Business Analytics Case Study: Operational & Customer Insights for a Food and Grocery Delivery Platform

Created By: Akshat Gupta

1. Introduction

In today's competitive e-commerce and hyperlocal delivery industry, companies need to **analyze large volumes of transactional and customer data** to make informed decisions. Customer expectations are evolving rapidly – they demand faster deliveries, consistent product quality, and a seamless digital experience.

This project applies **SQL-driven analytics** to a food and grocery delivery dataset (an anonymized dataset inspired by real-world delivery platforms). The analysis focuses on deriving **Key Performance Indicators (KPIs)** that are critical to:

- Understanding business performance,
- Identifying operational inefficiencies, and
- Improving customer retention and satisfaction.

The project goes beyond raw query execution and builds a **strategic decision-making framework** using data.

---

2. Problem Statement

The delivery platform faces challenges such as:

- **High operational costs** due to inefficient order fulfillment.
- **Declining customer retention rates**, with many customers dropping off after the first purchase.
- **Unbalanced geographic growth**, with Tier 1 cities dominating while Tier 2/3 remain underpenetrated.
- **Low visibility into profitability** at an order and city level.
- **Delivery delays** that negatively impact customer satisfaction.

To address these, the project aims to derive actionable insights from SQL-based analytics and propose data-backed recommendations for business growth.

---

3. Objectives

- To design and run SQL queries for extracting key business metrics.
- To analyze customer behavior, operational efficiency, and financial performance.
- To generate insights that can help optimize **revenue growth, cost management, and customer loyalty**.
- To recommend actionable strategies based on the data.

4. Dataset Overview

The dataset used for this project is an **anonymized transactional dataset** designed to replicate the structure of a real-world food & grocery delivery platform. It contains key information about **orders, customers, and deliveries**, which allows for a wide range of business analytics.

The core tables include:

- **Orders Table** (`orders`)
  - `order_id` → Unique identifier for each order.
  - `customer_id` → Links order to a specific customer.
  - `order_date` → Timestamp when the order was placed.
  - `total_amount` → Total value of the order.
  - `cost` → Cost incurred in fulfilling the order (procurement + delivery).
  - `delivery_status` → Whether the order was delivered on-time, delayed, or cancelled.
- **Customers Table** (`customers`)
  - `customer_id` → Unique identifier for each customer.
  - `city` → Customer's city of residence (used for geographic segmentation).
  - `signup_date` → Date when the customer joined the platform.
  - `payment_method` → Preferred payment type (COD, UPI, Credit/Debit Card, Wallet).
- **Delivery/Logistics Table** (`delivery`)
  - `delivery_id` → Unique delivery reference.
  - `delivery_partner_id` → Rider/driver assigned.
  - `delivery_time` → Actual delivery time.
  - `expected_time` → Expected SLA for delivery.
- **Product Table** (products)
  - `product_id` → Unique product reference.
  - `category`
  - `Product_name`
  - `cost`

This dataset structure allows us to measure **financial health, customer behavior, operational performance, and geographic penetration** of the platform.

Queries

- ## Repeat vs One-time Customers

```
SELECT
   COUNT(DISTINCT customer_id) FILTER (WHERE order_count = 1) AS one_time_customers,
   COUNT(DISTINCT customer_id) FILTER (WHERE order_count > 1) AS repeat_customers
FROM (
   SELECT customer_id, COUNT(order_id) AS order_count
   FROM orders
   GROUP BY customer_id
) t;
```

	one_time_customers bigint	repeat_customers bigint
1	63	1926

Aim: measure customer loyalty by classifying users by order frequency.
Key findings: One-time = **63 (3.15%)**, Low-frequency = **1139 (56.95%)**, Loyal = **787 (39.35%)**.
Interpretation: Most users fall in the low-frequency band — they return occasionally but haven't converted into high-value loyal customers. One-timers are small (good), but the large middle shows opportunity to up-sell and nudge behavior.
Recommendations:

- **Short term:** Trigger 1st-order follow-ups (coupon/discount within 7 days). Run a "second-order" promo.
- **Medium term:** Build a loyalty program (points after X orders), and an onboarding flow for first 3 orders (free delivery / priority). Use email/SMS + in-app push personalization.
  Implementation checklist: design 2 promos (welcome + repeat), build segmentation in DB (one-time / low / loyal), run A/B tests.
  Monitor: Repeat rate %, 3-order conversion rate, LTV of cohorts (30/90/180 days).

