Project Deliverable 5b

Manmeet Kaur, Babina Basnet, Shubham Patil, Rahul Yatagiri

Report 1 – Low Stock and High-Cost Inventory Analysis Report

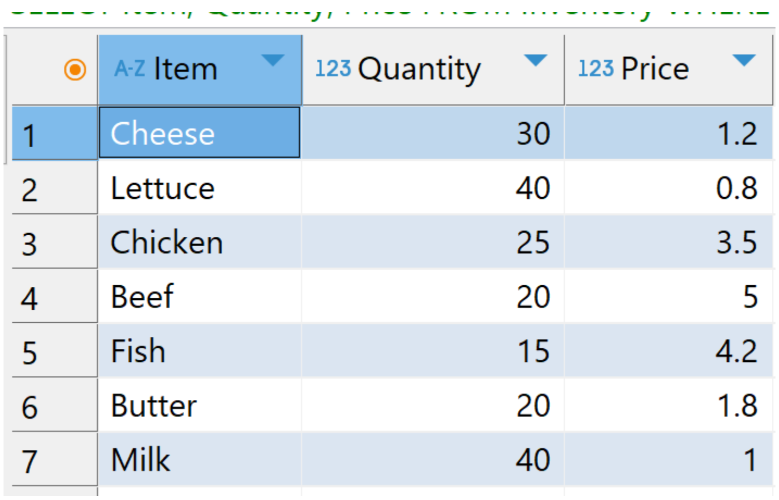
Code-

```
-- view for average quantity and price in inventory

CREATE VIEW Inventory_Averages AS

SELECT

  AVG(Quantity) AS Avg_Quantity,

  AVG(Price) AS Avg_Price

FROM Inventory;


-- Query to identify low-stock items (below average quantity)

SELECT

  Item,

  Quantity,

  Price

FROM Inventory

WHERE Quantity < (SELECT Avg_Quantity FROM Inventory_Averages);


-- Query to identify high-cost items (above average price)

SELECT

  Item,

  Quantity,

  Price

FROM Inventory

WHERE Price > (SELECT Avg_Price FROM Inventory_Averages);


-- Combined query to find items that are either low stock or high cost using UNION

SELECT
```

```
  Item,
  Quantity,
  Price
FROM Inventory
WHERE Quantity < (SELECT Avg_Quantity FROM Inventory_Averages)
UNION
SELECT
  Item,
  Quantity,
  Price
FROM Inventory
WHERE Price > (SELECT Avg_Price FROM Inventory_Averages);

-- Combined query to find items that are both low stock and high cost using INTERSECT
SELECT
  Item,
  Quantity,
  Price
FROM Inventory
WHERE Quantity < (SELECT Avg_Quantity FROM Inventory_Averages)
INTERSECT
SELECT
  Item,
  Quantity,
  Price
FROM Inventory
WHERE Price > (SELECT Avg_Price FROM Inventory_Averages);
```

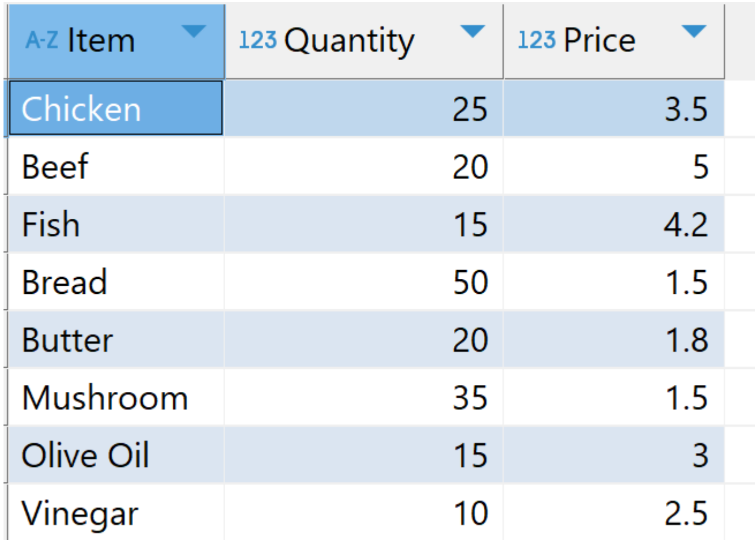
Output-

Query to identify low-stock items (below average quantity)



