainability: Database schema must support future modifications and scalability.