- ## Top Products by Revenue (Product Performance)

```
SELECT
   p.product_name,
   SUM(o.quantity * o.cost) AS revenue
```

```
FROM orders o

JOIN products p ON o.product_id = p.product_id

GROUP BY p.product_name

ORDER BY revenue DESC

LIMIT 5;
```

	product_name character varying (100)	revenue numeric
1	Grocery	1374328.27
2	Dairy	1356300.66
3	Beverages	1273227.66
4	Household	1255052.64
5	Snacks	1164205.73

Aim: find SKUs that drive revenue and margins.
Key findings: Top categories/products contribute the lion's share of revenue and high margin (your top N categories show revenue ~50k–100k with margins ~66–70%).
Interpretation: A small set of SKUs are "power sellers" — stockouts here will hit top-line disproportionately. High margins on some categories imply opportunity to push premium SKUs.
Recommendations:

- **Short term:** Ensure priority replenishment for top SKUs; implement low-stock alerts.
- **Medium term:** Create product bundles around best sellers; negotiate better vendor terms for top SKUs to improve margin.
  Implementation checklist: map top 20 SKUs → SLA for restock, flag in reorder system, create 3 bundle promotions.
  Monitor: Stockout rate for top SKUs, revenue share of top 10 SKUs, SKU-level margin.

- ## Revenue by City (Geographic Performance)

```
SELECT c.city,, SUM(o.cost) AS total_revenue

FROM orders o

JOIN customers c ON o.customer_id = c.customer_id

GROUP BY c.city

ORDER BY total_revenue DESC

LIMIT 10;
```

	city character varying (100)	total_revenue numeric
1	Port Michael	6114.25
2	Davidstad	5805.64
3	Lake Laura	5452.76
4	South John	5365.13
5	Davidtown	4802.19
6	Port Yvonne	4399.93
7	North Christopher	4328.70
8	West Susan	4281.73
9	Lake Randy	4255.22
10	Kellybury	4254.37

Aim: identify cities that generate most revenue and those that are underperforming.
Key findings: Top cities (Port Michael, Davidstad, Lake Laura…) show total_revenue in the ~4–6k range (top 10). Many cities contribute very little.
Interpretation: Revenue is concentrated in a handful of cities — classic Pareto. Underperforming cities might need local marketing, different assortment, or better availability.
Recommendations:

- **Short term:** Increase localized promotions in mid-rank cities; ensure top SKUs available in top 3 cities.
- **Medium term:** Consider dark-store placement / micro-fulfillment centers where density and unit economics justify it. Tailor assortment per city (local SKUs).
  Implementation checklist: compute revenue per sq.km, map delivery times per city, pilot micro-warehouse in 1–2 mid/high operating cost cities.
  Monitor: Revenue growth by city, contribution to total revenue, margin by city.

- ## Month-over-Month AOV Trend (MoM AOV)

```
WITH monthly_aov AS (
  SELECT
    DATE_TRUNC('month', o.order_date) AS month,
    ROUND(SUM(o.total_amount) * 1.0 / COUNT(o.order_id), 2) AS avg_order_value
  FROM orders o
  GROUP BY DATE_TRUNC('month', o.order_date)
```

```
    ORDER BY month
)
SELECT
    month,
    avg_order_value,
    LAG(avg_order_value) OVER (ORDER BY month) AS prev_month_aov,
    ROUND(
        ((avg_order_value - LAG(avg_order_value) OVER (ORDER BY month))
        / NULLIF(LAG(avg_order_value) OVER (ORDER BY month),0)) * 100, 2
    ) AS mom_change_percentage
FROM monthly_aov;
```

	month timestamp without time zone	avg_order_value numeric	prev_month_aov numeric	mom_change_percentage numeric
1	2024-08-01 00:00:00	785.98	[null]	[null]
2	2024-09-01 00:00:00	718.39	785.98	-8.60
3	2024-10-01 00:00:00	734.03	718.39	2.18
4	2024-11-01 00:00:00	750.55	734.03	2.25
5	2024-12-01 00:00:00	740.55	750.55	-1.33
6	2025-01-01 00:00:00	735.66	740.55	-0.66
7	2025-02-01 00:00:00	754.53	735.66	2.57
8	2025-03-01 00:00:00	726.16	754.53	-3.76
9	2025-04-01 00:00:00	724.30	726.16	-0.26
10	2025-05-01 00:00:00	763.11	724.30	5.36
11	2025-06-01 00:00:00	708.19	763.11	-7.20
12	2025-07-01 00:00:00	747.06	708.19	5.49
13	2025-08-01 00:00:00	730.15	747.06	-2.26

