# Sale performance

# Sale by countries

### *1. What is the total revenue generated across all transactions?*

In [1]:

```
-- 1. Total Revenue Across All Transactions
SELECT
    COUNT(DISTINCT TransactionNo) as TotalTransactions,
    SUM(Price * Quantity) as TotalRevenue,
    AVG(Price * Quantity) as AvgTransactionValue
FROM [sale_transaction-data];
```

(1 row affected)

Total execution time: 00:00:01.019

Out[1]:

| TotalTransactions | TotalRevenue | AvgTransactionValue |
|---|---|---|
| 19789 | 62965892.34 | 119.306910 |

### *2. Top 10 countries generate the highest total revenue.*

In [3]:

```
--Total transactions by countries
SELECT TOP 10
    Country,
    COUNT(DISTINCT TransactionNo) AS TotalTransactions,
    FORMAT(SUM(Price * Quantity), 'N2') AS TotalRevenue,
    FORMAT(AVG(Price * Quantity), 'N2') AS AvgTransactionValue,
    FORMAT(SUM(Price * Quantity) * 100.0 /
           (SELECT SUM(Price * Quantity) FROM [sale_transaction-data]), 'N2') + '%' AS RevenuePercentage
FROM [sale_transaction-data]
GROUP BY Country
ORDER BY SUM(Price * Quantity) DESC;
```

(10 rows affected)

Total execution time: 00:00:01.502

Out[3]:

| Country | TotalTransactions | TotalRevenue | AvgTransactionValue | RevenuePercentage |
|---|---|---|---|---|
| United Kingdom | 17907 | 52,524,576.47 | 109.94 | 83.42% |
| Netherlands | 94 | 2,151,553.59 | 925.00 | 3.42% |
| EIRE | 277 | 1,713,410.95 | 219.47 | 2.72% |
| Germany | 453 | 1,371,543.27 | 133.94 | 2.18% |
| France | 404 | 1,330,652.89 | 128.03 | 2.11% |
| Australia | 62 | 995,607.91 | 610.43 | 1.58% |
| Sweden | 33 | 401,879.89 | 985.00 | 0.64% |
| Switzerland | 55 | 361,969.25 | 157.17 | 0.57% |
| Japan | 20 | 293,155.44 | 869.90 | 0.47% |
| Spain | 78 | 281,012.27 | 117.78 | 0.45% |

- The UK alone contributes 83.42% of the total revenue, significantly higher than all other countries combined.
- It also has the highest number of transactions (17,907), indicating a strong domestic market.
- The Netherlands (925.00)*andSweden*(985.00) **have much higher average transaction values** than the UK ($109.94).
- This suggests that although they have fewer transactions, customers in these countries purchase in larger amounts per order.
- The Netherlands (3.42%) and EIRE (2.72%) are the next biggest contributors.
- Countries like Japan (0.47%) and Spain (0.45%) contribute the least among the top 10, indicating potential for growth in these regions.

### *3. Top 10 countries generate the highest average transaction value*

In [4]:

```
Select TOP 10 country,
 FORMAT(SUM(Price * Quantity), 'N2') AS TotalRevenue,
    FORMAT(AVG(Price * Quantity), 'N2') AS AvgTransactionValue
from [sale_transaction-data]
group by country
order by  AvgTransactionValue desc
```

(10 rows affected)
Total execution time: 00:00:00.691

Out[4]:

| country | TotalRevenue | AvgTransactionValue |
|---|---|---|
| Sweden | 401,879.89 | 985.00 |
| Portugal | 176,110.40 | 95.82 |
| Malta | 12,717.23 | 94.20 |
| Netherlands | 2,151,553.59 | 925.00 |
| Japan | 293,155.44 | 869.90 |
| Austria | 69,147.26 | 78.22 |
| RSA | 4,259.83 | 74.73 |
| Unspecified | 32,699.05 | 73.15 |
| Australia | 995,607.91 | 610.43 |
| Iceland | 38,321.90 | 48.69 |

### *4. Top 10 customers with highest sale transaction*

In [5]:
```
-- Top 10 customer sale transaction
Select top 10 CustomerNo, FORMAT(SUM(Price * Quantity), 'N2')  as CustomerSale,
FORMAT(AVG(Price * Quantity), 'N2') AS AvgCustomersale
from [sale_transaction-data]
group by  CustomerNo
order by CustomerSale desc
```

