Celebal Technologies

Celebal Summer Internship Project

Technical Report

E-Commerce Application Database System

Developed By:

Name: Aman Kishore

Email ID:

amankishore05@gmail.com

Acknowledgement

I have immense pleasure in presenting the report for my project entitled “**E-Commerce Application Database”**.

I would like to take this opportunity to express my gratitude to a number of people who have been sources of help & encouragement during the course of this project.

I am very grateful and indebted to my project guide **Mr. Prabal Sharma** sir for providing his enduring patience, guidance & invaluable suggestions. He was the one who never let my morale down & always supported me through my thick & thin. He was the constant source of inspiration for me & took utmost interest in my project.

I would also like to thank all my friends for their invaluable co-operation & providing me the assistance to work for this important project.

Aman Kishore

Email ID: amankishore05@gmail.com

Table of Contents

List of Figures:

Abstract

The E-Commerce Database system serves as a pivotal component within the realm of online commercial activities. With the rapid expansion of digital commerce, this system's purpose is to provide an organized and efficient platform for managing diverse data related to products, customers, orders, and interactions. By seamlessly integrating information and processes, the system contributes to enhancing user experiences, optimizing operations, and enabling informed decision-making.

From product inventory management and order processing to customer engagement and analytics, the system offers an encompassing solution that underpins the dynamic e-commerce landscape. By facilitating secure data storage, real-time tracking, and seamless integration with complementary systems, the E-Commerce Database system empowers businesses to adapt, grow, and excel in the digital marketplace.

In an era characterized by heightened customer expectations and evolving business strategies, the E-Commerce Database system represents more than a technological framework. As businesses harness the power of data-driven insights and personalized experiences, the E-Commerce Database system stands as a testament to the innovation and advancement driving the e-commerce ecosystem forward.

Purpose of the Project

The purpose of an E-Commerce Database system is to provide a robust and efficient platform for managing and organizing the vast amount of data involved in online commercial activities. This system serves as the backbone of an e-commerce business, facilitating various crucial functions to ensure smooth operations, optimal user experiences, and effective decision-making. Some key purposes of an E-Commerce Database system include:

Product Management: The system enables the storage and management of product information, including details such as product descriptions, images, prices, stock levels, and attributes. This ensures accurate and up-to-date information is available to both customers and the business.

Inventory Control: The database system helps track inventory levels in real-time, reducing the likelihood of overselling or running out of stock. This allows businesses to manage their supply chain effectively and maintain customer satisfaction.

Order Processing: The system manages the end-to-end order processing workflow, including order placement, payment processing, order fulfilment, shipping, and order tracking. This streamlines the customer purchasing journey and ensures orders are accurately fulfilled.

Customer Information: The database stores customer profiles, contact details, purchase histories, and preferences. This information enables personalized marketing, targeted promotions, and improved customer support.

Security and Authentication: The system handles user authentication and authorization, ensuring that sensitive customer data, payment information, and account details are securely stored and accessed only by authorized personnel.

Analytics and Reporting: E-Commerce Database systems collect and analyse data on sales, customer behaviour, and website traffic. This information is used to generate insights, identify trends, and make informed business decisions.

Marketing and Personalization: The system supports marketing efforts by allowing businesses to segment their customer base, create targeted campaigns, recommend products based on user behaviour, and offer personalized shopping experiences.

Scalability: As an e-commerce business grows, the database system can scale to accommodate increased data volume, user traffic, and transactions while maintaining performance and responsiveness.

In essence, the E-Commerce Database system plays a critical role in supporting the overall functionality, growth, and success of an online business by effectively managing data and facilitating seamless interactions.

Intended Audiences

The intended audiences for an E-Commerce Database system encompass a range of stakeholders who interact with, utilize, or benefit from the system's functionalities. These audiences play distinct roles in the system's development, operation, and overall success.

Business Owners and Management: Business owners and top-level management utilize the system to gain insights into sales trends, customer behaviour, and overall business performance. They use these insights to make informed decisions about inventory management, marketing strategies, and expansion plans.

Administrators and Managers: Administrators and managers oversee the day-to-day operations of the e-commerce business. They use the system to manage product information, inventory levels, order processing, and customer interactions. They ensure that the system runs smoothly and efficiently.

