Exploratory data analysis

In this section, I will perform an exploratory analysis of the merged dataset to answer several key business questions. The task is to suggest marketing or operational strategies to increase revenue during low-performing months.

Business questions

- What is the time taken (in days) for each order to be delivered to the customer?
- What are the top 10 most sold products based on the number of items sold?
- What is the total revenue (product price + freight value) generated by each seller?
- What is the total revenue for each month?

Problem: Provide marketing or operational strategies to increase revenue during low-performing months

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import pandas as pd

order_df = pd.read_csv("order_df.csv")
#inspect the dataset
order_df.head(10)
```

Out[1]:

	order_id	customer_id	order_status	order_purchase_timestamp	order_approved_at	order_delivered_carrier_d
0	e481f51cbdc54678b7cc49136f2d6af7	9ef432eb6251297304e76186b10a928d	delivered	2017-10-02 10:56:33	2017-10-02 11:07:15	2017-10-04 19:55
1	53cdb2fc8bc7dce0b6741e2150273451	b0830fb4747a6c6d20dea0b8c802d7ef	delivered	2018-07-24 20:41:37	2018-07-26 03:24:27	2018-07-26 14:31
2	47770eb9100c2d0c44946d9cf07ec65d	41ce2a54c0b03bf3443c3d931a367089	delivered	2018-08-08 08:38:49	2018-08-08 08:55:23	2018-08-08 13:50
3	949d5b44dbf5de918fe9c16f97b45f8a	f88197465ea7920adcdbec7375364d82	delivered	2017-11-18 19:28:06	2017-11-18 19:45:59	2017-11-22 13:39
4	ad21c59c0840e6cb83a9ceb5573f8159	8ab97904e6daea8866dbdbc4fb7aad2c	delivered	2018-02-13 21:18:39	2018-02-13 22:20:29	2018-02-14 19:46
5	a4591c265e18cb1dcee52889e2d8acc3	503740e9ca751ccdda7ba28e9ab8f608	delivered	2017-07-09 21:57:05	2017-07-09 22:10:13	2017-07-11 14:58
6	6514b8ad8028c9f2cc2374ded245783f	9bdf08b4b3b52b5526ff42d37d47f222	delivered	2017-05-16 13:10:30	2017-05-16 13:22:11	2017-05-22 10:07
7	76c6e866289321a7c93b82b54852dc33	f54a9f0e6b351c431402b8461ea51999	delivered	2017-01-23 18:29:09	2017-01-25 02:50:47	2017-01-26 14:16
8	e69bfb5eb88e0ed6a785585b27e16dbf	31ad1d1b63eb9962463f764d4e6e0c9d	delivered	2017-07-29 11:55:02	2017-07-29 12:05:32	2017-08-10 19:45
9	e6ce16cb79ec1d90b1da9085a6118aeb	494dded5b201313c64ed7f100595b95c	delivered	2017-05-16 19:41:10	2017-05-16 19:50:18	2017-05-18 11:40

1. What is the time taken (in days) for each order to be delivered to the customer?

The delivery time to the customer is the duration between when the carrier completes the delivery process and when the customer actually receives the order.

In [2]:
```
# Convert the columns to datetime
order_df['order_delivered_customer_date'] = pd.to_datetime(order_df['order_delivered_customer_date'])
order_df['order_delivered_carrier_date'] = pd.to_datetime(order_df['order_delivered_carrier_date'])

# subtract and get days
delivery_time_days = (order_df['order_delivered_customer_date'] - order_df['order_delivered_carrier_date']).dt.days
print(f"Delivery time (in days):\n{delivery_time_days.head(10)}")
```