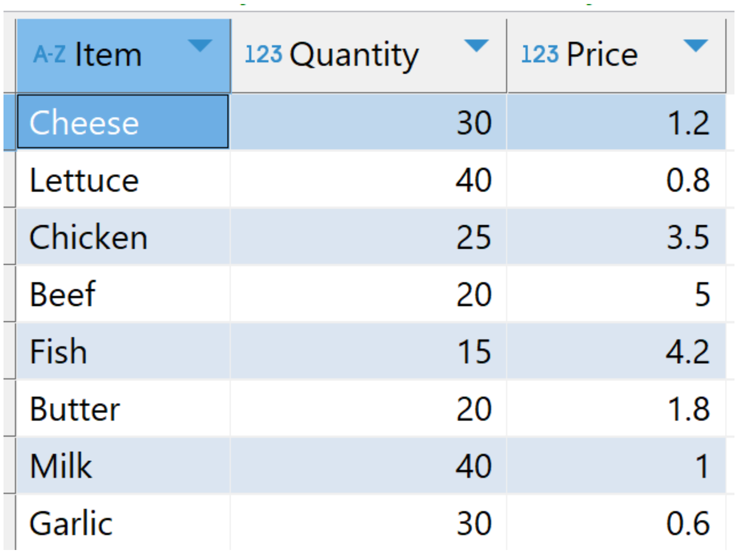
	Item	Quantity	Price
1	Cheese	30	1.2
2	Lettuce	40	0.8
3	Chicken	25	3.5
4	Beef	20	5
5	Fish	15	4.2
6	Butter	20	1.8
7	Milk	40	1

Query to identify high-cost items (above average price)



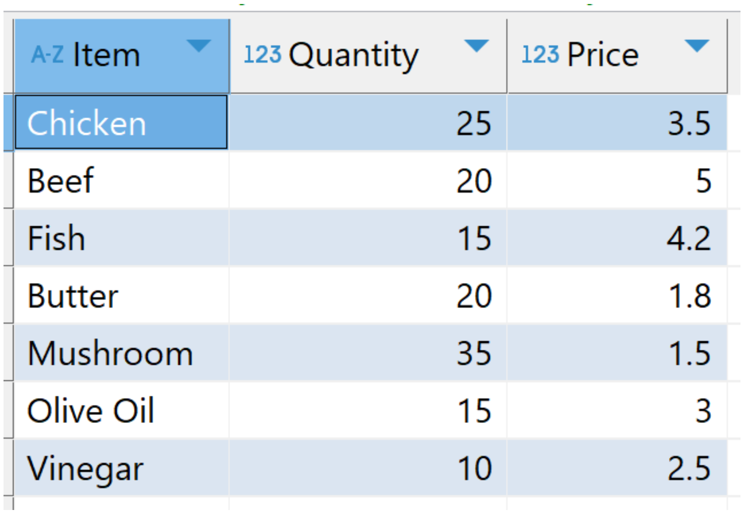
Item	Quantity	Price
Chicken	25	3.5
Beef	20	5
Fish	15	4.2
Bread	50	1.5
Butter	20	1.8
Mushroom	35	1.5
Olive Oil	15	3
Vinegar	10	2.5

Combined query to find items that are either low stock or high cost using UNION



Item	Quantity	Price
Cheese	30	1.2
Lettuce	40	0.8
Chicken	25	3.5
Beef	20	5
Fish	15	4.2
Butter	20	1.8
Milk	40	1
Garlic	30	0.6

Combined query to find items that are both low stock and high cost using INTERSECT



Item	Quantity	Price
Chicken	25	3.5
Beef	20	5
Fish	15	4.2
Butter	20	1.8
Mushroom	35	1.5
Olive Oil	15	3
Vinegar	10	2.5

Business Value-

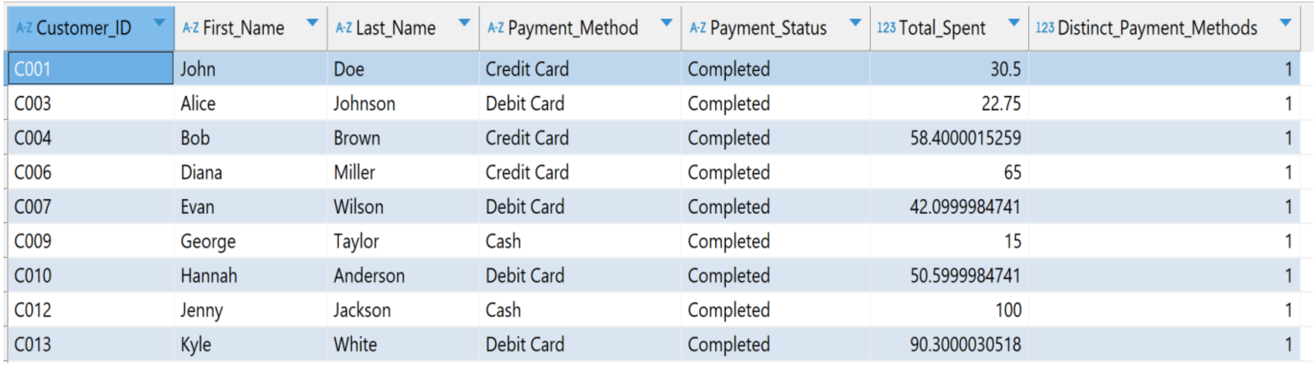
This report helps identify critical inventory items that either require restocking due to low quantities or careful monitoring due to high costs, supporting efficient inventory management and cost control for the restaurant.