Marketing and Sales Teams: Marketing and sales teams leverage the system's analytics and customer data to create targeted marketing campaigns, promotions, and product recommendations. They use the data to segment the customer base and tailor their strategies to specific customer segments.

Customer Support: Customer support teams access the system to retrieve customer information, order histories, and communication logs.

Customers: While customers do not directly interact with the database system, they benefit from its functionalities. The system ensures accurate product information, smooth order processing, secure payment handling, and personalized shopping experiences.

Developers and IT Staff: Developers and IT staff are responsible for the system's design, development, implementation, and maintenance. They ensure the system is secure, scalable, and integrates seamlessly with other components of the e-commerce infrastructure.

Data Analysts and Business Intelligence Teams: Data analysts and business intelligence teams use the system's data to generate reports, analyse trends, and provide insights that guide strategic decisions. They extract valuable information from the database to help the business remain competitive.

Third-party Integrators: Professionals responsible for integrating third-party services, such as payment gateways, shipping providers, and marketing tools, interact with the system to establish and maintain these integrations.

The E-Commerce Database system serves as a central hub that connects these diverse audiences, enabling effective collaboration, informed decision-making, and seamless e-commerce operations.

Project Description

The E-Commerce Database System project is an ambitious endeavour aimed at creating a robust and dynamic platform to support the operations of an online retail business. This system will serve as the backbone of the entire e-commerce infrastructure, facilitating the management of products, orders, customers, and crucial data that drive the business forward in the digital marketplace.

Project Goals:

Efficient Product Management: Develop a comprehensive product information management system that enables the seamless addition, modification, and organization of products. This includes support for various product categories, attributes, images, and real-time inventory tracking.

Streamlined Order Processing: Implement an end-to-end order processing workflow that spans from order placement to order fulfilment and delivery. This workflow should integrate with payment gateways for secure and efficient transactions.

Enhanced Customer Experience: Create a user-friendly interface for customers to browse products, place orders, and track their purchases. Personalization features, such as recommended products based on browsing history, should be integrated to improve customer satisfaction.

Advanced Analytics and Reporting: Design a data analytics module that collects and analyses sales data, customer behaviour, and website traffic. This module should generate actionable insights and reports to guide marketing strategies and business decisions.

Seamless Integration: Ensure compatibility with third-party services, including payment gateways, shipping providers, and marketing tools. Integration capabilities should be designed to facilitate smooth data exchange and operations.

Scalability and Security: Build the system to be scalable, accommodating increasing data volumes, user traffic, and transactions as the business grows. Implement stringent security measures to protect sensitive customer data and maintain regulatory compliance.

Key Features of the project:

Product Catalogue: A comprehensive database of products, complete with descriptions, images, prices, and attributes. This module enables easy product management and browsing for both customers and administrators.

Order Management: A centralized order processing system that tracks orders from placement to delivery. This includes order status updates, shipping details, and customer notifications.

Customer Profiles: Secure storage of customer information, order histories, and preferences. The system allows customers to create accounts for personalized shopping experiences.

Analytics Dashboard: An intuitive dashboard that presents key performance indicators, sales trends, and customer behaviour insights. Customizable reports provide valuable data for informed decision-making.

Marketing Integration: Integration with marketing tools to enable targeted campaigns, promotions, and product recommendations based on customer data.

Scalable Architecture: A well-architected system that can scale seamlessly to accommodate growing user bases and increasing data volumes.