(10 rows affected)
Total execution time: 00:00:00.576

Out[5]:

| CustomerNo | CustomerSale | AvgCustomersale |
|---|---|---|
| 173250 | 998.42 | 35.66 |
| 168470 | 997.68 | 83.14 |
| 170080 | 997.64 | 26.25 |
| 156910 | 997.44 | 332.48 |
| 173590 | 996.59 | 66.44 |
| 177180 | 993.47 | 165.58 |
| 164480 | 993.05 | 49.65 |
| 136990 | 992.76 | 70.91 |
| 153350 | 991.99 | 27.56 |
| 138580 | 990.23 | 90.02 |

- All top 10 customers **have similar total sales**, around **$990 - 998**.
- However, the average transaction value **(AvgCustomerSale) varies significantly**, ranging from 26.25 to 332.48.
- **Customer 156910** has a very high average sale value (332.48), suggesting they make fewer but high-value transactions.
- Some customers, like Customer 170080 (Avg: 26.25) and Customer 153350 (Avg: 27.56), likely make many small purchases.
- Others, like Customer 177180 (Avg: 165.58) and Customer 138580 (Avg: 90.02), might buy less frequently but with higher value per purchase.

# Sale by Date

## 1. Yearly Revenue Trend

In [6]:
```
-- Yearly Revenue Trend
SELECT
    YEAR(Date) AS Year,
    COUNT(DISTINCT TransactionNo) AS YearlyTransactions,
    FORMAT(SUM(Quantity * Price), 'N2') AS YearlyRevenue,
    FORMAT(AVG(Quantity * Price), 'N2') AS AvgYearlyTransactionValue,
    -- Calculate Year-over-Year Growth
    FORMAT(LAG(SUM(Quantity * Price)) OVER (ORDER BY YEAR(Date)), 'N2') AS PreviousYearRevenue,
    FORMAT(
        (SUM(Quantity * Price) - LAG(SUM(Quantity * Price)) OVER (ORDER BY YEAR(Date))) /
        NULLIF(LAG(SUM(Quantity * Price)) OVER (ORDER BY YEAR(Date)), 0) * 100, 'N2'
    ) +'%' AS YearOverYearGrowth
FROM [sale_transaction-data]
```

```sql
GROUP BY YEAR(Date)
ORDER BY Year;
```

(2 rows affected)

Total execution time: 00:00:00.730

Out[6]:

| Year | YearlyTransactions | YearlyRevenue | AvgYearlyTransactionValue | PreviousYearRevenue | YearOverYearGrowth |
|---|---|---|---|---|---|
| 2018 | 1552 | 4,415,415.52 | 106.61 | NULL | NULL |
| 2019 | 18237 | 58,550,476.82 | 120.39 | 4,415,415.52 | 1,226.05% |

## 2. Monthy revenue trend

In [7]:

```sql
-- -- Create a stored procedure to analyze monthly revenue trends
-- CREATE PROCEDURE GetMonthlyRevenueTrend
--     @SelectedYear INT = NULL  -- Allow NULL to fetch all years if not specified
-- AS
-- BEGIN
--     SET NOCOUNT ON;

--     SELECT
--         YEAR(Date) AS YearNumber,
--         MONTH(Date) AS MonthNumber,
--         COUNT(DISTINCT TransactionNo) AS MonthlyTransactions,
--         FORMAT(SUM(Quantity * Price), 'N2') AS MonthlyRevenue,
--         FORMAT(AVG(Quantity * Price), 'N2') AS AvgMonthlyTransactionValue,
--         -- Calculate Month-over-Month Growth
--         FORMAT(LAG(SUM(Quantity * Price)) OVER (PARTITION BY YEAR(Date) ORDER BY MONTH(Date)), 'N2') AS Prev
--         FORMAT(
--             (SUM(Quantity * Price) - LAG(SUM(Quantity * Price)) OVER (PARTITION BY YEAR(Date) ORDER BY MONTH
--             NULLIF(LAG(SUM(Quantity * Price)) OVER (PARTITION BY YEAR(Date) ORDER BY MONTH(Date)), 0) * 100,
--         ) + '%' AS MonthOverMonthGrowth
--     FROM [sale_transaction-data]
--     WHERE (@SelectedYear IS NULL OR YEAR(Date) = @SelectedYear)  -- Filter by year if provided
--     GROUP BY YEAR(Date), MONTH(Date)
--     ORDER BY YearNumber, MonthNumber;
-- END;


--DROP PROCEDURE GetMonthlyRevenueTrend;
```

Commands completed successfully.