Report 2 – Payment Summary Analysis

Code-

```
-- view creation

CREATE VIEW customer_payment_details AS

SELECT Customer_ID, Payment_method, Status, SUM(Amount) AS Total_Spent

FROM Payment

GROUP BY Customer_ID, Payment_method, Status;


-- payment summary query

SELECT c.Customer_ID, c.First_Name, c.Last_Name, p.Payment_Method, p.Status AS Payment_Status,

    p.Total_Spent,

    (SELECT COUNT(DISTINCT Payment_method)

     FROM Payment

     WHERE Customer_ID = c.Customer_ID) AS Distinct_Payment_Methods

FROM Customer c

LEFT JOIN customer_payment_details p ON c.Customer_ID = p.Customer_ID

WHERE p.Status = 'Completed'


UNION


SELECT c.Customer_ID, c.First_Name, c.Last_Name, p.Payment_method, p.Status AS Payment_Status,

    p.Total_Spent,

    NULL AS Distinct_Payment_Methods

FROM Customer c

LEFT JOIN customer_payment_details p ON c.Customer_ID = p.Customer_ID

WHERE p.Status = 'Failed';
```

Output-



Customer_ID	First_Name	Last_Name	Payment_Method	Payment_Status	Total_Spent	Distinct_Payment_Methods
C001	John	Doe	Credit Card	Completed	30.5	1
C003	Alice	Johnson	Debit Card	Completed	22.75	1
C004	Bob	Brown	Credit Card	Completed	58.4000015259	1
C006	Diana	Miller	Credit Card	Completed	65	1
C007	Evan	Wilson	Debit Card	Completed	42.0999984741	1
C009	George	Taylor	Cash	Completed	15	1
C010	Hannah	Anderson	Debit Card	Completed	50.5999984741	1
C012	Jenny	Jackson	Cash	Completed	100	1
C013	Kyle	White	Debit Card	Completed	90.3000030518	1

Business Value-

The report will help the owner to keep track of the financials of the restaurant and keep an eye on payments as these reports generate the detailed summary of payments that consists of Customer ID, First & Last Name, Payment Method and Status.

Report modified with subquery

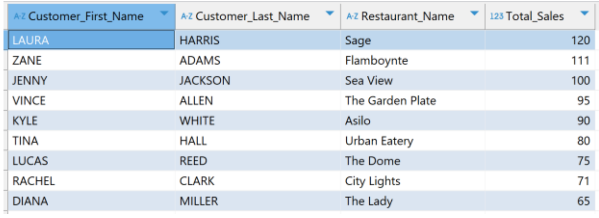
Code without subquery-

```
select
        UPPER(c.First_Name) as Customer_First_Name,
        UPPER(c.Last_Name) as Customer_Last_Name,
        r.Name as Restaurant_Name,
        ROUND(SUM(o.Total_amount), 2) AS Total_Sales
from Orders o
JOIN Customer c on o.Customer_ID = c.Customer_ID
JOIN Restaurant r on o.Restaurant_ID = r.Restaurant_ID
GROUP BY c.First_Name , c.Last_Name , r.Name
ORDER BY Total_amount DESC;
```

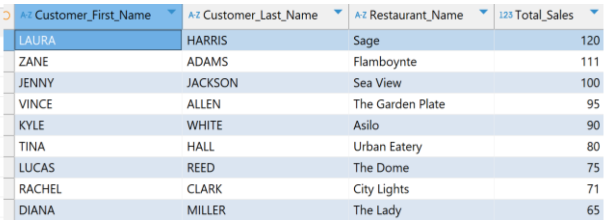
Code with subquery-

```
SELECT
  UPPER(Customer_First_Name) AS Customer_First_Name,
  UPPER(Customer_Last_Name) AS Customer_Last_Name,
  Restaurant_Name, Total_Sales
FROM (
  SELECT
    c.First_Name AS Customer_First_Name,
    c.Last_Name AS Customer_Last_Name,
    r.Name AS Restaurant_Name,
    ROUND(SUM(o.Total_amount), 2) AS Total_Sales
  FROM Orders o
  JOIN Customer c ON o.Customer_ID = c.Customer_ID
  JOIN Restaurant r ON o.Restaurant_ID = r.Restaurant_ID
  GROUP BY c.First_Name, c.Last_Name, r.Name
) AS Subquery
ORDER BY Total_Sales DESC;
```

Output- (Similar for both the codes)



Customer_First_Name	Customer_Last_Name	Restaurant_Name	Total_Sales
LAURA	HARRIS	Sage	120
ZANE	ADAMS	Flamboynte	111
JENNY	JACKSON	Sea View	100
VINCE	ALLEN	The Garden Plate	95
KYLE	WHITE	Asilo	90
TINA	HALL	Urban Eatery	80
LUCAS	REED	The Dome	75
RACHEL	CLARK	City Lights	71
DIANA	MILLER	The Lady	65



Customer_First_Name	Customer_Last_Name	Restaurant_Name	Total_Sales
LAURA	HARRIS	Sage	120
ZANE	ADAMS	Flamboynte	111
JENNY	JACKSON	Sea View	100
VINCE	ALLEN	The Garden Plate	95
KYLE	WHITE	Asilo	90
TINA	HALL	Urban Eatery	80
LUCAS	REED	The Dome	75
RACHEL	CLARK	City Lights	71
DIANA	MILLER	The Lady	65