In [3]:

```
import sqlite3

# Creating a TABLE in Python

connection = sqlite3.connect('sales_data.db')
cursor = connection.cursor()

# SQL command should be in triple quotes

sql_command = '''CREATE TABLE sales (id INTEGER, Product TEXT, Quantity INTEGER, Price R

cursor.execute(sql_command)
connection.close()
```

```
---------------------------------------------------------------------------
OperationalError                          Traceback (most recent call last)
Cell In[3], line 10
      6 # SQL command should be in triple quotes
      8 sql_command = '''CREATE TABLE sales (id INTEGER, Product TEXT, Quantity INTEGER, 
Price REAL)'''
---> 10 cursor.execute(sql_command)
     11 connection.close()

OperationalError: table sales already exists
```

In [21]:

```
# Insert sample data

data = [('Product A', 10, 100),
        ('Product B', 5, 200),
        ('Product A',7, 100),
        ('Product C', 3, 150),
        ('Product B', 4, 200)]

cursor.executemany('INSERT INTO sales (Product, Quantity, Price) values(?,?,?)', data)

connection.commit()
connection.close()
```

```
---------------------------------------------------------------------------
ProgrammingError                          Traceback (most recent call last)
Cell In[21], line 9
      1 # Insert sample data
      3 data = [('Product A', 10, 100),
      4         ('Product B', 5, 200),
      5         ('Product A',7, 100),
      6         ('Product C', 3, 150),
      7         ('Product B', 4, 200)]
----> 9 cursor.executemany('INSERT INTO sales (Product, Quantity, Price) values
(?,?,?)', data)
     11 connection.commit()
     12 connection.close()

ProgrammingError: Cannot operate on a closed database.
```

In [17]:

```
# Read data from sql in python
```

```python
connection = sqlite3.connect('sales_data.db')
cursor = connection.cursor()

# Fetching the data using sql query

cursor.execute('''SELECT * FROM sales;''')

# Fetch all the data using result variable

result = cursor.fetchall()

for i in result:
    print(i)

connection.commit()
connection.close()
```

```
(None, 'Product A', 10, 100.0)
(None, 'Product B', 5, 200.0)
(None, 'Product A', 7, 100.0)
(None, 'Product C', 3, 150.0)
(None, 'Product B', 4, 200.0)
```

In [27]:

```python
print(data)
```

```
[('Product A', 10, 100), ('Product B', 5, 200), ('Product A', 7, 100), ('Product C', 3,
150), ('Product B', 4, 200)]
```

In [33]:

```python
# Total price

connection = sqlite3.connect('sales_data.db')
cursor = connection.cursor()

cursor.execute(''' SELECT sum(Price) as Total_price from sales;''')

Total_price = cursor.fetchall()

print('Total Price:', Total_price)

connection.commit()
connection.close()
```

```
Total Price: [(750.0,)]
```

In [37]:

```python
# Total quantity

connection = sqlite3.connect('sales_data.db')
cursor = connection.cursor()

cursor.execute(''' select Product, sum(Quantity) as Total_quantity from sales group by P
Total = cursor.fetchall()

for i in Total:
    print(i)

# close the connection
```

```
connection.commit()
connection.close()
```

```
('Product A', 17)
('Product B', 9)
('Product C', 3)
```

