**Query**

-- 1. Inventory Overview by Store

-- This query can help management quickly see the inventory levels at each store, crucial for ensuring that stock levels are maintained and for planning re-orders.

SELECT

s.store\_id,

s.city,

p.product\_name,

i.quantity\_in\_stock

FROM

inventory i

JOIN stores s ON i.store\_id = s.store\_id

JOIN products p ON i.product\_id = p.product\_id

ORDER BY

s.store\_id,

i.quantity\_in\_stock DESC;

-- 2. Sales Performance by Product and Store

-- This query is useful for identifying top-performing products and stores, which can assist in marketing and sales strategies.

SELECT

s.store\_id,

s.city,

p.product\_name,

COUNT(sale\_id) AS total\_sales,

SUM(t.total\_cost) AS total\_revenue

FROM

sales sa

JOIN transactions t ON sa.transaction\_id = t.transaction\_id

JOIN stores s ON t.store\_id = s.store\_id

JOIN products p ON sa.product\_id = p.product\_id

GROUP BY

s.store\_id,

p.product\_id

ORDER BY

total\_revenue DESC;

-- 3. Detailed Customer Purchases with Promotions

-- This query provides insights into customer behavior, including how promotions affect purchasing patterns. This is vital for understanding the effectiveness of marketing campaigns.

SELECT

c.customer\_id,

c.customer\_name,

t.transaction\_date,

p.product\_name,

t.total\_cost,

pr.promotion\_type

FROM

transactions t

JOIN customers c ON t.customer\_id = c.customer\_id

JOIN sales sa ON t.transaction\_id = sa.transaction\_id

JOIN products p ON sa.product\_id = p.product\_id

LEFT JOIN promotions pr ON t.promo\_id = pr.promo\_id

WHERE

t.discount\_applied = TRUE

ORDER BY

t.transaction\_date DESC;

-- 4. Employee Shifts and Store Allocation

-- This query helps in managing workforce distribution across different locations, crucial for optimizing operational efficiency.

SELECT

e.employee\_id,

e.employee\_name,

e.shift,

s.city,

s.store\_type

FROM

employees e

JOIN stores s ON e.store\_id = s.store\_id

ORDER BY

s.city,

e.shift;

--5.Orders and Vendor Management

-- Understanding supplier relationships and order history can optimize purchasing strategies and ensure timely replenishment of inventory.

SELECT

v.vendor\_name,

o.order\_date,

p.product\_name,

o.quantity\_ordered

FROM

orders o

JOIN vendors v ON o.vendor\_id = v.vendor\_id

JOIN inventory i ON o.inventory\_id = i.inventory\_id

JOIN products p ON i.product\_id = p.product\_id

ORDER BY

o.order\_date DESC;