E-commerce and Delivery Database Design Report

Student: Abishel Mathonsi

Module: PDB470S – Programming and Database Development

Date: 31 October 2025

1. Executive Summary

This project presents the design and implementation of a relational database for a South African e-commerce and delivery platform similar to Takealot or Superbalist. The system integrates vendors, products, orders, payments, deliveries, customers, and inventory management within a scalable, normalized structure. The database ensures data integrity through foreign keys, constraints, and indexing while supporting analytical queries for business decision-making.

2. Problem Analysis

The South African e-commerce ecosystem faces logistical challenges due to multiple seller types, diverse delivery partners, and region-specific payment methods (EFT, card, PayFast, SnapScan). The goal was to design a database capable of:

- Managing multi-vendor marketplace operations.
- Capturing complex product variations and categories.
- Enabling accurate order tracking and fulfilment.
- Integrating multiple delivery methods and service levels.
- Supporting loyalty programs and multi-warehouse inventory management.

The problem required balancing **normalization** for integrity with **performance optimization** for scalability.

3. Solution Design

3.1 Core Entities

- **Customers & Addresses** – manage profiles and delivery locations.
- **Vendors & Vendor Types** – support B2C and marketplace sellers.
- **Products, Categories, Variants** – enable multi-category and size/colour SKUs.
- **Orders & Order Items** – capture sales transactions and link to payments.
- **Payments & Methods** – support card, EFT, COD, mobile payments.
- **Shipments, Delivery Slots, Delivery Partners, Warehouses** – enable tracking and logistics planning.
- **Loyalty Accounts & Ledger** – manage points and redemptions.

3.2 Relationships

- **1-to-many** between vendors → products, customers → orders, orders → items.

- **Many-to-many** between products ↔ categories (resolved via junction tables).
- **1-to-1** between order ↔ payment.
- **Foreign key constraints** maintain referential integrity.
- **CHECK constraints** enforce valid payment status and shipment state.

3.3 Normalization & Optimization

- Up to **3rd Normal Form** for data consistency.
- **Indexes** on foreign keys and search-heavy columns (e.g., email, order_date).
- **View layer** for analytics and reporting.

---

4. Implementation Steps

1. **Schema Design:** Defined 22 interrelated tables with primary and foreign keys.
2. **Database Creation:** Executed `database_schema.sql` in SQLite/SSMS.
3. **Data Population:** Ran `sample_data.sql` to insert ≈50 records across tables.
4. **Query Development:** Implemented `business_queries.sql` for operational insights:
    - Top vendors by revenue
    - Order fulfilment status tracking
    - Customer retention and loyalty points reports
5. **Diagramming:** Generated an ERD (`database_diagram.png`) showing key relations.

---

5. Technical Justifications

- **Relational Database Model**: Ensures data integrity and transactional safety (ACID compliance).
- **Foreign Keys + Constraints**: Prevent orphaned records and invalid states (e.g., orders without customers).
- **Scalability**: Design supports partitioning and indexing for future growth.
- **Query Performance**: Pre-indexed fields optimize JOIN and filter operations.
- **Compliance**: Aligns with POPIA for customer data and BEE vendor classification.

---

6. Testing and Validation

Testing included:

- **Schema Validation:** Ensured all foreign keys resolved successfully.
- **Constraint Testing:** CHECK conditions for payment status and shipment states.
- **Query Accuracy:** Verified aggregations and JOIN logic using test orders.
- **Performance Tests:** Executed query plans to validate index usage.

---

7. Reflection and Improvements

Future enhancements:

- Integrate real-time API for delivery partner tracking.
- Add trigger-based inventory alerts for low stock.
- Extend schema for international currency and tax modules.
- Introduce OLAP cube or Power BI dashboard for managerial analytics.

Appendices

- `database_schema.sql`
- `sample_data.sql`
- `business_queries.sql`
- `database_diagram.png`