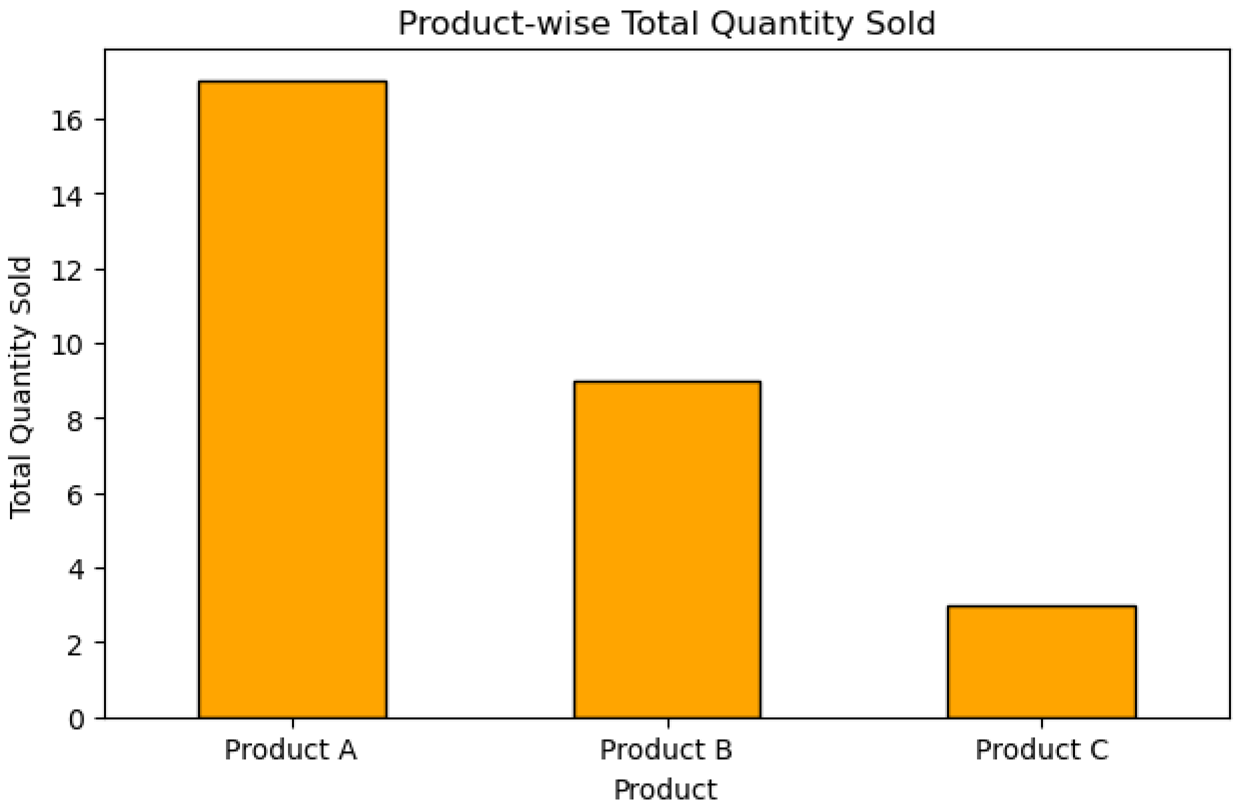
In [47]:

```
import matplotlib.pyplot as plt
import pandas as pd
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[47], line 4
      1 import matplotlib.pyplot as plt
      3 # Create a bar chart
----> 4 data.barplot(kind='bar', x='Product', y='total_quantity', legend=False, color='s
kyblue')
      6 # Add labels and title
      7 plt.xlabel("Product")

AttributeError: 'list' object has no attribute 'barplot'
```