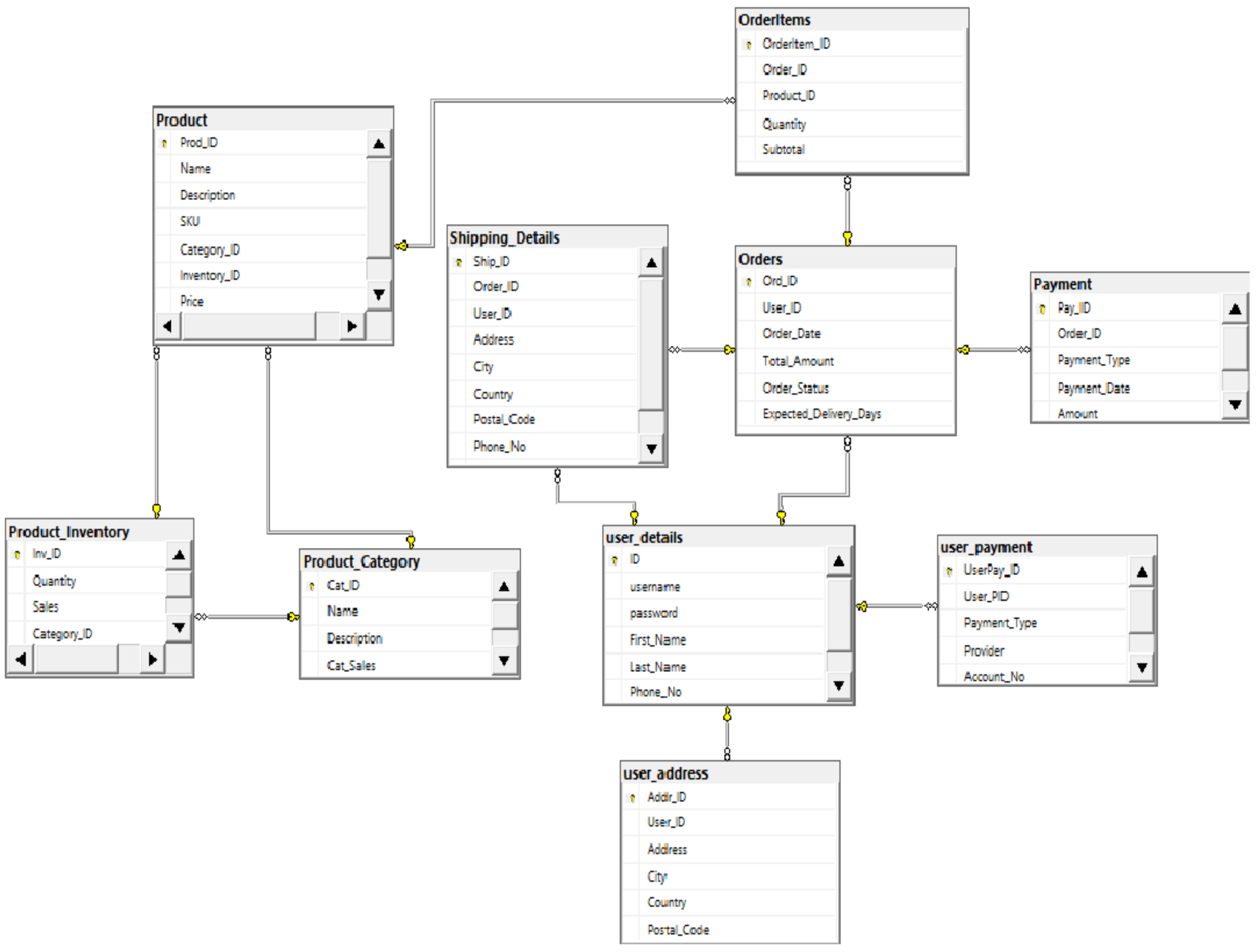
Project Impact:

The successful implementation of the E-Commerce Database System will revolutionize the business's online presence. It will streamline operations, provide valuable insights for strategic planning, enhance customer interactions, and position the business for sustained growth in the competitive e-commerce landscape. By creating a secure, efficient, and user-friendly platform, the project aims to elevate the business's brand, profitability, and customer satisfaction. This project represents a pivotal step toward embracing digital transformation and harnessing the power of data-driven commerce to create a truly exceptional online shopping experience.