Total execution time: 00:00:00.012

In [8]:

```sql
-- USE THIS PROCEDURE
--1. To analyze a specific year
EXEC GetMonthlyRevenueTrend @SelectedYear = 2019;
-- 2. To analyze all available years
--EXEC GetMonthlyRevenueTrend @SelectedYear = NULL;
```

Commands completed successfully.

Total execution time: 00:00:00.894

Out[8]:

| YearNumber | MonthNumber | MonthlyTransactions | MonthlyRevenue | AvgMonthlyTransactionValue | PreviousMonthRevenue | MonthOve |
|---|---|---|---|---|---|---|
| 2019 | 1 | 1081 | 4,559,856.37 | 133.58 | NULL | |
| 2019 | 2 | 1096 | 3,335,017.18 | 123.56 | 4,559,856.37 | |
| 2019 | 3 | 1442 | 4,398,401.60 | 123.40 | 3,335,017.18 | |
| 2019 | 4 | 1235 | 3,589,497.88 | 124.05 | 4,398,401.60 | |
| 2019 | 5 | 1670 | 4,578,965.08 | 127.14 | 3,589,497.88 | |
| 2019 | 6 | 1527 | 4,494,648.81 | 125.53 | 4,578,965.08 | |
| 2019 | 7 | 1452 | 4,593,867.06 | 119.64 | 4,494,648.81 | |
| 2019 | 8 | 1341 | 4,758,356.02 | 138.65 | 4,593,867.06 | |
| 2019 | 9 | 1818 | 6,628,303.06 | 135.25 | 4,758,356.02 | |
| 2019 | 10 | 2005 | 7,237,417.36 | 122.58 | 6,628,303.06 | |
| 2019 | 11 | 2753 | 7,861,197.12 | 94.67 | 7,237,417.36 | |
| 2019 | 12 | 817 | 2,514,949.28 | 100.55 | 7,861,197.12 | |

- **Strongest Months for Revenue**
  - **November (7.86*M)andOctober*(7.23M)** had the highest revenue.
  - **September (6.63*M)andAugust*(4.75M)** also performed well.
  - This suggests a strong demand in Q4, possibly due to holiday sales or seasonal promotions.

- **Weakest Months**
  - **December ($2.51M) saw a massive drop (-68.01%)** from November.
  - **February ($3.33M) had the second-lowest revenue** but showed growth in March (+31.89%).
- **Fluctuations & Patterns**
  - **Sharp declines** in March (-18.39%) and June (-1.84%) could indicate post-holiday or mid-year slowdowns.
  - **Significant growth in April (+27.57%) and September (+39.30%)** suggests key marketing efforts or seasonal demand spikes.

## 3. Daily revenue trend

In [12]:
```
-- -- Create Daily procedure
-- CREATE PROCEDURE GetRevenueReport
--     @StartDate DATE = NULL,
--     @EndDate DATE = NULL
-- AS
-- BEGIN
--     SET NOCOUNT ON;

--     -- CTE to get previous revenue before @StartDate
--     WITH PreviousTotal AS (
--         SELECT SUM(Price * Quantity) AS PreviousRevenue
--         FROM [sale_transaction-data]
--         WHERE (@StartDate IS NOT NULL AND Date < @StartDate)
--     )

--     -- Main query to calculate daily transactions, revenue, and running total revenue
--     SELECT
--         Date,
--         COUNT(DISTINCT TransactionNo) AS DailyTransactions,
--         SUM(Price * Quantity) AS DailyRevenue,
--         -- Running total including previous revenue if applicable
--         COALESCE((SELECT PreviousRevenue FROM PreviousTotal), 0) +
--         SUM(SUM(Price * Quantity)) OVER (ORDER BY Date) AS RunningTotalRevenue
--     FROM [sale_transaction-data]
--     WHERE (@StartDate IS NULL OR Date >= @StartDate)
--       AND (@EndDate IS NULL OR Date <= @EndDate)
--     GROUP BY Date
--     ORDER BY Date;
-- END;
```

```
Commands completed successfully.
Total execution time: 00:00:00.012
```

In [13]:
```
-- 1. Get revenue for all dates:
--EXEC GetRevenueReport NULL, NULL;
--2. Get revenue for a specific date range
EXEC GetRevenueReport '2019-01-01', '2019-01-10';
```

```
Commands completed successfully.
Total execution time: 00:00:00.513
```