```
Delivery time (in days):
0     6
1    12
2     9
3     9
4     1
5    14
6     4
7     6
8     5
9    10
dtype: int64
```

Distribution of delivery times

In [3]:
```
plt.hist(delivery_time_days, bins=50, edgecolor='black')
plt.title('Distribution of delivery times', fontweight='bold')
plt.xlabel('Days', fontweight='bold')
plt.ylabel('Frequency')
plt.show()
```

Average delivery time

In [4]:
```
avg_delivery_time = delivery_time_days.mean()
print(f'The average delivery time is: {avg_delivery_time} days')
```

```
The average delivery time is: 7.346736175455388 days
```

Any noticeable outliers ?

In cleaning process, outliers were addressed by removing negative values representing inconsistent dates (e.g., deliveries before the purchase date) and keeping excessive values (e.g., deliveries exceeding 100 days) in the relevant columns (order_approval, carrier_pickup_time, delivery_time, delivery_accuracy).

The range of delivery times

In [5]:
```
max_days = delivery_time_days.max()
min_days = delivery_time_days.min()
range_days = max_days - min_days
print(f'Range of delivery times: {range_days} days')
```

```
Range of delivery times: 48 days
```

> If 48 days is long:
>
> - "A 48-day delivery range indicates significant variation in delivery times. This could be due to various factors like shipping method, distance, or unforeseen delays."

> If 48 days seems acceptable:
>
> - "A 48-day range might be within acceptable limits depending on the nature of the deliveries and the expectations of the customers."

2. What are the top 10 most sold products based on the number of items sold?

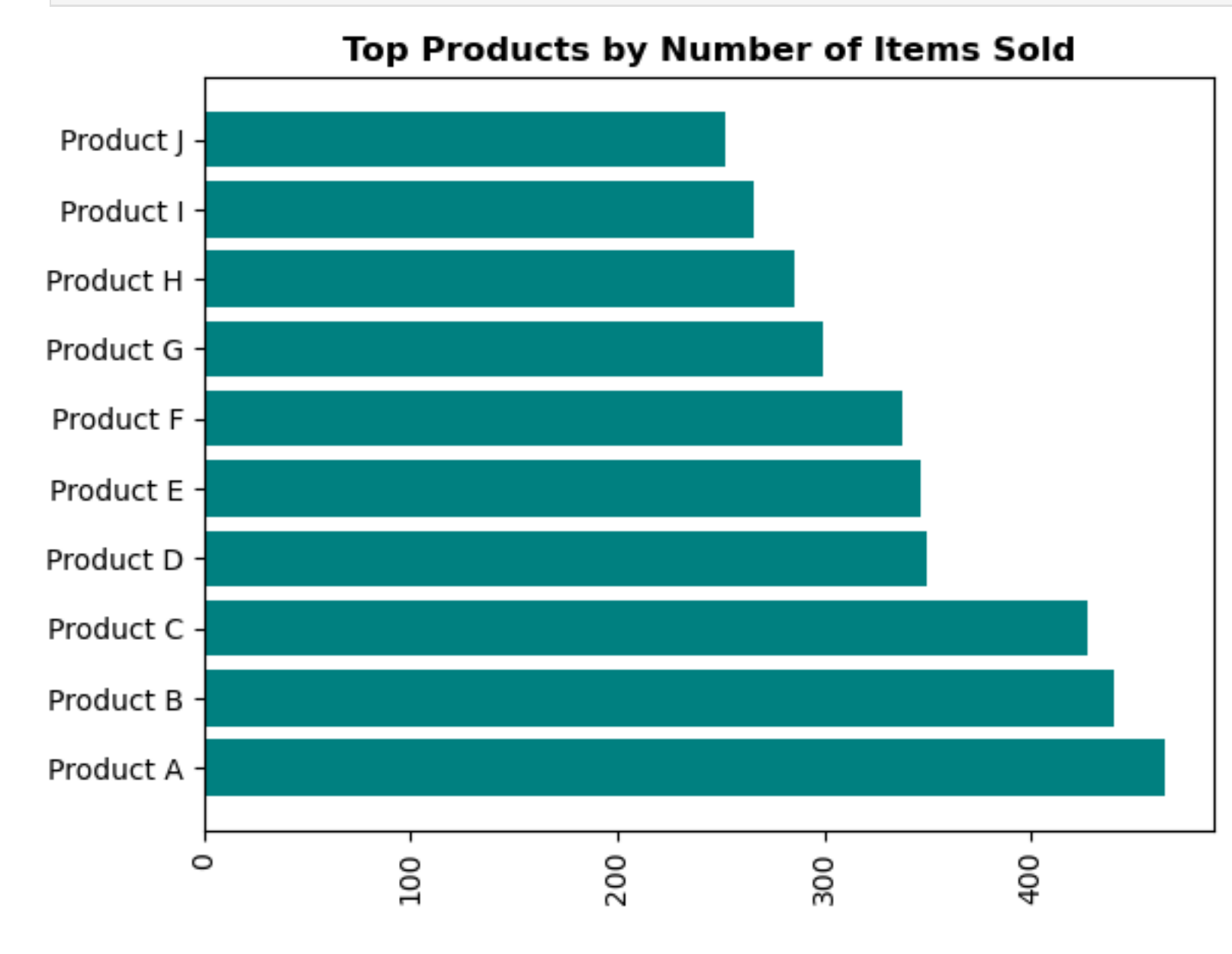
In [6]:
```
top_product = order_df.groupby('product_id', observed=False)['order_item_id'].count().reset_index()
top_ten = top_product.sort_values(by='order_item_id', ascending=False).head(10)