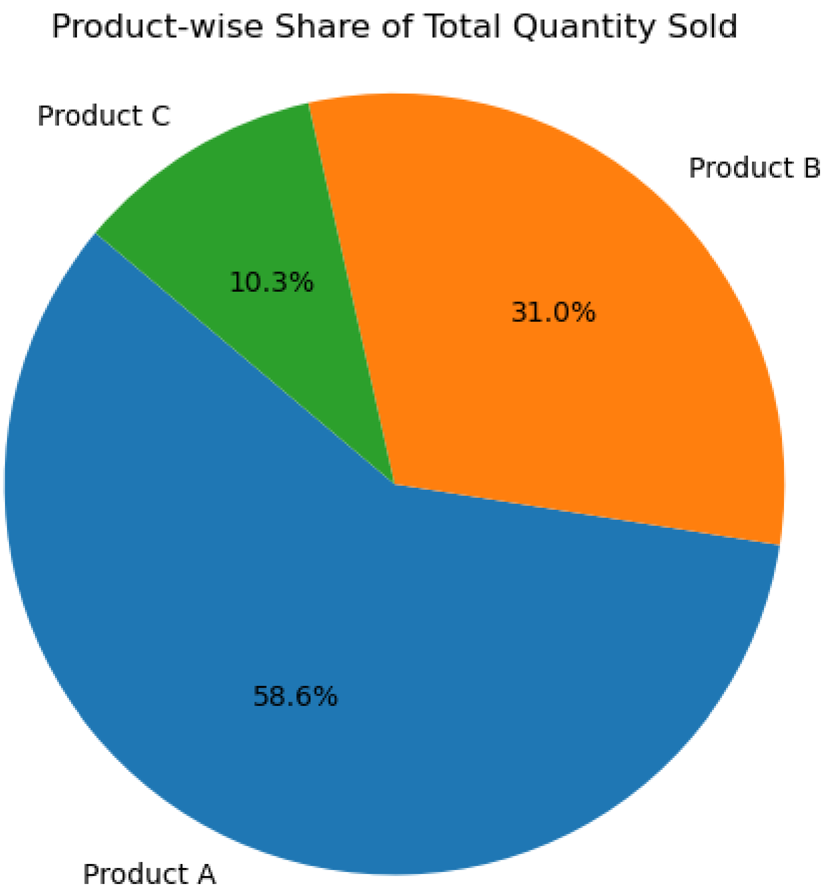
In [103]:

```
# Connect to the database
connection = sqlite3.connect('sales_data.db')

# Query to get product-wise total quantity
query = ('''
SELECT Product, SUM(Quantity) AS total_quantity
FROM sales
GROUP BY Product
''')

# Read query into DataFrame
df = pd.read_sql_query(query, connection)

# Close the connection
connection.close()

# Plotting the bar chart
df.plot(kind='bar', x='Product', y='total_quantity', legend=False, color='orange', edgec

# Add labels and title
plt.xlabel('Product')
plt.ylabel('Total Quantity Sold')
plt.title('Product-wise Total Quantity Sold')
plt.tight_layout()
plt.xticks(rotation=0)

plt.show()
```