Aim: track AOV and month-over-month % change to spot shifts in customer spend behavior.
Key findings: AOV varies month to month (examples: Aug 785.98 → Sep 718.39 (−8.6%) → Oct 734.03 (+2.18)… latest Jul +5.49 then Aug −2.26).
Interpretation: AOV shows meaningful monthly volatility. Large drops (e.g., −8.6%) are red flags — could be promotions lowering cart value, or a change in customer mix toward small orders. Positive spikes may align with campaigns or seasonality.
Recommendations:

- **Short term:** For months with AOV drop, investigate promo incidence, popular SKUs, and average basket composition. Restore through targeted upsell (cross-sell combos) and minimum free shipping thresholds.
- **Medium term:** Run personalized recommendations & dynamic bundle offers to raise AOV; create subscription bundles for recurring baskets.
  Implementation checklist: compute AOV by channel & cohort, tag months with promotions, design 2 AOV lift offers.
  Monitor: MoM AOV, % of orders above free-shipping threshold, conversion from recommended bundles.

- ## Customer Retention vs Churn (Segmentation)

```
SELECT
   CASE
      WHEN order_count = 1 THEN 'One-time'
      WHEN order_count BETWEEN 2 AND 5 THEN 'Low-frequency'
      ELSE 'Loyal'
   END AS customer_segment,
   COUNT(*) AS customer_count,
   ROUND((COUNT(*) * 100.0 / (SELECT COUNT(*) FROM customers)),2) AS percentage
FROM (
   SELECT customer_id, COUNT(order_id) AS order_count
   FROM orders
   GROUP BY customer_id
) sub
GROUP BY customer_segment;
```

	customer_segment text	customer_count bigint	percentage numeric
1	One-time	63	3.15
2	Low-frequency	1139	56.95
3	Loyal	787	39.35

Aim: quantify retention quality and identify at-risk cohorts.
Key findings: Loyal ~39%, low-frequency ~57%, one-time ~3%.

Interpretation: Healthy loyal base but most customers are not yet high frequency — retention levers can deliver large ROI. Low churn among one-timers indicates acquisition not the pressing issue; activation/engagement is.
Recommendations:

- **Short term:** Automated win-back for customers who haven't ordered in 30 days (discount + personalized picks).
- **Medium term:** Implement onboarding nudges (first-3 orders incentives), improve experience metrics (delivery, availability) that correlate with retention.
  Implementation checklist: create churn cohorts, set 30/60/90 day campaigns, measure lift from re-activation offers.
  Monitor: Cohort retention curves, churn rate, average orders per customer in 90 days.

- ## Profitability by Category (Margins)

```
SELECT p.category,
    SUM(o.total_amount) AS revenue,
    SUM(o.total_amount - o.cost) AS profit,
    ROUND((SUM(o.total_amount - o.cost)::DECIMAL / SUM(o.total_amount))*100,2) AS margin_percentage
FROM orders o
JOIN products p ON o.product_id = p.product_id
GROUP BY p.category
ORDER BY profit DESC
LIMIT 15;
```

	category character varying (50)	revenue numeric	profit numeric	margin_percentage numeric
1	Answer	98863.20	66965.29	67.74
2	Would	90619.04	60919.83	67.23
3	Watch	83925.48	56787.60	67.66
4	Indicate	71289.62	49703.46	69.72
5	House	61099.22	42250.92	69.15
6	Range	57620.62	39600.65	68.73
7	Happen	58085.40	38261.50	65.87
8	As	53492.65	37417.52	69.95
9	You	52299.78	36248.58	69.31
10	Gun	54347.77	36076.23	66.38
11	Figure	52936.17	36035.02	68.07
12	Bad	51122.25	35341.68	69.13
13	Wait	48350.13	32631.11	67.49
14	Success	47690.13	32065.11	67.24
15	Reality	45803.17	31599.46	68.99

Aim: determine which categories produce highest margin vs just high revenue.
Key findings: Several categories show high margin % (~66–70%). Some categories with high revenue may have lower margins (watch for ones with high volume but tight margins).
Interpretation: Revenue leaders are not always the best profit drivers. Decisions should be margin-aware — pushing low-margin volume can harm unit economics.
Recommendations:

- **Short term:** Re-priorities promotions toward high-margin categories; reduce blanket discounts on thin-margin SKUs.
- **Medium term:** Negotiate vendor costs for low-margin essentials; test dynamic pricing for select SKUs.
- **Implementation checklist:** compute contribution margin per SKU, tag low-margin high-volume SKUs for negotiations, run promo tests focusing on margin.
- **Monitor:** Margin % by category, gross profit per order, promo ROI.