#change top 10 products to readable names
product_name_mapping = {
    "aca2eb7d00ea1a7b8ebd4e68314663af": "Product A",
    "99a4788cb24856965c36a24e339b6058": "Product B",
    "422879e10f46682990de24d770e7f83d": "Product C",
    "368c6c730842d78016ad823897a372db": "Product D",
    "389d119b48cf3043d311335e499d9c6b": "Product E",
    "53759a2ecddad2bb87a079a1f1519f73": "Product F",
    "d1c427060a0f73f6b889a5c7c61f2ac4": "Product G",
    "53b36df67ebb7c41585e8d54d6772e08": "Product H",
    "154e7e31ebfa092203795c972e5804a6": "Product I",
    "3dd2a17168ec895c781a9191c1e95ad7": "Product J"

}
# Replace product_id with custom names
top_ten['product_id'] = top_ten['product_id'].astype(str).replace(product_name_mapping)
top_ten
```

Out[6]:

	product_id	order_item_id
19869	Product A	465
17744	Product B	440
7746	Product C	427
6356	Product D	350
6611	Product E	347
9746	Product F	338
24256	Product G	299
9769	Product H	286
2520	Product I	266
7244	Product J	252

bar chart to display these top-selling products

In [7]:
```
plt.barh(top_ten['product_id'], top_ten['order_item_id'], color='teal')
plt.xticks(rotation=90)
plt.title('Top Products by Number of Items Sold', fontweight='bold')
plt.show()
```



Why certain products might be more popular than others ?

In [8]:
```
top_product_2 = order_df.groupby('product_id', observed=False)['order_item_id'].count().reset_index()
top_product_2['total_price'] = order_df.groupby('product_id', observed=False)['price'].transform('sum')
top_ten = top_product_2.sort_values(by='order_item_id', ascending=False).head(10)
top_ten
```

Out[8]:

	product_id	order_item_id	total_price
19869	aca2eb7d00ea1a7b8ebd4e68314663af	465	1909.00
17744	99a4788cb24856965c36a24e339b6058	440	279.90
7746	422879e10f46682990de24d770e7f83d	427	15.99