Project Insights:

Database Design:

Designing the database for an E-Commerce System involves creating a structured and efficient schema to store, manage, and retrieve data related to products, customers, orders, and other relevant aspects of the e-commerce business. Below is a high-level overview of the database design for an E-Commerce Database System, including the main entities and their relationships:



Entities and Attributes:

User_Details:

- **ID** Primary Key
- **Username**
- **Password**
- **First_Name**
- **Last_Name**
- **Phone No.**

User_Address:

- **Addr_ID** Primary Key
- **User_ID** Foreign Key references User_Details
- **Address**
- **City**
- **Country**
- **Postal Code**

User_Payment:

- **UserPay_ID** Primary Key
- **User_PID** Foreign Key references User_Details
- **Payment Type**
- **Provider**
- **Account No.**

Product_Category:

- **Cat_ID** Primary Key
- **Name**
- **Description**
- **Cat_Sales**

Product_Inventory:

- **Inv_ID** Primary Key
- **Quantity**
- **Sales**
- **Category_ID** Foreign Key references Product_Cat

Product:

- **Prod_ID** Primary Key
- **Name**
- **Description**
- **SKU**
- **Category_ID** Foreign Key references Product_Cat
- **Inventory_ID** Foreign Key references Product_Inv
- **Price**

Payment:

- **Pay_ID** Primary Key
- **Order_ID** Foreign Key references Orders
- **Payment_Type**
- **Date**

Shipping_Details:

Ship_ID	Primary Key
Order_ID	Foreign Key references Orders
User_ID	Foreign Key references User_Details
Address	
City	
Country	
Postal Code	
Phone no.	

Orders:

Ord_ID	Primary Key
User_ID	Foreign Key references User_Details
Order_Date	
Total _Amount	
Order_Status	
Expected_Delivery_Days	

OrderItems:

OrderItem_ID	Primary Key
Order_ID	Foreign Key references Orders
Product_ID	Foreign Key references Product
Quantity	
Sub-Total	

Relationship used in the Tables:

- **One-to-Many Relationship:** One User_Details can have many User_Address entries.
- **One-to-Many Relationship:** One User_Details can have multiple User_Payment entries.
- **One-to-Many Relationship:** One Product_Category can have many Products.
- **One-to-One Relationship:** One Product can have a corresponding entry in Product_Inventory.
- **One-to-Many Relationship**: One Order can have multiple OrderItems.
- **One-to-One Relationship:** One Order can have a corresponding entry in Payments.
- **One-to-One Relationship:** One Order can have a corresponding entry in Shipping_Details.

Database Design Considerations:

- Follow normalization rules to reduce redundancy and ensure data integrity.
- Implement foreign keys and indexes to maintain referential integrity and optimize query performance.
- Apply appropriate data types for each attribute to ensure accurate data storage.
- Implement security measures such as encryption for sensitive data like card numbers and CVV.
- Implement authentication and authorization mechanisms for user access.

- Consider scalability by designing the database to accommodate future growth in data volume and user base.
- Regularly perform maintenance and backups to ensure data availability and reliability.

Views in the database:

Creating views is an important aspect of database management for several reasons, each contributing to better data organization, security, performance, and user experience. Here's why creating views is considered essential in a database:

Data Abstraction and Simplification:

Views allow complex underlying table structures to be abstracted and simplified for users. This simplification makes querying and interacting with the database easier, especially for users who might not be familiar with the intricacies of the database schema.

Data Security and Access Control:

Views provide a layer of security by allowing you to grant different levels of access to different users. You can limit the columns and rows that users can access, ensuring sensitive data remains protected while providing a controlled interface to the data.

Data Integrity and Consistency:

Views ensure that users always see consistent and accurate data, regardless of changes made to the underlying tables. This is particularly important when data in the tables is frequently updated or modified.

Query Optimization and Performance:

Views can be optimized for specific query patterns. You can use indexing, filtering, and other techniques to improve the performance of queries executed against views, enhancing overall database performance.

Business Logic Separation:

Views allow you to encapsulate business logic within the database. This separation ensures that application logic and data manipulation rules are maintained at the database level, reducing redundancy and ensuring consistency.

Customization and Personalization:

Views can be tailored to provide specific data sets for different user groups or applications. This customization improves user experiences by delivering relevant and focused information.

There are 3 Views created in this database:

- **Monthly sales View:** with revenue, number of orders, and top-selling products.
- **Customer loyalty view:** showing the number of repeat customers and their purchase history.
- **Shipping performance view:** analysing average delivery times and tracking delayed orders.

