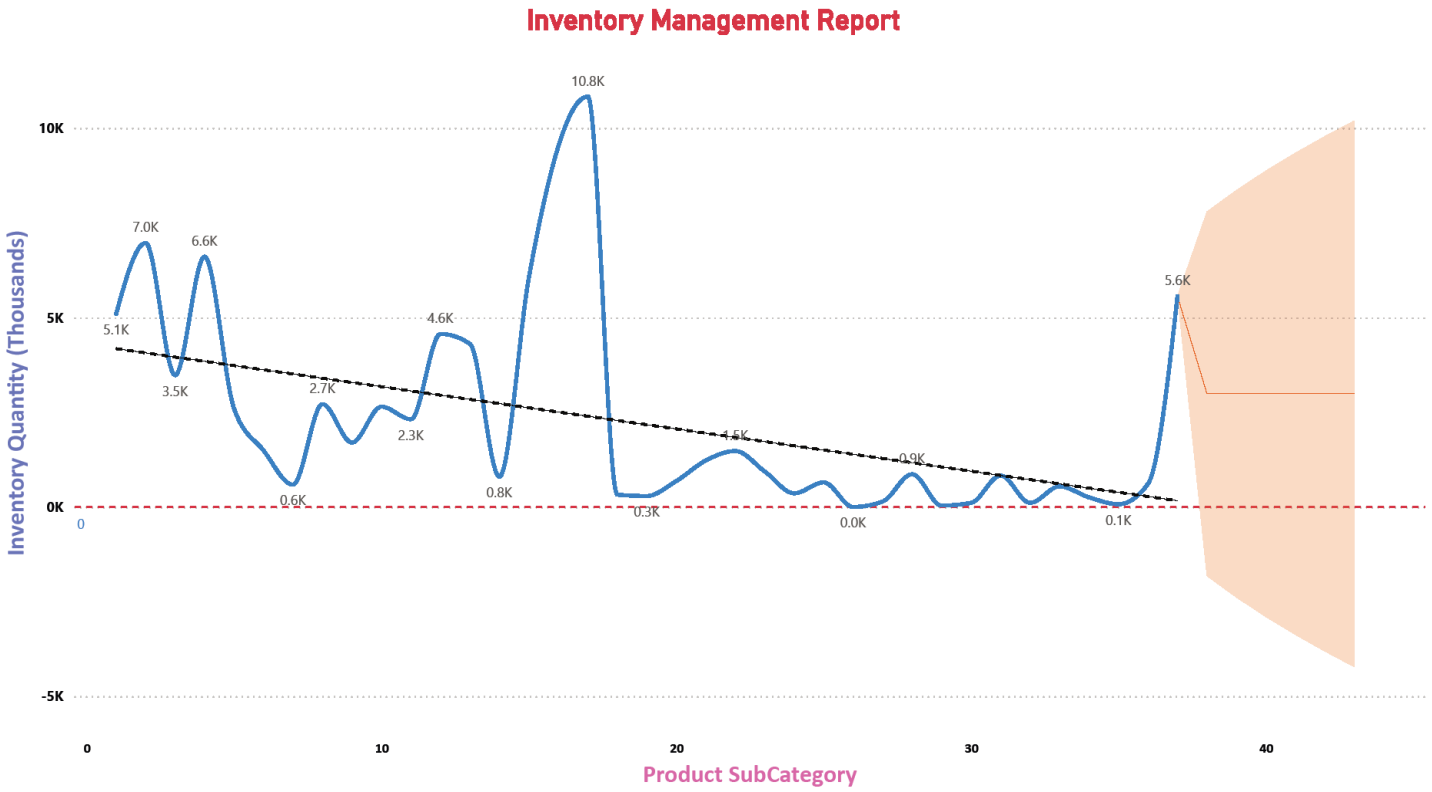
Title: Inventory Management Report

The Inventory Management Report analyzes inventory levels across product subcategories, visualizing trends and forecasting potential stock shortages. This report aids in optimizing stock management and decision-making for AdventureWorks.



Key Insights:

1. The highest inventory level is observed in one product subcategory, peaking at 10.8K units.
2. Overall, inventory levels are decreasing across most subcategories, indicating potential stock depletion in the future.
3. Forecasting suggests a continued downward trend, with some subcategories potentially reaching zero stock in the near future.
4. A small number of subcategories show an upward trend in inventory levels, signaling restocking efforts.