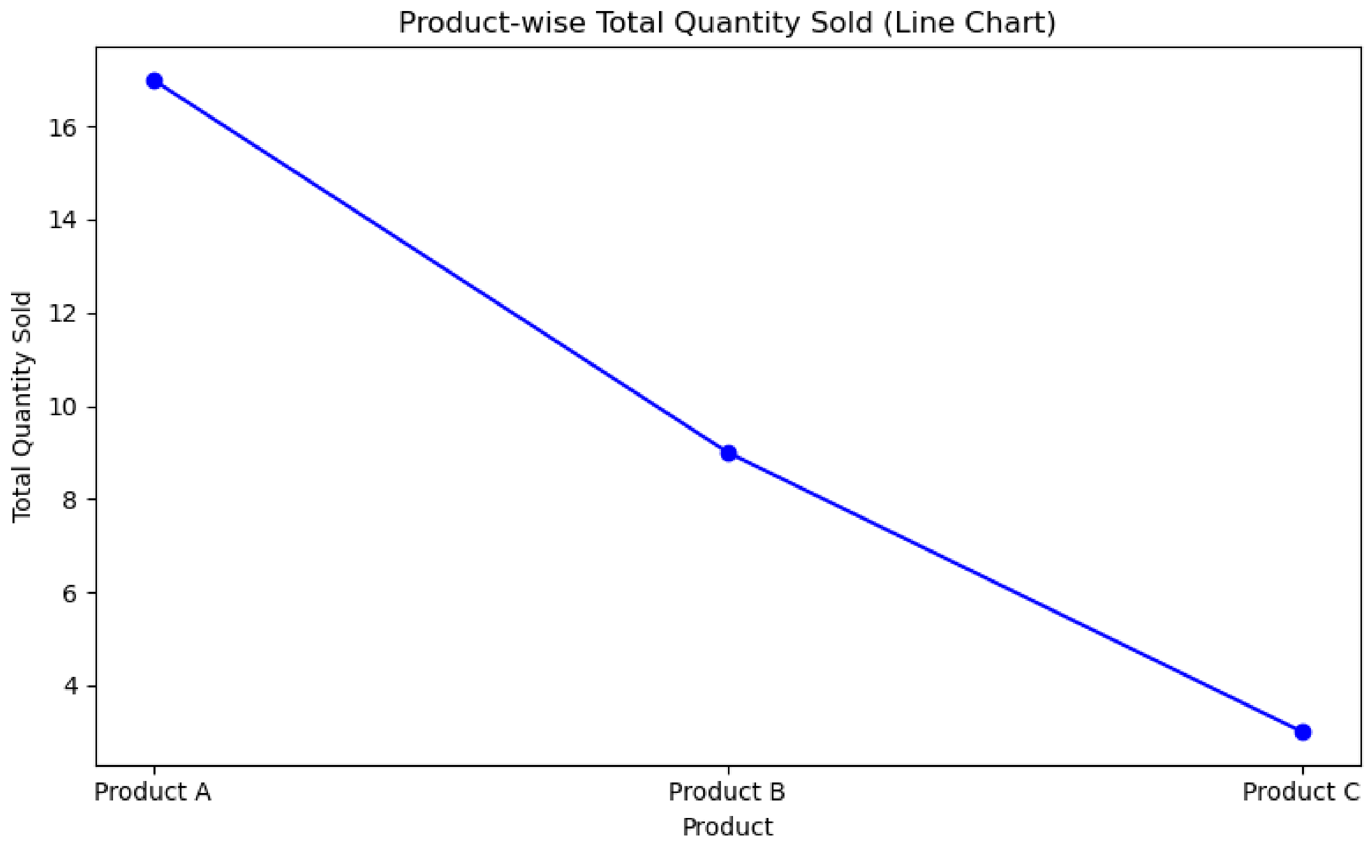
In [65]:

```
# Connect to the database
connection = sqlite3.connect('sales_data.db')

# Query to get total quantity per product
query = ('''
SELECT Product, SUM(Quantity) AS total_quantity FROM sales GROUP BY Product''')

# Read into DataFrame
df = pd.read_sql_query(query, connection)

# Close connection
connection.close()

# Plotting a pie chart
plt.pie(df['total_quantity'], labels=df['Product'], autopct='%1.1f%%', startangle=140 )
plt.title('Product-wise Share of Total Quantity Sold')
plt.axis('equal')  # Equal aspect ratio makes the pie circular
plt.tight_layout()
plt.show()
```



In [69]:

```python
# Connect to the database
connection = sqlite3.connect('sales_data.db')

# SQL query: total quantity by product
query = '''
SELECT Product, SUM(Quantity) AS total_quantity
FROM sales
GROUP BY Product
'''

# Load into DataFrame
df = pd.read_sql_query(query, connection)
connection.close()

# Line chart
plt.figure(figsize=(8, 5))
plt.plot(df['Product'], df['total_quantity'], marker='o', linestyle='-', color='blue')

# Add labels and title
plt.xlabel('Product')
plt.ylabel('Total Quantity Sold')
plt.title('Product-wise Total Quantity Sold (Line Chart)')
plt.tight_layout()

# Show the chart
plt.show()
```


Product-wise Total Quantity Sold (Line Chart)
Total Quantity Sold
16
14
12
10
8
6
4
Product A
Product B
Product C
Product

```
In [ ]:
```