Implementation of database systems for order management for Mi Pymes

Carmen Sofia Florez Juajibioy
Systems Engineer- Database Designer
Universidad Distrital Fracisco José de Caldas
Bogota D.C, Colombia
Email: sofiflorez@gmail.com

***Abstract*—In the context of order management in stores, the problem focuses on the need to process transactions efficiently in a market that demands speed and precision. The proposed solution is the design of a relational database, based on the entity-relationship modeling principles learned in the database fundamentals course, to optimize and automate the management process. As a result, the generation of a set of functional SQL queries is anticipated, demonstrating the database's ability to effectively handle ordering operations.**

I. Introduction

Efficient order management is a critical challenge for commerce in the digital age, where speed and accuracy are essential for customer satisfaction and market competitiveness. As a systems engineering student, this project is inspired by the knowledge acquired in the subject of database fundamentals, to address this problem from a technical and theoretical perspective, with the aim of designing and creating a database that optimizes order management in stores of various types.

Historically, stores have handled order management through manual methods or basic computer systems, which are often inefficient in the face of current demand. With the advancement of technology, more sophisticated solutions have been developed that seek to automate and improve this process. Previous studies have demonstrated the effectiveness of systems such as ERP (Enterprise Resource Planning) and CRM (Customer Relationship Management) in improving order management (Smith, 2018; Johnson, 2019). However, these solutions can be expensive and complex, limiting their accessibility for MSMEs.

In response to these limitations, this project proposes a solution based on the collection of detailed requirements and the definition of user stories in simple sentence format, allowing a clear and direct understanding of the needs of the end user. From this information, a database is designed using entity-relationship modeling techniques, a methodology established by Chen (1976), which remains relevant today for its simplicity and effectiveness in representing data and their interrelationships.

The design of the database is based on normalization principles and relational calculation techniques, ensuring the integrity and efficiency of the data (Codd, 1970). Database implementation is done through an interactive and hands-on approach, using SQL for data manipulation and creating queries that reflect actual business operations.

II. DEVELOPMENT

A. *Descripción del Proyecto*

This project focuses on the design and implementation of a relational database for an in-store order management application, with the objective of improving efficiency and accuracy in transaction processing. The project originates as an academic initiative within the framework of the database fundamentals course in systems engineering, where theoretical knowledge and practical skills are applied to address a real challenge in the commercial sector.

Stakeholders: The stakeholders identified for this project are two main groups:

- Customers: The primary group of interest are customers who place orders in stores. Your direct experience with the order management system is essential, and your satisfaction is a key indicator of project success. Customer requirements and expectations have been collected to ensure that the designed database meets their needs for an intuitive interface and a hassle-free ordering process.

- Store Managers or Owners: The second interest group is the store managers or owners, who are in charge of the daily administration and operation. This group directly benefits from a database that offers efficient order management, accurate reporting and better inventory control. They have been involved in the design process to ensure that the proposed solution aligns with your business and operational objectives.

The design of the database will be carried out using the entity-relationship model to structure the information in a logical and coherent manner, thus facilitating order management and data-based decision making. The implementation is carried out in MySQL, taking advantage of its robustness and flexibility to develop SQL queries that demonstrate the functionality of the system.

III. CONCEPTUAL MODEL

The methodology adopted for the design of the database is based on the theoretical and practical principles of the entity-relationship model, an essential tool in systems engineering for the structured representation of information. This methodological approach contributes to the comprehensive development of the order management system.

A. User stories

A list of user stories is created, which are short and simple descriptions of a feature or function of the system from the perspective of the end user.

- As a customer, I want to browse a diverse and updated product catalog, to easily find what I need and make my purchases.

- As a customer, I want to place orders quickly and safely, to enjoy a convenient shopping experience without leaving home.

- As a store manager, I want to receive immediate alerts when orders are placed, to ensure a quick response and improve service efficiency.