Data Analysis:

Data analysis based on an E-commerce Application database involves extracting insights, patterns, and trends from the collected data. This analysis can provide valuable information for making informed business decisions, improving customer experiences, and optimizing various aspects of the e-commerce operation. Here's how data analysis can be performed on an E-commerce Application database:

Sales Analysis:

- Identify top-selling products and product categories.
- Analyse sales trends over different time periods (daily, weekly, monthly, etc.).
- Determine peak sales times and seasons.
- Explore sales performance across different customer segments.

Customer Behaviour Analysis:

- Analyse customer demographics, such as age, gender, and location.
- Understand purchasing patterns and preferences of different customer groups.
- Identify high-value customers and their buying behaviours.
- Study customer retention rates and reasons for churn.

Product Performance Analysis:

- Assess the popularity of products based on views, cart additions, and purchases.

- Identify products with high cart abandonment rates.
- Analyse customer reviews and ratings for product satisfaction.

Inventory Management:

- Monitor stock levels and identify products at risk of stockouts.
- Optimize inventory turnover rates to minimize overstocking and understocking.
- Analyse slow-moving or non-performing inventory items.

Conversion Rate Optimization:

- Calculate conversion rates at various stages of the sales funnel (e.g., product view to purchase).
- Identify bottlenecks in the conversion process and take measures to improve them.
- Analyse the impact of website design changes on conversion rates.

Marketing and Campaign Analysis:

- Evaluate the effectiveness of marketing campaigns and promotions.
- Track click-through rates and conversion rates from marketing channels.
- Analyse the ROI of different marketing strategies.

Security:

Security is a critical aspect of an E-commerce database system, as it involves protecting sensitive customer data, financial information, and business operations. Here are key security considerations for an E-commerce database:

Data Encryption:

- Encrypt sensitive data such as customer passwords, payment information, and personal details to prevent unauthorized access.
- Use encryption for data transmission (SSL/TLS) to secure data transferred between the user's browser and the server.

User Authentication and Authorization:

- Implement strong user authentication mechanisms (username/password, multi-factor authentication) to ensure only authorized users can access the system.
- Set up role-based access control (RBAC) to determine what actions each user role can perform within the system.

SQL Injection Prevention:

- Use parameterized queries to prevent SQL injection attacks, which involve inserting malicious SQL code into input fields.

Cross-Site Scripting (XSS) Prevention:

- Sanitize user input to prevent cross-site scripting attacks, where malicious scripts are injected into web pages viewed by other users.

Payment Card Industry Data Security Standard (PCI DSS) Compliance:

- Follow PCI DSS requirements for handling payment card data securely. Use secure payment gateways to process transactions.

Implemented Security measures in the Database for user Roles and Permission:

- We create roles using the CREATE ROLE statement.
- We grant different levels of permissions on different tables to the roles using the GRANT statement.
- We use the ALTER ROLE statement to add users to roles.
- We create a stored procedure (Insert Product) that can only be executed by users in the "Manager" role.

Implemented Security measures to restrict access to sensitive data:

- We first create a function **dbo.OrdersRowFilterPredicate** that returns a table with a single row if the provided @UserID matches the current user's ID. This function will be used as the filter predicate.

- Then, we create the security policy OrdersRLS and attach the filter predicate function to the dbo.Orders table. The WITH (STATE = ON) option enables the policy.
- This setup will enforce Row-Level Security on the Orders table, ensuring that users can only access rows where their user ID matches the User_ID column in the table. Please replace User_ID with the actual column name that represents the user ID in your schema.

<u>Transactions and Rollbacks:</u>

Transactions:

- A transaction is a logical unit of work that groups one or more database operations together. It ensures that a series of operations are either completed successfully or completely undone if an error occurs. Transactions are essential for maintaining data integrity and consistency, especially when multiple changes need to be made as a single unit.
- In SQL, transactions are started using the BEGIN TRANSACTION statement and can be ended with either a COMMIT TRANSACTION to apply the changes or a ROLLBACK TRANSACTION to discard the changes.

Rollbacks:

A rollback is an operation that reverses the changes made within a transaction and brings the database back to its state before the transaction began. It's used when an error occurs or when you explicitly decide to discard the changes made within a transaction.

In this database:

- We begin a transaction using BEGIN TRANSACTION.
- We insert a new order into the Orders table.
- We use SCOPE_IDENTITY() to retrieve the ID of the newly inserted order.
- We insert an order item into the OrderItems table, associating it with the new order.
- If all inserts were successful, we commit the transaction using COMMIT TRANSACTION.