Out[13]:

| Date | DailyTransactions | DailyRevenue | RunningTotalRevenue |
|---|---|---|---|
| 2019-01-04 | 36 | 102905.96 | 4518321.48 |
| 2019-01-05 | 55 | 225004.01 | 4743325.49 |
| 2019-01-06 | 50 | 267324.81 | 5010650.30 |
| 2019-01-07 | 53 | 189552.39 | 5200202.69 |
| 2019-01-09 | 48 | 96937.98 | 5297140.67 |
| 2019-01-10 | 39 | 155158.65 | 5452299.32 |

This table presents daily sales performance on specific days ( from 2019-01-01 to 2019-01-10)

January 6th saw the highest revenue (267,324.81) in the table

## 4. Highest and lowest sales of specific month

In [16]:
```
DECLARE @SelectedYear INT = 2019;  -- Change this to the desired year
DECLARE @SelectedMonth INT = 6;    -- Change this to the desired month

SELECT
    FORMAT(MAX(Quantity* Price), 'N2') AS HighestSales,
    FORMAT(MIN(Quantity* Price), 'N2') AS LowestSales
FROM [sale_transaction-data]
WHERE YEAR(Date) = @SelectedYear AND MONTH(Date) = @SelectedMonth;
```

```
(1 row affected)
```

```
Total execution time: 00:00:00.293
```

Out[16]:

| HighestSales | LowestSales |
|---|---|
| 16,496.00 | 5.13 |

In June 2019, the highest transaction sale was 16,496, while the lowest transaction sale was 5.13.

## 5. How did sales perform in specific year or month or week ?

In [17]:
```sql
-- CREATE PROCEDURE usp_GetWeeklySalesSummary
--     @Year INT = NULL,  -- Set NULL to include all years
--     @Month INT = NULL, -- Set NULL to include all months
--     @Week INT = NULL   -- Set NULL to include all weeks
-- AS
-- BEGIN
--     SET NOCOUNT ON;

--     SELECT
--         DATENAME(WEEKDAY, Date) AS DayOfWeek,
--         COUNT(DISTINCT TransactionNo) AS TotalTransactions,
--         FORMAT(SUM(Quantity), 'N0') AS TotalQuantity,
--         FORMAT(SUM(Price * Quantity), 'N2') AS TotalRevenue,
--         FORMAT(AVG(Price * Quantity), 'N2') AS AvgTransactionValue,
--         COUNT(DISTINCT Date) AS NumberOfDays,
--         FORMAT(SUM(Price * Quantity) / NULLIF(COUNT(DISTINCT Date), 0), 'N2') AS AvgDailyRevenue
--     FROM [sale_transaction-data]
--     WHERE
--         (@Year IS NULL OR YEAR(Date) = @Year)
--         AND (@Month IS NULL OR MONTH(Date) = @Month)
--         AND (@Week IS NULL OR DATEPART(WEEK, Date) = @Week)
--     GROUP BY DATENAME(WEEKDAY, Date)
--     ORDER BY
--         CASE DATENAME(WEEKDAY, Date)
--             WHEN 'Sunday' THEN 7
--             WHEN 'Monday' THEN 1
--             WHEN 'Tuesday' THEN 2
--             WHEN 'Wednesday' THEN 3
--             WHEN 'Thursday' THEN 4
--             WHEN 'Friday' THEN 5
--             WHEN 'Saturday' THEN 6
--         END;
-- END;
```

```
Commands completed successfully.
Total execution time: 00:00:00.013
```

In [23]:
```sql
-- Get sales summary for June 2019
EXEC usp_GetWeeklySalesSummary @Year = 2019, @Month = 6, @Week = NULL;
```

```
Commands completed successfully.
Total execution time: 00:00:00.491
```

Out[23]:

| DayOfWeek | TotalTransactions | TotalQuantity | TotalRevenue | AvgTransactionValue | NumberOfDays | AvgDailyRevenue |
|---|---|---|---|---|---|---|
| Monday | 196 | 40,020 | 465,869.04 | 122.53 | 4 | 116,467.26 |
| Wednesday | 192 | 41,803 | 474,248.13 | 107.05 | 4 | 118,562.03 |
| Thursday | 233 | 47,129 | 539,542.33 | 95.44 | 4 | 134,885.58 |
| Friday | 244 | 70,001 | 812,713.05 | 137.42 | 4 | 203,178.26 |
| Saturday | 300 | 90,562 | 1,041,114.57 | 136.09 | 5 | 208,222.91 |
| Sunday | 362 | 99,761 | 1,161,161.69 | 138.94 | 5 | 232,232.34 |

**1⃣ Best Sales Day: Sunday**

- Highest Total Revenue: $1,161,163.69
- Highest Total Quantity Sold: 99,761 units
- Highest Average Transaction Value: $138.94
- Highest Avg. Daily Revenue: $232,232.34

→ Sunday is the most profitable day, likely due to high customer traffic or promotional events

**2⃣ Worst Sales Day: Monday**

- Lowest Total Revenue: $465,869.04
- Lowest Total Quantity Sold: 40,020 units
- Low Avg. Transaction Value: $122.53

- Lowest Avg. Daily Revenue: $116,467.26

→ Monday has the weakest performance, suggesting lower customer engagement at the start of the week.