- As a store manager, I want to consult customers' purchase history and preferences, to offer personalized service and improve customer satisfaction.

- As a store manager, I want to obtain detailed sales reports and data analysis, to identify trends and make strategic decisions based on reliable information.

- As a store manager, I want an effective order tracking system, to ensure on-time delivery and maintain customer trust.

IV. CREATION OF THE CONCEPTUAL MODEL

Establishes the fundamental structure on which the database will be built. This process is carried out through a 10-step ontology methodology, which ensures a design of the entity-relationship (ER) model.

A. Identification of entities

It begins by identifying all the entities relevant to the order management system.

- E1: Customer
- E2: Manager
- E3: Order
- E4: Ordered Item
- E5: Product
- E6: Category
- E7: Transaction
- E8: Delivery
- E9: Store

B. Definition of attributes

For each identified entity, its attributes are defined, which are the properties or characteristics that describe the entity.

- Customer: customer-id(PK), name-customer, lastname-customer, email-customer, address-customer, phone-customer, registration-date
- Manager: manager-id(PK), manager-name, manager-lastname, manager-email, manager-phone, store-id (FK)
- Order: order-id (PK), customer-id (FK)), store-id (FK), order-status, order-date
- Ordered-Item: ordered-item-id (PK), order-id (FK), product-id(FK), quantity, unit-price
- Product: product-id (PK), product-name, product-price, stock-quantity, category-id(FK)
- Category: category-id (PK), category-name, category-description
- Transaction: transaction-id(PK), order-id(FK), transaction-date, total-amount, payment-method
- Delivery: delivery-id (PK), order-id (FK), delivery-address, delivery-date, delivery-instructions
- Store: store-id (PK), store-name, store-address, store-phone, store-email

C. Identification of Relationships

The relationships between entities are identified, which are associations that exist between two or more entities, for this purpose a relationship matrix or also called adjacency matrix is used, where:

- The rows represent the entities from which the relationship originates.
- The columns represent the entities where the relationship is directed.
- An "X" indicates that a relationship exists between the entities in the row and column.
- A "///" indicates that there is no direct relationship.

TABLE I
RELATIONSHIP MATRIX

	E1	**E2**	**E3**	**E4**	**E5**	**E6**	**E7**	**E8**	**E9**
E1	///		X						
E2		///							X
E3			///	X			X	X	X
E4			X	///	X				
E5					///	X			
E6					X	///			
E7			X				///		
E8			X					///	
E9		X	X		X				///

- Customer to Order: E1 to E3
- Manager to Store: E9 to E9
- Order to Ordered-Item: E3 to E4
- Ordered-Item to Product: E4 to E5
- Product to Category: E5 to E6
- Transaction to Order: E7 to E3
- Delivery to Order: E8 to E3
- Store to Order: E9 to E3

D. Definition of the type of relationships

Cardinality is defined, which describes the number of instances of an entity that can be associated with instances of another entity.

- Customer to Order: (1-n)
- Manager to Store: (1-1)
- Order to Ordered-Item: (1-n)
- Ordered-Item to Product: (1-n)
- Product to Category: (n-1)

- Transaction to Order: (1-1)
- Delivery to Order: (1-1)
- Store to Order: (1-n)

One-to-one (1-1) relationships are characterized by the exclusive association between an entity in A and an entity in B. In contrast, one-to-many (1-n) relationships allow an entity in A to be associated with multiple entities of B, reflecting situations such as a customer with several orders.

E. *Defining constraints and data properties*

Defining constraints and data properties is essential in building the database for the order management application. These aspects, essential for the integrity and performance of the system, will be presented visually in the attached entity-relationship diagram. This diagram illustrates the structure and relationships between entities, offering a more intuitive and direct understanding of the designed database, avoiding redundancies in the text.