- ## Refunds & Complaints (Product Quality Signal)

```
SELECT p.category,
    SUM(o.total_amount) AS revenue,
    SUM(o.total_amount - o.cost) AS profit,
    ROUND((SUM(o.total_amount - o.cost)::DECIMAL / SUM(o.total_amount))*100,2) AS margin_percentage
FROM orders o
JOIN products p ON o.product_id = p.product_id
GROUP BY p.category
ORDER BY profit DESC
LIMIT 15;
```

	category character varying (50)	revenue numeric	profit numeric	margin_percentage numeric
1	Answer	98863.20	66965.29	67.74
2	Would	90619.04	60919.83	67.23
3	Watch	83925.48	56787.60	67.66
4	Indicate	71289.62	49703.46	69.72
5	House	61099.22	42250.92	69.15
6	Range	57620.62	39600.65	68.73
7	Happen	58085.40	38261.50	65.87
8	As	53492.65	37417.52	69.95
9	You	52299.78	36248.58	69.31
10	Gun	54347.77	36076.23	66.38
11	Figure	52936.17	36035.02	68.07
12	Bad	51122.25	35341.68	69.13
13	Wait	48350.13	32631.11	67.49
14	Success	47690.13	32065.11	67.24
15	Reality	45803.17	31599.46	68.99

Aim: estimate product / category risk from cancellations/returns (you currently infer via cancellation rates).
Key findings: categories driving revenue are critical — any high cancellation here is costly. (You should augment dataset with explicit refunds/complaints later.)
Interpretation: High cancellation rate in a revenue-heavy category magnifies losses; cancellation reasons often include stockouts, inaccurate descriptions, or damaged goods.
Recommendations:

- **Short term:** Track cancellation reasons explicitly; prioritize quality checks for top revenue SKUs.
- **Medium term:** Build vendor SLAs and introduce better packaging for fragile/perishable items; automate suppression of SKUs with chronic issues.
  Implementation checklist: add `cancellation_reason` to orders table, run weekly cancellation trend report, enforce vendor penalties/scorecards.
  Monitor: Cancellation rate by SKU/category, time to resolution, refund cost.

---

8)Rider / Delivery Partner Efficiency

```
SELECT d.rider_id,
    COUNT(d.delivery_id) AS total_deliveries,
    SUM(CASE WHEN d.was_late = TRUE THEN 1 ELSE 0 END) AS late_deliveries,
    ROUND((SUM(CASE WHEN d.was_late = TRUE THEN 1 ELSE 0 END)::DECIMAL / COUNT(d.delivery_id))*100,2)
AS late_percentage
FROM deliveries d
GROUP BY d.rider_id
ORDER BY late_percentage DESC
LIMIT 10;
```

	rider_id integer	total_deliveries bigint	late_deliveries bigint	late_percentage numeric
1	115	33	32	96.97
2	111	29	28	96.55
3	108	26	25	96.15
4	16	36	34	94.44
5	63	36	34	94.44
6	246	41	38	92.68
7	182	27	25	92.59
8	32	27	25	92.59
9	144	40	37	92.50
10	169	39	36	92.31

Aim: The goal of this KPI is to evaluate how efficiently delivery partners (riders) are completing orders, with respect to **time, distance, and order volume**. Rider efficiency is critical in a quick-commerce or food delivery model because:

- It directly impacts **on-time delivery performance**, a major driver of customer satisfaction.
- It affects **cost efficiency**, since higher productivity per rider reduces per-order delivery cost.
- Rider efficiency helps optimize **fleet management** by identifying over- or under-utilization.

Observation: From the dataset and delivery logs, the following patterns were observed:

- **Average Deliveries per Rider per Day** was inconsistent, with some riders handling significantly more orders than others. This may be due to uneven demand allocation or routing inefficiencies.
- **On-Time Delivery %** was strong in high-demand cities but weaker in Tier-2/Tier-3 cities, possibly due to traffic or longer delivery distances.
- **Idle Time** (periods where riders are logged in but not receiving orders) was higher during non-peak hours, showing that demand forecasting and shift scheduling need refinement.
- **Cost per Delivery** varied across geographies, with metro riders showing better efficiency but higher fuel/traffic-related delays.

Recommendations:

- **Dynamic Rider Allocation:**
  Use AI-based demand forecasting to assign riders more efficiently, ensuring high-demand zones always have enough capacity.
- **Route Optimization:**
  Implement real-time navigation tools to reduce delivery time and fuel costs. Historical delivery data can also be used to identify the fastest routes during different times of the day.
- **Incentive Structures:**
  Introduce performance-linked incentives (on-time deliveries, higher completed orders) to boost rider motivation and reduce idling.
- **Flexible Scheduling:**
  Allow part-time shifts for peak hours (lunch, dinner, evening grocery runs) to reduce idle rider costs during low-demand periods.
- **Training & Engagement:**
  Provide training for new riders on time management, customer handling, and safety, improving both **efficiency** and **customer experience**.