- Friday to Sunday drive the most revenue, with over $800K+ revenue per day.
- Saturday & Sunday outperform weekdays, averaging $200K + dailyrevenue, comparedto$116K–$134K on weekdays.

## Sale by Product

### 1. Top 10 product has highest sales

In [10]:

```sql
SELECT TOP 10
    ProductName,
    SUM(Quantity) AS SumQTY,
    FORMAT(SUM(Quantity * Price), 'N2') AS ProductSale
FROM [sale_transaction-data]
GROUP BY ProductName
ORDER BY SUM(Quantity * Price) DESC;
```

(10 rows affected)
Total execution time: 00:00:01.275

Out[10]:

| ProductName | SumQTY | ProductSale |
|---:|---:|---:|
| Paper Craft Little Birdie | 80995 | 1,002,718.10 |
| Medium Ceramic Top Storage Jar | 78033 | 881,990.18 |
| Popcorn Holder | 56921 | 587,433.94 |
| World War 2 Gliders Asstd Designs | 55047 | 569,735.39 |
| Cream Hanging Heart T-Light Holder | 37956 | 484,592.69 |
| Assorted Colour Bird Ornament | 36493 | 421,318.74 |
| Pack Of 72 Retrospot Cake Cases | 36515 | 391,485.03 |
| Rabbit Night Light | 30788 | 329,029.89 |
| Regency Cakestand 3 Tier | 13890 | 307,483.85 |
| Jumbo Bag Red Retrospot | 48478 | 297,205.04 |

- *Paper Craft Little Birdie* has the highest revenue ($1,002,718.10) and also the highest quantity sold (**80,995 units**).
- *Regency Cakestand 3 Tier* is ranked **#9** in revenue, but it has only **13,890 units sold**, suggesting it has a **high price per unit** compared to others.

### 2. Which products have generated the highest and lowest revenue?

#### Highest revenue

In [11]:

```sql
-- First, calculate revenue for each product
WITH ProductRevenue AS (
    SELECT
        ProductName,
        FORMAT(Price, 'N2') AS Price,
        SUM(Quantity) AS TotalQuantity,
        FORMAT(SUM(Price * Quantity), 'N2') AS TotalRevenue,
        RANK() OVER (ORDER BY SUM(Price * Quantity) DESC) AS RevenueRank
    FROM [sale_transaction-data]
    GROUP BY ProductName, Price
)

-- Then select the product with the highest revenue
SELECT
    ProductName,
    Price,
    TotalQuantity,
    TotalRevenue
FROM ProductRevenue
WHERE RevenueRank = 1;
```

(1 row affected)
Total execution time: 00:00:01.533

Out[11]:

| ProductName | Price | TotalQuantity | TotalRevenue |
|---|---|---|---|
| Paper Craft Little Birdie | 12.38 | 80995 | 1,002,718.10 |

There are 1 product with highest revenue.

## Lowest revenue product

In [12]:

```
-- First, calculate revenue for each product
WITH ProductRevenue AS (
    SELECT
        ProductName,
        FORMAT(Price, 'N2') AS Price,
        SUM(Quantity) AS TotalQuantity,
        FORMAT(SUM(Price * Quantity), 'N2') AS TotalRevenue,
        RANK() OVER (ORDER BY SUM(Price * Quantity) ASC) AS RevenueRank
    FROM [sale_transaction-data]
    GROUP BY ProductName, Price
)

-- Then select the product with the lowest revenue
SELECT
    ProductName,
    Price,
    TotalQuantity,
    TotalRevenue
FROM ProductRevenue
WHERE RevenueRank = 1;
```

```
(9 rows affected)
Total execution time: 00:00:01.512
```