6356	368c6c730842d78016ad823897a372db	350	149.94
6611	389d119b48cf3043d311335e499d9c6b	347	636.00
9746	53759a2ecddad2bb87a079a1f1519f73	338	13879.11
24256	d1c427060a0f73f6b889a5c7c61f2ac4	299	33168.80
9769	53b36df67ebb7c41585e8d54d6772e08	286	269.97
2520	154e7e31ebfa092203795c972e5804a6	266	144.40
7244	3dd2a17168ec895c781a9191c1e95ad7	252	20477.48

Some products sell more than others because they are useful, affordable, or in high demand. The top-selling product is bought much more often than others, showing that many customers prefer it. Some items have high sales but low prices, meaning they are budget-friendly and bought frequently. This could be because they are needed often, trusted by buyers, or promoted more. Products that many people use daily or find popular are more likely to be best-sellers.

How this information can be used in marketing strategies ?

This information helps businesses sell more by focusing on popular products. They can keep more stock, offer discounts, or create bundle deals to increase sales. Low-priced products with high sales can be used in promotions to attract customers. For less popular items, they can adjust prices, improve descriptions, or run ads to get more interest. Knowing what customers like helps businesses make better marketing choices.

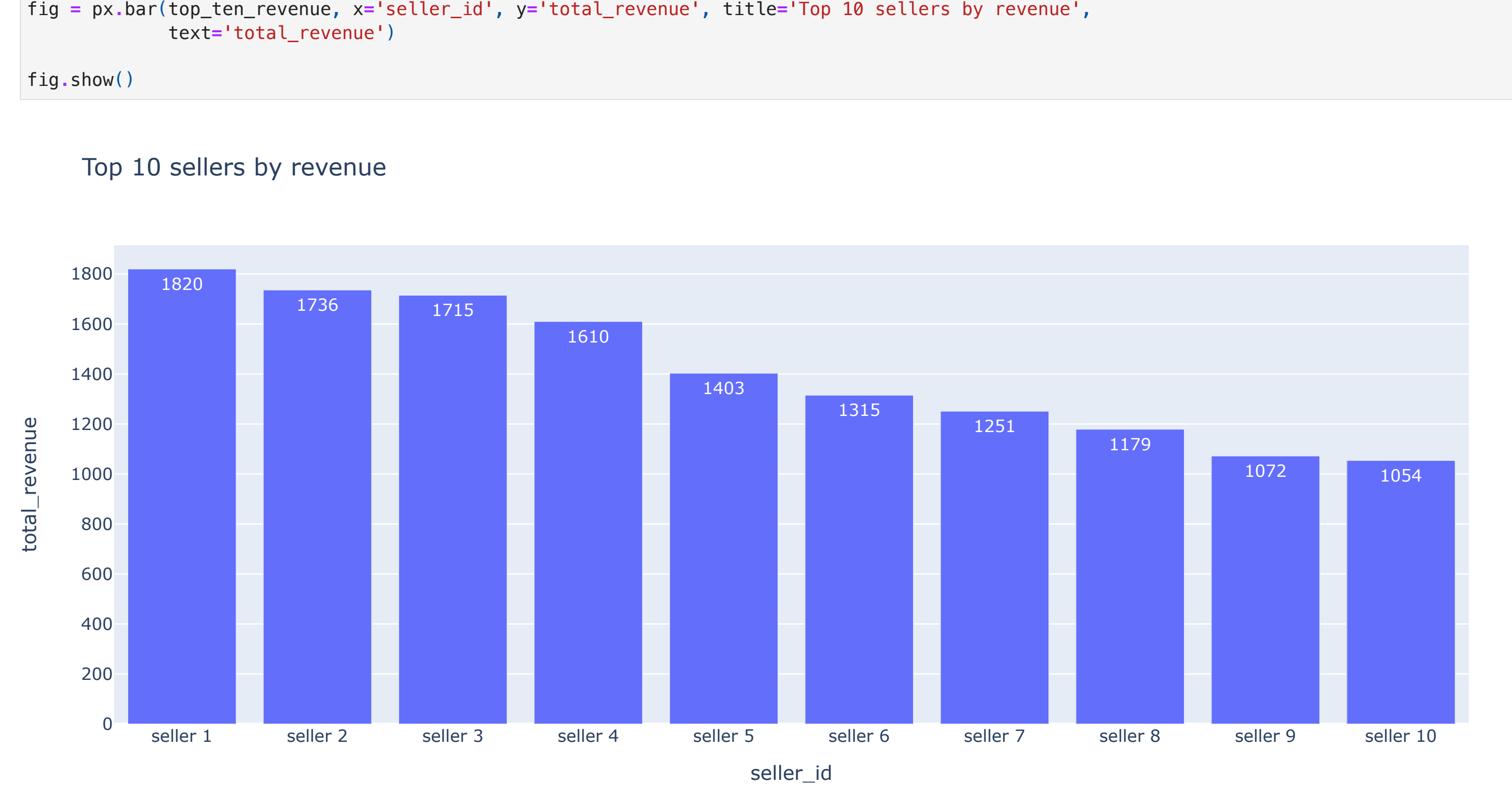
3. What is the total revenue (product price + freight value) generated by each seller?

In [9]:
```
order_df['total_revenue'] = order_df['price'] + order_df['freight_value']
seller_revenue = order_df.groupby('seller_id', observed=False)['total_revenue'].count().reset_index()
top_ten_revenue = seller_revenue.sort_values(by='total_revenue', ascending=False).head(10)
print(f'{top_ten_revenue}')
```

