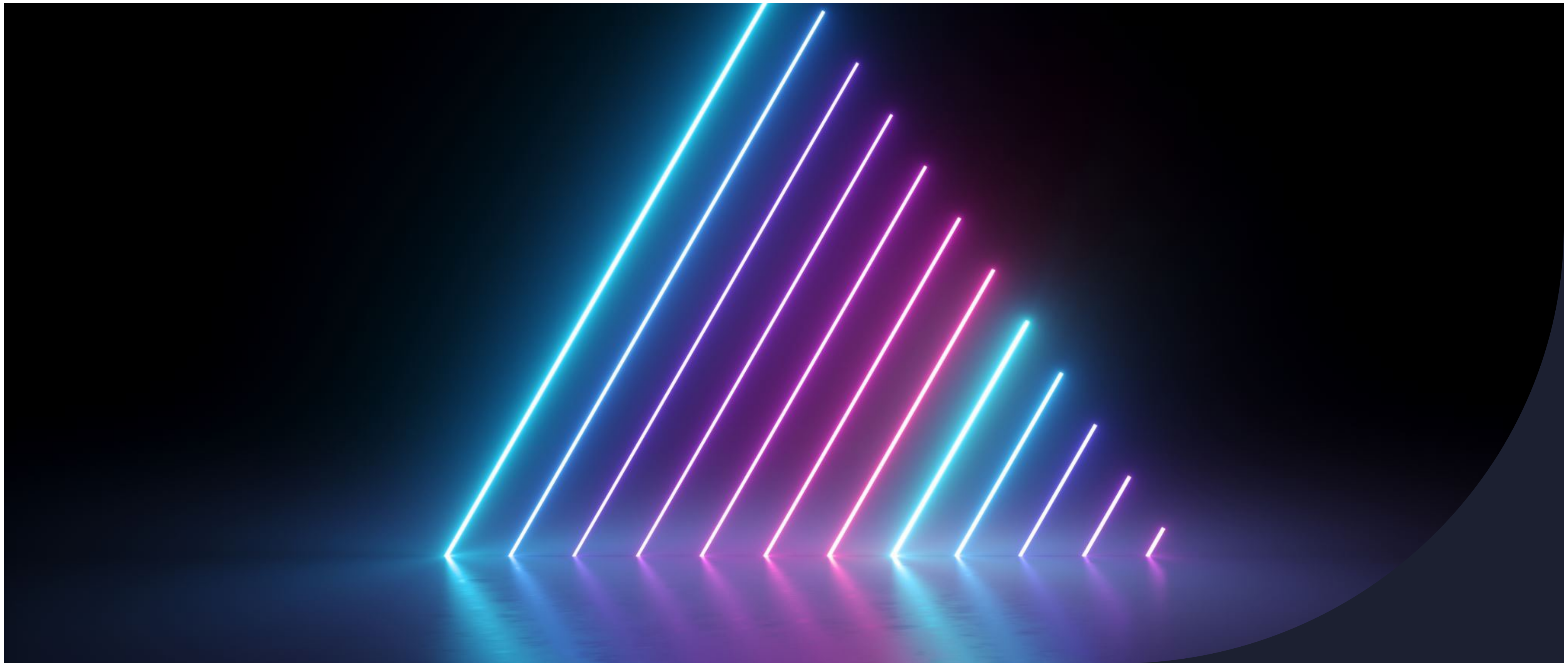

Case Study For Bonga Ecommerce

Offic1alHW (Henry Williams)

Executive Summary

- Bonga eCommerce aims to revolutionize the online retail space by offering a wide range of products to its customers with an emphasis on user experience, efficiency, and reliability. To support its business operations, Bonga eCommerce requires a robust, scalable, and secure database system that can handle vast amounts of data seamlessly.
- This case study explores the **development and implementation of a database solution tailored for Bonga eCommerce, leveraging both serverless and server-based technologies** to manage products, customers, orders, and order items effectively.

Business Problem Statement

Bonga eCommerce faces the challenge of managing an ever-growing inventory of products, a broad customer base, and an increasing volume of orders. The current system struggles with scalability, flexibility in querying data, and efficient data manipulation.

There is a need for a database solution that not only scales with the growing demands but also provides comprehensive data analysis capabilities to support business decisions.

Objectives

- Design and Implement a Scalable Database Schema: Create a database schema that efficiently organizes data for products, customers, orders, and order items.
- Enable Efficient Data Manipulation and Querying: Implement SQL operations that allow for easy insertion, updating, deletion, and querying of data.
- Facilitate Data Analysis and Reporting: Utilize SQL queries to generate reports, perform data analysis, and gain insights into sales trends, customer behavior, and inventory management.
- Explore Serverless and Server-based Database Systems: Compare the use of SQLite for local development and testing with the deployment of a PostgreSQL database for production, understanding the benefits and limitations of each.

Benefits

- Enhanced Data Management: Improved data structure and management facilitate efficient data retrieval, manipulation, and storage.
- Scalability and Flexibility: The chosen tech stack allows for easy scaling of database operations and flexibility in handling different data types and complex queries.
- Data Integrity and Security: Implementing transaction controls and understanding the differences between serverless and server-based environments enhance data integrity and security.
- Insights and Decision Support: Advanced querying capabilities and aggregate functions support comprehensive data analysis, offering valuable insights for business decision-making.

Tech Stack

- Python: For scripting and automating database operations, including generating synthetic data sets.
- SQL: Utilized for data definition, manipulation, and query operations within both SQLite and PostgreSQL databases.
- SQLite: A lightweight, serverless database engine used for local development and testing, providing a simple and efficient way to learn SQL operations.
- PostgreSQL (Pgadmin Server): A powerful, open-source object-relational database system used for production, offering advanced features like transaction controls and complex joins.

Project Scope

Part 1: Serverless Integration with SQLite

- Objective: Gain foundational knowledge in SQL through the creation and manipulation of a simple database schema without the complexity of server setup.
- Scenario: Design a simplified database schema for managing products, customers, orders, and order items.
- Database Schema Design: Includes tables for Products, Customers, Orders, and OrderItems with respective attributes.
- Tasks: Cover DDL, DML, and DQL operations ranging from table creation, data insertion, updates, deletions, to complex queries for data retrieval and analysis.

Project Scope

Part 2: Postgres Server Integration

- Objective: Understand the application of SQL in a server-based database

system, exploring advanced features and differences from SQLite.

- Setup: Install and configure PostgreSQL, recreate the SQLite schema in PostgreSQL, and perform advanced SQL operations.

- Advanced Tasks: Explore PostgreSQL-specific features, complex join operations, and utilize transaction controls for data integrity.

- Deliverables: Include SQL scripts for both SQLite and PostgreSQL, demonstrating key concepts and operations, alongside a comparative analysis of both databases.

Dataset And Database Schema Design:

Products Table:

product_id (Primary Key, Integer) name (Text)
price (Decimal)
category (Text)

Customers Table:

customer_id (Primary Key, Integer) name (Text)
email (Text, unique)

Orders Table:

order_id (Primary Key, Integer) customer_id (Integer, Foreign Key) order_date (Date)

OrderItems Table:

order_item_id (Primary Key, Integer) order_id (Integer, Foreign Key) product_id (Integer, Foreign Key) quantity (Integer)