Here’s the Entity Relationship Diagram (ERD) description for the AdventureWorks Inventory Management Report data model:

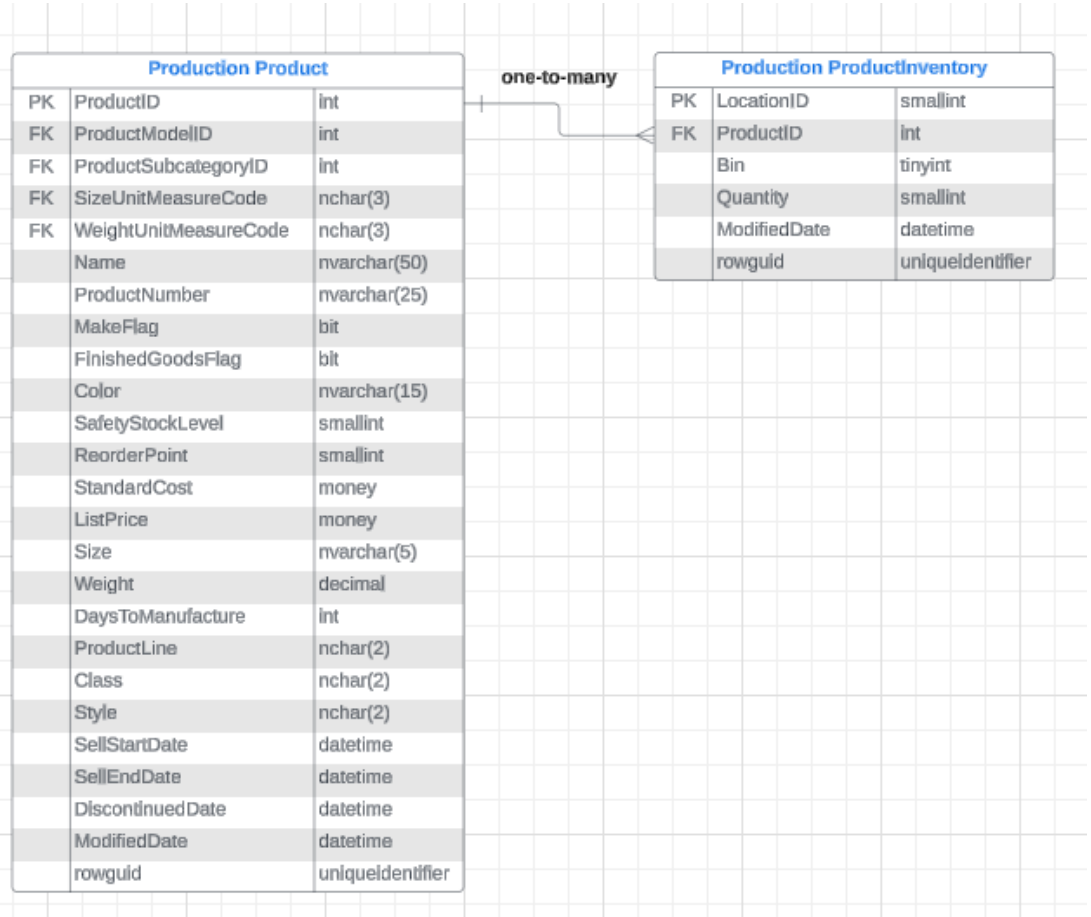
Entity Relationship Diagram Description:

1. Production.Product has a one-to-many relationship with Production.ProductInventory:
    - Each Product can have multiple inventory records across different locations.
    - This means that a single product can be stored in different warehouses or bins, and each of these locations will have its own inventory level (tracked in the ProductInventory table).

    Specifically:

    - The ProductID field in Production.Product is a primary key that uniquely identifies each product.
    - The ProductID in Production.ProductInventory acts as a foreign key, linking each inventory record to its respective product.

This relationship is essential for tracking inventory levels across various storage locations and managing stock efficiently.



SQL Query to validate the results of Power BI Inventory Management Report

```
SELECT v.ProductSubcategoryID, SUM(v.quantity_sum) AS total_quantity
FROM (
   SELECT t.productid, t.quantity_sum, u.ProductSubcategoryID
   FROM (
      -- Summing the quantities for each product in ProductInventory
      SELECT p.productid, SUM(p.quantity) AS quantity_sum
      FROM Production.ProductInventory p
      GROUP BY p.productid
   ) t
   -- Joining the summed quantities with Product to get ProductSubcategoryID
   INNER JOIN Production.Product u
   ON u.ProductID = t.productid
) v
-- Filtering out null ProductSubcategoryID to avoid irrelevant data
WHERE v.ProductSubcategoryID IS NOT NULL
GROUP BY v.ProductSubcategoryID;
```

ProductSubCategoryID	Total Qty
1	5091
2	6968
3	3477
4	6607
5	2600
6	1490
7	589
8	2713
9	1700
10	2646
11	2315
12	4566
13	4305
14	796

15	6052
17	10835
18	324
19	288
20	684
21	1224
22	1476
23	936
24	360
25	648
26	0
27	144
28	864
29	36
30	108
31	828
32	108
33	540
34	252
35	72
36	612
37	5564