Out[12]:

| ProductName | Price | TotalQuantity | TotalRevenue |
|---|---|---|---|
| Doormat Home Sweet Home Blue | 5.13 | 1 | 5.13 |
| Flower Vine Raffia Food Cover | 5.13 | 1 | 5.13 |
| Lunch Bag Woodland | 5.13 | 1 | 5.13 |
| Set Of 6 Strawberry Chopsticks | 5.13 | 1 | 5.13 |
| Round Storage Tin Vintage Leaf | 5.13 | 1 | 5.13 |
| Jumbo Bag Toys | 5.13 | 1 | 5.13 |
| Lunch Bag Vintage Leaf Design | 5.13 | 1 | 5.13 |
| Lunch Bag Suki Design | 5.13 | 1 | 5.13 |
| Set Of 6 Cake Chopsticks | 5.13 | 1 | 5.13 |

There are 9 products with the same price , same quanity and same revenue

## 3. Product Sales Variance Report

In [19]:

```
-- CREATE PROCEDURE GetProductSalesSummary
--     @Year INT = NULL,        -- Set NULL for all years
--     @Month INT = NULL,       -- Set NULL for all months
--     @ProductName NVARCHAR(255) = NULL -- Set NULL for all products
-- AS
-- BEGIN
--     SET NOCOUNT ON;

--     SELECT
--         ProductName,
--         YEAR(Date) AS YearNumber,
--         MONTH(Date) AS MonthNumber,
--         SUM(Quantity) AS TotalQuantity,
--         SUM(Price * Quantity) AS TotalRevenue,
--         AVG(SUM(Quantity)) OVER (PARTITION BY ProductName) AS AvgMonthlyQuantity,
--         SUM(Quantity) - AVG(SUM(Quantity)) OVER (PARTITION BY ProductName) AS QuantityVarianceFromMean
--     FROM [sale_transaction-data]
--     WHERE
--         (@Year IS NULL OR YEAR(Date) = @Year)
--         AND (@Month IS NULL OR MONTH(Date) = @Month)
--         AND (@ProductName IS NULL OR ProductName = @ProductName)
--     GROUP BY ProductName, YEAR(Date), MONTH(Date)
--     HAVING SUM(Quantity) > 0
--     ORDER BY ProductName, YearNumber;
-- END;

--QuantityVarianceFromMean : Đo lường mức chênh lệch của tổng số lượng sản phẩm bán ra (Quantity) so với trung
```

```
Commands completed successfully.
Total execution time: 00:00:00.003
```

In [22]:
```sql
EXEC GetProductSalesSummary @Year = 2019, @Month = NULL, @ProductName = '10 Colour Spaceboy Pen';
```

```
Commands completed successfully.
Total execution time: 00:00:00.198
```

Out[22]:

| ProductName | YearNumber | MonthNumber | TotalQuantity | TotalRevenue | AvgMonthlyQuantity | QuantityVarianceFromMean |
|---|---|---|---|---|---|---|
| 10 Colour Spaceboy Pen | 2019 | 6 | 691 | 7672.55 | 498.416666 | 193 |
| 10 Colour Spaceboy Pen | 2019 | 3 | 409 | 4548.88 | 498.416666 | -89 |
| 10 Colour Spaceboy Pen | 2019 | 12 | 110 | 695.36 | 498.416666 | -388 |
| 10 Colour Spaceboy Pen | 2019 | 9 | 578 | 6415.84 | 498.416666 | 80 |
| 10 Colour Spaceboy Pen | 2019 | 11 | 805 | 8406.77 | 498.416666 | 307 |
| 10 Colour Spaceboy Pen | 2019 | 8 | 614 | 6791.04 | 498.416666 | 116 |
| 10 Colour Spaceboy Pen | 2019 | 2 | 247 | 2751.44 | 498.416666 | -251 |
| 10 Colour Spaceboy Pen | 2019 | 5 | 636 | 7053.60 | 498.416666 | 138 |
| 10 Colour Spaceboy Pen | 2019 | 10 | 839 | 9328.56 | 498.416666 | 341 |
| 10 Colour Spaceboy Pen | 2019 | 4 | 297 | 3329.89 | 498.416666 | -201 |
| 10 Colour Spaceboy Pen | 2019 | 7 | 471 | 5222.80 | 498.416666 | -27 |
| 10 Colour Spaceboy Pen | 2019 | 1 | 284 | 3164.54 | 498.416666 | -214 |

- **Highest Sales:**
  - October: 839 units → 341 units above average.
  - November: 805 units → 307 units above average.
- **Lowest Sales:**
  - December: 110 units → 388 units below average.
  - February: 247 units → 251 units below average.
- Highest revenue in October $(9,328.56) and November$(8,406.77).
- Lowest revenue in December ($695.36), indicating a sharp decline in sales.