If an error occurs after inserting the order but before inserting the order item. In such cases, we can use a rollback to undo the changes made within the transaction

In this database, since we encounter an error and use ROLLBACK TRANSACTION, both the order and order item changes will be undone, and the database will return to its original state.

It's important to use transactions carefully, especially with critical operations, to ensure data consistency and integrity.

Optimization using Indexes:

Indexes in a database are data structures that improve the speed of data retrieval operations by providing a quick way to locate rows in a table. In the context of the provided database, let's discuss the indexes that could be used to optimize query performance for the various tables:

Product Table:

- The primary key index on the ID column is automatically created as a clustered index.
- If you frequently search for products by name or price, consider creating a non-clustered index on columns like Name and Price.

Orders Table:

- The primary key index on the ID column is automatically created as a clustered index.
- If you often search for orders based on User_ID, Order_Date, or Order_Status, consider creating non-clustered indexes on these columns.

Shipping_Details Table:

- The primary key index on the ID column is automatically created as a clustered index.
- If you frequently filter or join based on User_ID or Region, consider creating non-clustered indexes on these columns.

User_Details Table:

- The primary key index on the ID column is automatically created as a clustered index.
- If you often search for users by Username, you can create a unique non-clustered index on the Username column.

User_Payment Table:

- The primary key index on the ID column is automatically created as a clustered index.
- If you frequently filter or join based on User_ID, consider creating a non-clustered index on this column.

OrderItems Table:

- The primary key index on the OrderItem_ID column is automatically created as a clustered index.
- Depending on your query patterns, you might need non-clustered indexes on columns like Order_ID and Product_ID for efficient joins and filters.

It's important to note that while indexes significantly speed up read operations, they can impact the performance of write operations such as inserts, updates, and deletes. Therefore, you should carefully evaluate which indexes to create based on your specific query patterns and usage.

Remember to regularly monitor the performance of your queries and indexes, and be prepared to adjust or reorganize indexes as your database evolves over time.

Data Backup and Recovery Methods:

Data Backup Methods:

Backup methods involve creating copies of your data and storing them in a safe location to prevent data loss. In the context of your database, here are the common backup methods:

- **Full Backup:** A full backup copies the entire database, including all data, objects, and schema. It's a baseline backup that can be used to restore the database to a specific point in time.
- **Differential Backup:** A differential backup captures the changes that have occurred since the last full backup. It's smaller and quicker to create compared to a full backup and is often used in conjunction with full backups.
- **Transaction Log Backup:** If your database is in the Full or Bulk-Logged Recovery Model, you can perform transaction log backups. These backups capture all the transactions that have occurred since the last log backup, allowing you to restore the database to a specific point in time.

Data Recovery Methods:

Recovery methods involve restoring the database from backups in case of data loss or corruption. In the context of your database, here are the common recovery methods:

- **Point-in-Time Recovery:** Using transaction log backups, you can restore the database to a specific point in time before a disaster or error occurred. This is crucial for minimizing data loss.
- **Full Database Restore:** When you need to recover the entire database, you can use a combination of full, differential, and transaction log backups to restore the database to a specific point in time.
- **Page-Level Restore:** In some cases, you might need to restore specific damaged pages rather than the entire database. This is more advanced and requires careful planning.
- **Rollback of Transactions:** In case of a failed transaction or incorrect data change, you can use transaction logs to perform a point-in-time recovery and roll back the problematic transaction.
- **Recovery Testing:** Regularly practice restoring the database from backups to ensure that your backup and recovery processes work as expected. This is crucial to be prepared for any actual data loss scenario.

In this database we have use the Full Backup Methods to recover our data from the accidental affairs.

We had Backed-Up our data in our local system as a matter of the project explanation but in the real scenario it is done in many other systems which are present even outside the country.

Challenges Faced

Designing an e-commerce application database, especially as a beginner, can present several challenges. Here's a list of common problems that I faced during the database design process:

Lack of Domain Knowledge: Understanding the intricacies of e-commerce operations, such as product catalogue management, orders, payments, shipping, and user accounts, can be overwhelming for beginners.