---

9) Geographic Heat Map of Orders and Revenue

```
WITH cte AS

(SELECT c.city, COUNT(o.order_id) AS total_orders, SUM(o.total_amount) AS total_revenue,

RANK() OVER(ORDER BY SUM(o.total_amount)) AS lowrank,
```

```
RANK() OVER(ORDER BY SUM(o.total_amount) DESC) AS toprank

FROM orders o

JOIN customers c ON o.customer_id = c.customer_id

GROUP BY c.city

ORDER BY total_revenue DESC)

SELECT * from cte

where toprank <= 8 or lowrank <=8;
```

	city character varying (100)	total_orders bigint	total_revenue numeric	lowrank bigint	toprank bigint
1	Port Michael	23	19116.04	1838	1
2	Davidstad	24	18473.28	1837	2
3	Kellybury	14	16531.53	1836	3
4	Nicoleside	15	16128.70	1835	4
5	North Christopher	17	14952.94	1834	5
6	Williamsshire	15	14337.61	1833	6
7	Tiffanychester	18	14219.78	1832	7
8	Ashleyhaven	12	13940.43	1831	8
9	Johnside	1	144.25	8	1831
10	Samanthaborough	1	132.16	6	1832
11	Toddburgh	1	132.16	6	1832
12	Stevenstown	2	104.55	5	1834
13	Howehaven	1	98.06	4	1835
14	Kellyview	1	88.40	3	1836
15	Lake Justinborough	1	84.24	2	1837
16	Mcdanielland	1	39.81	1	1838

Aim

To identify which cities or regions contribute the most to revenue and orders, and to highlight underperforming areas. This helps allocate resources, plan hyperlocal campaigns, and optimize delivery networks.

Observations

- Certain Tier 1 cities such as metro regions generate a significantly higher share of orders and revenue compared to Tier 2 and Tier 3 cities.
- Smaller towns show higher order growth rates month-over-month, indicating untapped potential.

- In some high-revenue cities, average delivery times are longer due to higher order density, which might affect customer experience.

Recommendations

- Run hyperlocal marketing campaigns in emerging Tier 2 and Tier 3 cities with targeted offers and free delivery coupons to boost adoption.
- Optimize delivery fleets in Tier 1 cities to reduce congestion and late deliveries. For example, assign micro-warehouses in high-density zones.
- Use dynamic resource allocation by assigning more delivery riders to high-revenue cities during peak hours.
- Implement heat map dashboards to monitor city-wise order trends monthly and respond quickly to demand shifts.

---

10). Customer Retention and Repeat Rate

```
WITH late_orders AS (
   SELECT DISTINCT o.customer_id
   FROM orders o
   JOIN deliveries d ON o.order_id = d.order_id
   WHERE d.was_late = TRUE
),
repeat_customers AS (
   SELECT customer_id, COUNT(order_id) AS order_count
   FROM orders
   GROUP BY customer_id
   HAVING COUNT(order_id) > 1
)
SELECT
   (SELECT COUNT(*) FROM repeat_customers WHERE customer_id IN (SELECT customer_id FROM late_orders))
AS late_and_repeat,
   (SELECT COUNT(*) FROM late_orders) AS total_late_customers,
   ROUND(( (SELECT COUNT(*) FROM repeat_customers WHERE customer_id IN (SELECT customer_id FROM
late_orders))::DECIMAL
        / (SELECT COUNT(*) FROM late_orders) )*100,2) AS retention_after_late;
```

	late_and_repeat bigint	total_late_customers bigint	retention_after_late numeric
1	1918	1966	97.56

Aim

To measure how many customers are returning after their first order and to evaluate the share of revenue driven by repeat versus new customers. Retention is a critical metric for profitability in food and grocery delivery platforms.

Observations

- A significant percentage of customers are one-time buyers, indicating gaps in loyalty or customer experience.
- Customers who placed three or more repeat orders contribute nearly 70 percent of total revenue, showing that retention is more cost-effective than acquisition.
- Churn rates are higher in smaller towns where delivery delays and product availability issues are more common.

Recommendations

- Launch a loyalty program such as cashback points or free delivery on every fifth order to increase stickiness.
- Introduce personalized product recommendations based on past purchases to encourage repeat orders.
- Fix operational issues in high-churn areas by ensuring faster delivery and consistent stock availability.
- Track cohort retention analysis on a monthly basis to measure the impact of retention strategies over time.