```
                             seller_id  total_revenue
1128  6560211a19b47992c3666cc44a7e94c0           1820
338   1f50f920176fa81dab994f9023523100           1736
813   4a3ca9315b744ce9f8e9374361493884           1715
2271  cc419e0650a3c5ba77189a1882b7556a           1610
2417  da8622b14eb17ae2831f4ac5b9dab84a           1403
1673  955fee9216a65b617aa5c0531780ce60           1315
180   1025f0e2d44d7041d6cf58b6550e0bfa           1251
1406  7c67e1448b00f6e969d365cea6b010ab           1179
1377  7a67c85e85bb2ce8582c35f2203ad736           1072
674   3d871de0142ce09b7081e2b9d1733cb1           1054
```

bar chart visualizing sellers based on total revenue.

In [10]:
```
#change top 10 sellers to readable names
seller_name_mapping = {
    "6560211a19b47992c3666cc44a7e94c0": "seller 1",
    "1f50f920176fa81dab994f9023523100": "seller 2",
    "4a3ca9315b744ce9f8e9374361493884": "seller 3",
    "cc419e0650a3c5ba77189a1882b7556a": "seller 4",
    "da8622b14eb17ae2831f4ac5b9dab84a": "seller 5",
    "955fee9216a65b617aa5c0531780ce60": "seller 6",
    "1025f0e2d44d7041d6cf58b6550e0bfa": "seller 7",
    "7c67e1448b00f6e969d365cea6b010ab": "seller 8",
    "7a67c85e85bb2ce8582c35f2203ad736": "seller 9",
    "3d871de0142ce09b7081e2b9d1733cb1": "seller 10"

}
# Replace seller_id with custom names
top_ten_revenue['seller_id'] = top_ten_revenue['seller_id'].astype(str).replace(seller_name_mapping)

import plotly.express as px
# Create bar chart
fig = px.bar(top_ten_revenue, x='seller_id', y='total_revenue', title='Top 10 sellers by revenue',
             text='total_revenue')

fig.show()
```



Why the top sellers might outperform others ?

- **Competitive Pricing:** Competitive pricing strategies can also contribute to higher sales. Sellers who offer better value for money, discounts, or attractive pricing models can attract more customers.
- **Product Quality and Demand:** Top sellers often offer products that are in high demand or of superior quality. This can lead to higher customer satisfaction and repeat purchases, driving more sales.

What are the potential factors contributing to revenue disparities among sellers ?

Revenue disparities among sellers can be influenced by factors such as product demand, pricing strategies, marketing efforts, and customer trust. Sellers offering high-demand products, competitive pricing, and effective promotions tend to attract more buyers. Additionally, factors like fast shipping, strong brand reputation, and high customer ratings can drive repeat purchases and boost revenue.

4. What is the total revenue for each month?

In [11]:
```
month_revenue = order_df[['order_purchase_timestamp', 'total_revenue']].copy()
month_revenue['order_purchase_timestamp'] = pd.to_datetime(month_revenue['order_purchase_timestamp'])

