PROJECT REPORT

Solving Inventory Inefficiencies using Advanced SQL Analytics

Presented by: Priyanshu Keshav Bawane

IIT Roorkee

priyanshu_kb@ma.iitr.ac.in

Project Title: Inventory Optimizer – SQL-Driven Solution for Urban Retail Co.

1. Project Context & Business Need

Urban Retail Co. is a growing retail chain with a diverse product portfolio spread across multiple physical stores and online channels. As operations scaled, the company faced mounting issues in inventory management. These issues were largely driven by the lack of real-time visibility into stock levels, forecasting limitations, over-reliance on manual processes, and a general underutilization of available data.

Key pain points identified included:

- Frequent stockouts of in-demand items, leading to lost sales and poor customer experience.
- Overstocking of slow-moving goods, locking working capital and inflating storage costs.
- Fragmented visibility across product categories, store locations, and suppliers.
- Absence of data-backed KPIs to drive timely and effective decision-making.

Urban Retail Co. required a robust, SQL-based analytical framework that could transform raw transactional data into intelligent, actionable inventory insights.

---

2. Objectives and Deliverables

The project's goal was to simulate the work of a data analyst in a retail environment by creating:

- A normalized database schema to improve data efficiency and scalability.
- A suite of clean, modular SQL queries to analyze inventory behavior.
- Analytical insights to diagnose inefficiencies and recommend corrective actions.
- KPI metrics and a mock dashboard layout to summarize trends.
- Optional advanced queries to uncover hidden patterns and outliers.

The work was guided by two overarching aims:

- Enhance the accuracy, speed, and automation of inventory decisions.

- Improve supply chain balance between customer demand and operational feasibility.

---

3. Methodology

We adopted a modular, scalable approach in five phases:

a. Data Normalization

The initial CSV dataset was normalized into a star schema with dimension tables for products and stores, and a fact table for sales. This improved the manageability of relationships and allowed efficient querying.

b. Query Development

SQL queries were built and optimized to handle:

- Daily stock level tracking across locations.
- Dynamic reorder point estimation using historical sales trends.
- Low inventory flagging and turnover analysis.
- KPI summaries like average stock levels, forecast errors, and stockout days.

Window functions, indexes, conditional joins, and CTEs were used extensively to ensure performance.

c. Insight Generation

Beyond standard metrics, exploratory analysis revealed key patterns:

- Inventory jumps indicating delayed dispatch logging.
- High forecast error margins for some SKUs, highlighting flaws in prediction logic.
- Minimal stockout days, implying supply chain efficiency — or lack of exception logging.

d. Recommendation Engine

A SQL-driven logic engine was developed to classify SKUs as **"Increase"**, **"Reduce"**, or **"Balanced"**, using thresholds based on average stock, turnover ratio, and low inventory flags.

e. Visualization Design

Although mock dashboards were not built in software, layout concepts were proposed, including cards for low stock counts, bar charts for top-selling SKUs, and forecast error gauges.

---

4. Key Insights

From over 5,000 SKUs across multiple stores, the following insights emerged:

- **Overstock Zones:** Several products had high inventory levels with very low turnover, creating inefficiency in space and capital allocation.

- **Reorder Point Overestimation:** Short-term demand spikes skewed ROP values, leading to inflated restocking signals. This was corrected by using long-term rolling averages with inventory caps.
- **Inventory Lags:** Certain stores showed large sudden stock increases not explained by orders or sales — suggesting a delay in internal updates or missing delivery confirmations.
- **Forecast Inaccuracy:** Some products had consistent gaps between forecasted and actual sales, impacting replenishment decisions. A revised prediction model is necessary.
- **Balanced SKUs:** A majority of SKUs fell in a balanced zone, meaning stock and turnover ratios were in a healthy range — indicating that existing processes worked well for certain categories.

---

5. Recommendations

Based on analysis, the following steps are recommended:

- **Implement Inventory Buffers:** For SKUs with fluctuating demand or delivery lags, buffer levels should be added above the standard ROP.
- **Rebuild Forecast Models:** Forecast accuracy must be improved by incorporating weather, seasonality, competitor pricing, and promotions into the model.
- **Exception Logging:** Systems should log delivery confirmations and internal transfers more rigorously to avoid phantom stock jumps.
- **Supplier Benchmarking:** Incorporate supplier response time and order fill rate into performance analysis to align procurement decisions with reliability.
- **Dynamic ROPs:** Rather than static thresholds, use rolling weekly averages and business rules to auto-update reorder points.
- **Warehouse Strategy:** For overstocked SKUs with low turnover, consider markdowns, store-to-store transfers, or bundle promotions to free space.

---

6. Business Impact

By implementing this SQL-driven inventory intelligence system, Urban Retail Co. can expect:

- **Smarter, faster inventory decisions** backed by data instead of guesswork.
- **Reduced stockouts and lost sales** through early detection and restocking.
- **Lower carrying costs** by minimizing excessive stock of non-performers.
- **Improved supplier coordination** through performance analysis.
- **Enhanced customer satisfaction** due to better product availability.