F. *Construction of the ER Diagram*

With all the information collected, the ER diagram is constructed. This visual diagram represents the entities, their attributes, and the relationships between them, providing a clear and structured view of the data model. (Fig.1)

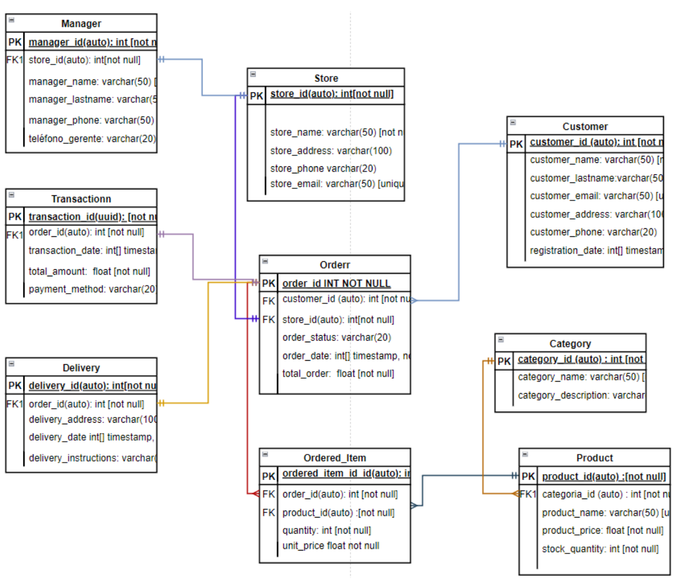

Fig. 1. E-R Diagram

amsmath

V. EXPERIMENTS

A. *Relational algebra queries*

In this chapter, queries performed using relational algebra are presented, a powerful tool for manipulating and querying databases. These experiments not only demonstrate the functionality of the proposed entity-relationship model, but also validate the efficiency and accuracy of relational algebra operations in data retrieval and management. The selected queries illustrate the practical application of the theory and provide an empirical basis for database design evaluation.

- To allow customers to browse the product catalog, a projection operation is performed on the "Product" entity, selecting the relevant attributes such as product ID, name, price and quantity in stock. This ensures that customers can see a complete list of available products along with their basic information.
  Consultation:
  $\pi_{product\_id,product\_name,product\_price,stock\_quantity,category}$ $\_id(Product)$
- When filtering products by category, a selection is applied on the "Product" entity to include only those that belong to the specific desired category. This makes it easier for customers to find specific products within a given category.
  Consultation:
  $\sigma_{category\_id=value\_category}(Product)$
- Searching for products by name involves a selection in the "Product" entity, where products whose names partially or fully match the search term provided by the customer are searched. This allows customers to easily find the products they want even if they don't remember the full name
  Consultation:
  $\sigma_{product\_name \text{ LIKE } \%value\_name\%}(Product)$
- When displaying details for a specific order, a selection is made on the "Order" entity to include only orders associated with the corresponding customer ID. This allows customers to quickly review the status and details of their previous orders.
  Consultation:
  $\pi_{order\_id,order\_status,order\_date,total\_order}(\sigma_{customer\_id=}$ $value\_customer\_id(Orderr))$
- When displaying the details of the items ordered in a specific order, a selection is made on the "Item Ordered" entity to include only the items associated with the corresponding order ID. This allows customers to see what products they have ordered and in what quantity.
  Consultation:
  $\pi_{ordered\_item\_id,product\_id,quantity,unit\_price}(\sigma_{order\_id=}$ $value\_order\_id(Ordered\_Item))$
- When searching for new orders that need attention, a selection is applied on the "Order" entity to include only orders that are in "new" status. This alerts store managers to recent orders that require immediate action.
  Consultation:
  $\pi_{order\_id,customer\_id,store\_id,}$ $order\_status, order\_date, total\_order(\sigma_{order\_status=}$ $'new'(Orderr))$
- When viewing the purchase history of a specific customer, a selection is made on the "Order" entity to include only orders associated with the corresponding customer ID. This provides store managers with valuable information about customers' purchasing preferences.

Consultation:
$\pi_{order\_id,order\_date,total\_order}(\sigma_{customer\_id=value\_customer\_id}(Orderr))$

- When generating detailed sales reports, an aggregation operation is performed on the "Order" entity to calculate the total sales by date. This provides store managers with key information on sales performance over different periods.
Consultation:
$\gamma_{order\_date,\mathrm{SUM}(total\_order)}(Orderr)$
- When identifying sales trends by product, an aggregation operation is performed on the "Ordered Item" entity to add the quantities sold by product. This helps store managers understand which products are most popular with customers.
Consultation:
$\gamma_{product\_id,\mathrm{SUM}(quantity)}(Ordered\_Item)$
- When querying the status and delivery information for a specific order, a join operation is performed between the "Order" and "Address" entities to include only the details of the selected order. This provides store managers with a complete view of the delivery process and helps ensure customer satisfaction.
Consultation: $\pi_{order\_id,order\_status,delivery\_date,delivery\_address,}$ $delivery\_instructions(\sigma_{order\_id=value\_order\_id}(Delivery) \bowtie \sigma_{order\_id=value\_order\_id}(Orderr))$

VI. RESULTS

After the meticulous design phase of the database, it was implemented in MySQL. The database was populated with fictitious data to validate its operation in simulated situations. Below are the results obtained from various queries, views and triggers, highlighting their impact on operational efficiency and customer satisfaction.

- ProductCatalogView: To obtain a clear and organized list of products available in the store catalog, the ProductCatalogView fulfills that function. It provides customers with a diverse and up-to-date product catalog, allowing them to easily navigate the available options and find what they need efficiently.

- Insertorder: To make it easier for customers to place new orders quickly and securely, the Insertorder stored procedure automates the order registration process. This ensures a hassle-free shopping experience and improves overall system efficiency.

- NotifyManagerOnNewOrder: When a new order is placed in the store, the NotifyManagerOnNewOrder trigger triggers an immediate notification to the manager. This feature allows the management team to stay on top of store activity and respond quickly to customer needs.

- UpdateInventoryOnOrderCompletion: After an order is completed, the UpdateInventoryOnOrderCompletion trigger is responsible for automatically updating the store inventory. This ensures that stock levels are always up to date and available to meet customer demand.

The successful implementation of the relational database and the designed SQL queries is reflected in the results obtained during order management in the store. The data collected and processed demonstrates a significant improvement in operational efficiency and responsiveness to market demands. Automating certain processes, such as reporting or updating inventory, thanks to crafted SQL queries, has the potential to further improve overall store efficiency and allow employees to focus on more strategic tasks.

For example, monthly sales reports provide a clear view of purchasing trends, allowing you to identify behavioral patterns and make informed strategic decisions. Similarly, generating best-selling product lists and viewing low inventory reveal crucial information for stock management and replenishment decision-making. By eliminating the need to perform these tasks manually, employees can save time and resources that were previously spent on repetitive, low-value-added activities.

These results evidence the database's ability to effectively handle ordering operations, thereby optimizing store workflow and improving the overall customer experience. Furthermore, by freeing employees from operational and repetitive tasks, automation allows them to focus on more strategic and high-value-added tasks, such as customer service, marketing planning or the development of business growth strategies. This contributes to the store's competitiveness and long-term success in the market.

REFERENCES

[1] C. J. Date, "An Introduction to Database Systems," Addison-Wesley, 8th Edition, 2003. ISBN: 978-0321197849.
[2] A. Silberschatz, H. F. Korth, and S. Sudarshan, "Database System Concepts," McGraw-Hill Education, 7th Edition, 2019. ISBN: 978-1260083502.
[3] T. M. Connolly and C. E. Begg, "Database Systems: A Practical Approach to Design, Implementation, and Management," Pearson, 6th Edition, 2014. ISBN: 978-0132943260.