# I format the datetime to full month name (e.g., "January", "February")
month_revenue['order_purchase_timestamp'] = month_revenue['order_purchase_timestamp'].dt.strftime('%B')

total_month_revenue = month_revenue.groupby('order_purchase_timestamp')['total_revenue'].count().reset_index()

total_month_revenue
```

Out[11]:

	order_purchase_timestamp	total_revenue
0	April	9171
1	August	10763
2	December	5603
3	February	8020
4	January	8269
5	July	10048
6	June	9922
7	March	8955
8	May	10733
9	November	7186
10	October	5164
11	September	4379

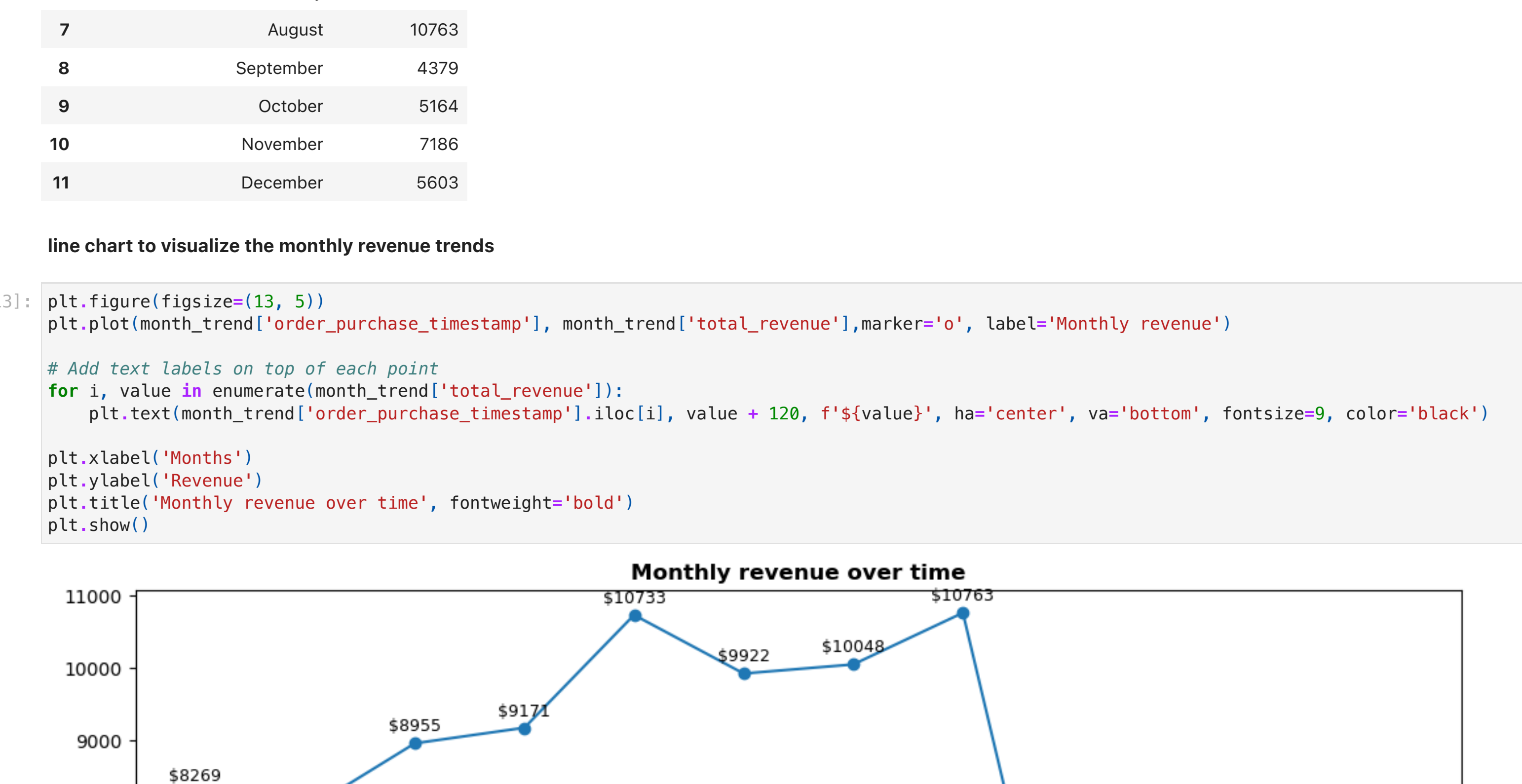
In [12]:
```
sort_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December']
total_month_revenue.index = pd.CategoricalIndex(total_month_revenue['order_purchase_timestamp'], categories=sort_order, ordered=True)
month_trend = total_month_revenue.sort_index().reset_index(drop=True)
month_trend
```

Out[12]:

	order_purchase_timestamp	total_revenue
0	January	8269
1	February	8020
2	March	8955
3	April	9171
4	May	10733
5	June	9922
6	July	10048
7	August	10763
8	September	4379
9	October	5164
10	November	7186
11	December	5603

line chart to visualize the monthly revenue trends

In [13]:
```
plt.figure(figsize=(13, 5))
plt.plot(month_trend['order_purchase_timestamp'], month_trend['total_revenue'],marker='o', label='Monthly revenue')