Data Complexity: E-commerce databases involve multiple interconnected entities, relationships, and attributes. Handling product variations, categories, and user data can be complex.

Scalability: Designing for future growth can be challenging. Ensuring the database structure can accommodate an increasing number of products, users, and transactions requires careful planning.

Data Integrity: Maintaining data accuracy and consistency is crucial. Ensuring that data is not duplicated, missing, or conflicting can be difficult, especially as the application scales.

Normalization: Striking the right balance between normalization (reducing redundancy) and denormalization (improving query performance) can be complex and requires understanding the trade-offs.

Performance: E-commerce applications need to handle heavy traffic. Designing the database to provide fast and efficient query responses while minimizing bottlenecks is challenging.

Complex Queries: Writing and optimizing complex SQL queries for fetching products, calculating prices, generating reports, and more can be daunting for beginners.

Security: Ensuring the security of user data, sensitive financial information, and preventing unauthorized access requires implementing proper authentication, authorization, and encryption.

Consistency and Concurrency: Handling multiple users concurrently accessing and updating data without conflicts requires dealing with issues like deadlocks and ensuring data consistency.

Changing Requirements: E-commerce projects often involve evolving business requirements. Adapting the database design to accommodate changes can be challenging, especially for beginners.

Data Migration: Transitioning from a development environment to production or migrating data from an old system to a new one can be complex and risky.

Backup and Recovery: Setting up robust backup and recovery processes to prevent data loss and ensure business continuity is vital but can be complicated for beginners.

Testing and Validation: Ensuring that the database schema, queries, and application interactions are properly tested and validated requires a good understanding of testing methodologies.

Optimization: Fine-tuning database performance, creating appropriate indexes, and optimizing query execution plans might be challenging for beginners.

Documentation: Properly documenting the database design, relationships, constraints, and business rules is crucial for collaboration and future maintenance.

To overcome these challenges, I started with a clear understanding of the application requirements, seek guidance from experienced developers or database administrators, and consider using tools and frameworks that simplify database design and management. Frequent testing, iteration, and continuous learning are essential to growing your skills in database design for e-commerce applications.

Conclusion

The design of an e-commerce application database is a complex endeavour that requires careful consideration, planning, and execution. Through this project, we've explored the various aspects of designing a database to support an e-commerce platform, encompassing product management, order processing, user accounts, payments, and shipping details. Here's a summary of the key takeaways and accomplishments from this project:

Requirements Analysis: We started by understanding the specific requirements of an e-commerce application, including product catalogue management, user registration, order placement, payments, and shipping.

Entity-Relationship Modelling: We translated the requirements into an Entity-Relationship Diagram (ERD), defining the entities, their attributes, and the relationships between them.

Normalization: Applying normalization techniques, we organized the database to minimize data redundancy and improve data integrity.

Table Creation: We created tables for various entities like Product, Orders, User_Details, Shipping_Details, etc., incorporating appropriate data types and constraints.

Indexing: To optimize query performance, we identified key columns and implemented indexes on relevant tables.

Views: We designed views to provide meaningful insights, such as monthly sales, customer loyalty, and shipping performance.

Security Measures: We considered implementing user roles and permissions to restrict access to sensitive data, enhancing data security.

Transaction Management: Demonstrating the use of transactions and rollbacks, we showcased how to ensure data consistency and integrity during complex operations.

Backup and Recovery: We discussed the importance of data backup and recovery strategies to safeguard against data loss and ensure business continuity.

Challenges: Throughout the project, we highlighted common challenges faced by beginners in e-commerce database design, such as data complexity, scalability, and performance optimization.

Conclusion: Successfully designing an e-commerce application database involves a deep understanding of business requirements, database concepts, and real-world challenges. The project showcases the foundational steps and considerations required for building a robust and efficient database to power an e-commerce platform.

References

- References taken from: https://www.google.com/
- Website: https://www.tutorialspoint.com/
- Website: https://www.guru99.com/
- Website: https://www.scaler.com/
- Website: https://www.geeksforgeeks.org/
- Website: https://learn.microsoft.com/
- References taken from Database Management Systems Book: **Database Management Systems (Rajiv Chopra)**