# Add text labels on top of each point
for i, value in enumerate(month_trend['total_revenue']):
    plt.text(month_trend['order_purchase_timestamp'].iloc[i], value + 120, f'${value}', ha='center', va='bottom', fontsize=9, color='black')

plt.xlabel('Months')
plt.ylabel('Revenue')
plt.title('Monthly revenue over time', fontweight='bold')
plt.show()
```



Any noticeable patterns or trends in the revenue data ?

The revenue fluctuated consistently from January to April, ranging between R8 269 and R9 171. It then remained relatively stable from May to August, hovering between R10 733 and R10 763, with August marking the highest point. A sudden drop occurred in September, reaching the lowest point. Following this, the revenue increased by approximately 17.93% from R4 379 in September to R5 164 in October, and then by 39.16% from October (R5 164) to November (R7 186). However, it decreased again in December, settling at R5 603.

Months with significantly higher or lower revenue, and potential reasons for these variations.

Months with higher or lower revenue could be due to certain trends, sales events, or outside factors. For example, revenue might be higher during holidays, special sales, or new product releases. Lower revenue could happen during quiet months or if there are issues with delivery. By looking closer at the data, we can figure out what might be causing these changes, like marketing efforts, stock problems, or shifts in customer buying patterns.

Marketing or operational strategies to increase revenue during low-performing months

To increase revenue during low-performing months, marketing efforts should be reviewed and adjusted to better target customers during these times. Implementing seasonal promotions or offering discounts could encourage more purchases. Running limited-time offers or flash sales would create a sense of urgency for customers to buy.

Engaging with customers through email campaigns or social media posts can help keep the brand top-of-mind during quieter months. Targeted ads or partnerships with influencers could help reach a wider audience. Additionally, reviewing product stock and ensuring fast, reliable delivery might encourage more customers to make a purchase, especially if they feel confident in the service.

Focusing on building customer loyalty through rewards or loyalty programs can provide incentives for repeat purchases. All of these strategies can help drive sales and improve overall revenue, even when things are slower.

In [ ]: