In [ ]:

```
import sqlite3 as sql
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [ ]:

```
db = "estore.db"

conn = sql.connect(db)
```

In [ ]:

```
# function to run a query on request of Dr. Innayat

def run_query(q):
    return pd.read_sql(q,conn)
```

In [ ]:

```
# function to see tables in database

def show_tables():
    q = '''Select name, type from sqlite_master where type = "table"
    '''
    return run_query(q)
```

In [ ]:

```
show_tables()
```

Scenario1

Selecting Albums to Purchase;

- Hip-Hop
- Punk
- Pop
- Blues

In [18]:

```
albumbs_to_purchase = '''With usa_tracks_sold as
                        (
    SELECT il.* FROM invoice_line il
    INNER JOIN invoice i on il.invoice_id = i.invoice_id
    INNER JOIN customer c on i.customer_id = c.customer_id
    WHERE c.country = "USA"
   )


                        Select
                        g.name genre,
                        count(uts.invoice_line_id) tracks_sold,
                        cast(count(uts.invoice_line_id) as float)/(select count(*) from usa_tracks_sold) percentage_sol
                        from usa_tracks_sold uts
                        inner join track t on t.track_id = uts.track_id
                        inner join genre g on g.genre_id = t.genre_id
                        Group by 1
                        order by 2 Desc limit 10'''


run_query(albumbs_to_purchase)
```

Out[18]:

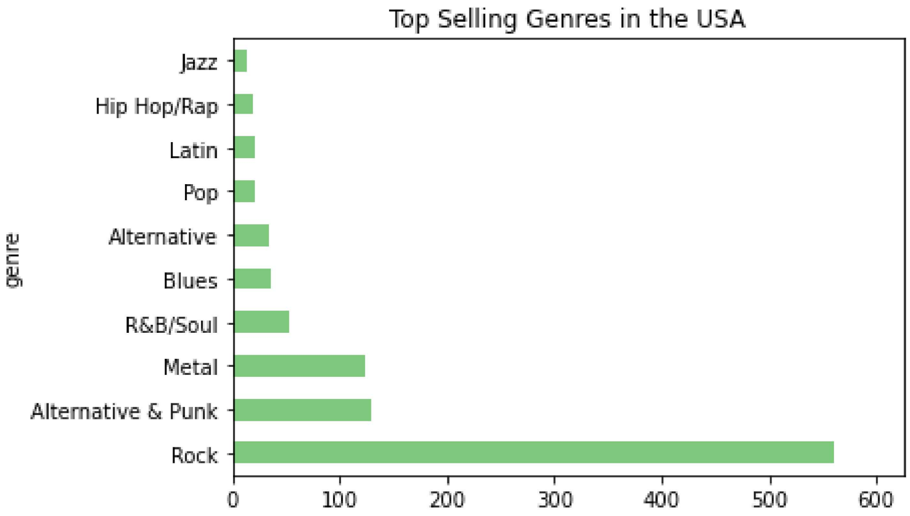
	genre	tracks_sold	percentage_sold
0	Rock	561	0.533777
1	Alternative & Punk	130	0.123692
2	Metal	124	0.117983
3	R&B/Soul	53	0.050428
4	Blues	36	0.034253
5	Alternative	35	0.033302
6	Pop	22	0.020932
7	Latin	22	0.020932
8	Hip Hop/Rap	20	0.019029
9	Jazz	14	0.013321

In [19]:

```
genre_sales_usa = run_query(albumbs_to_purchase)
genre_sales_usa.set_index("genre",inplace = True, drop = True)
```

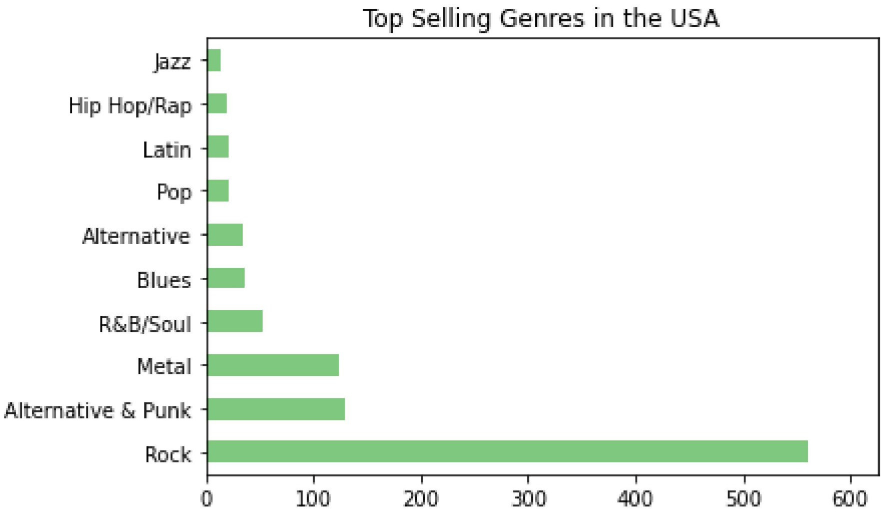
In [24]:

```
track_sold.plot.barh(title = "Top Selling Genres in the USA",
                                         colormap = plt.cm.Accent,
                                         xlim = (0,625))
plt.show()
```



In [25]:

```
track_sold.plot.barh(title = "Top Selling Genres in the USA",
                                         colormap = plt.cm.Accent,
                                         xlim = (0,625))
plt.ylabel("")

plt.show()
```

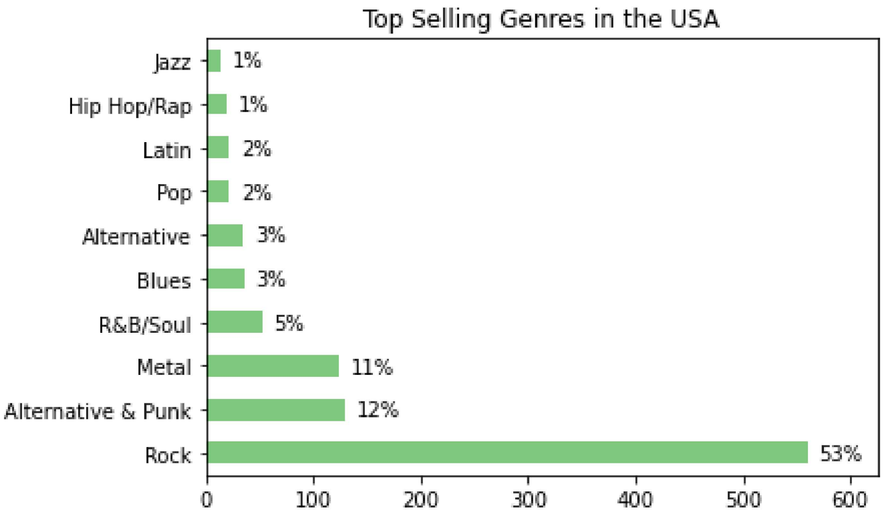


In [26]:

```
track_sold = genre_sales_usa["tracks_sold"]
pct_sold = genre_sales_usa["percentage_sold"]
pct_sold = (pct_sold*100).astype(int).astype(str)+"%"
```

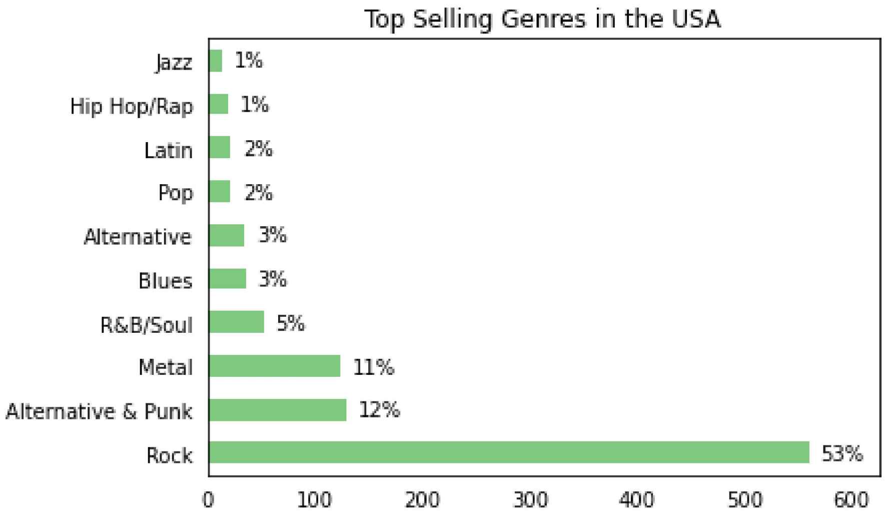
In [27]:

```
track_sold.plot.barh(title = "Top Selling Genres in the USA",
                                         colormap = plt.cm.Accent,
                                         xlim = (0,625))
plt.ylabel("")

for x,y,txt in zip(track_sold, range(10),pct_sold):
    plt.annotate(txt, (x  + 10, y -0.15))
plt.show()
```



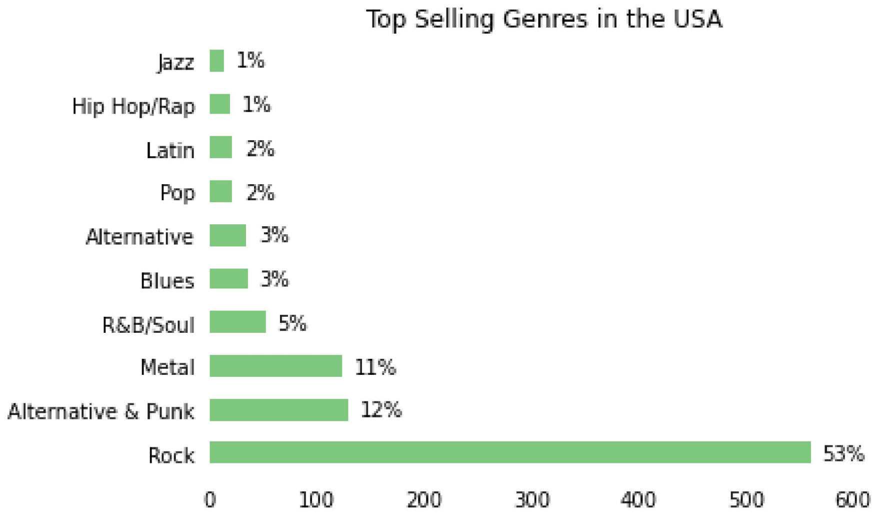
In [28]:

```
track_sold.plot.barh(title = "Top Selling Genres in the USA",
                                         colormap = plt.cm.Accent,
                                         xlim = (0,625))
plt.ylabel("")

for x,y,txt in zip(track_sold, range(10),pct_sold):
    plt.annotate(txt, (x  + 10, y -0.15))

plt.tick_params(bottom = False, left = False)
plt.show()
```



In [29]:

```
ax = track_sold.plot.barh(title = "Top Selling Genres in the USA",
                                         colormap = plt.cm.Accent,
                                         xlim = (0,625))
plt.ylabel("")

for x,y,txt in zip(track_sold, range(10),pct_sold): # labelling the bars
    plt.annotate(txt, (x  + 10, y -0.15))

plt.tick_params(bottom = False, left = False) # remove ticks

for spine in ax.spines.values():
    spine.set_visible(False)

plt.show()
```



Based on the sales of tracks across different genres in the USA, we should purchase the new albums by the following artists:

- Punk
- Blues
- Pop

It's worth keeping in mind that combined, these three genres only make up only 17% of total sales, so we should be on the lookout for artists and albums from the **rock** genre, which accounts for 53% of sales.

Scenario2

Analyzing Sales by Country

In [30]:

```
sales_by_country = '''With country_or_other AS (Select case when
            (select count(country) from customer where country = c.country) = 1
            then "other"
            Else c.country
            End as country, c.customer_id, il.invoice_id, il.unit_price
            From customer c
            inner join invoice i on c.customer_id = i.customer_id
            inner join invoice_line il on il.invoice_id = i.invoice_id)

            Select country,
            customers,
            total_sales,
            average_order,
            customer_lifetime_value
            From
            (Select
            country,
            count(distinct customer_id) customers,
            sum(unit_price) total_sales,
            sum(unit_price)/count(distinct invoice_id) average_order,
            sum(unit_price)/count(distinct customer_id) customer_lifetime_value,
            case when country = "other" then 1
            Else 0 END as sort
            From country_or_other group by 1 order by sort, total_sales DESC)'''

run_query(sales_by_country )
```

Out[30]:

	country	customers	total_sales	average_order	customer_lifetime_value
0	USA	13	1040.49	7.942672	80.037692
1	Canada	8	535.59	7.047237	66.948750
2	Brazil	5	427.68	7.011148	85.536000
3	France	5	389.07	7.781400	77.814000
4	Germany	4	334.62	8.161463	83.655000
5	Czech Republic	2	273.24	9.108000	136.620000
6	United Kingdom	3	245.52	8.768571	81.840000
7	Portugal	2	185.13	6.383793	92.565000
8	India	2	183.15	8.721429	91.575000
9	other	15	1094.94	7.448571	72.996000

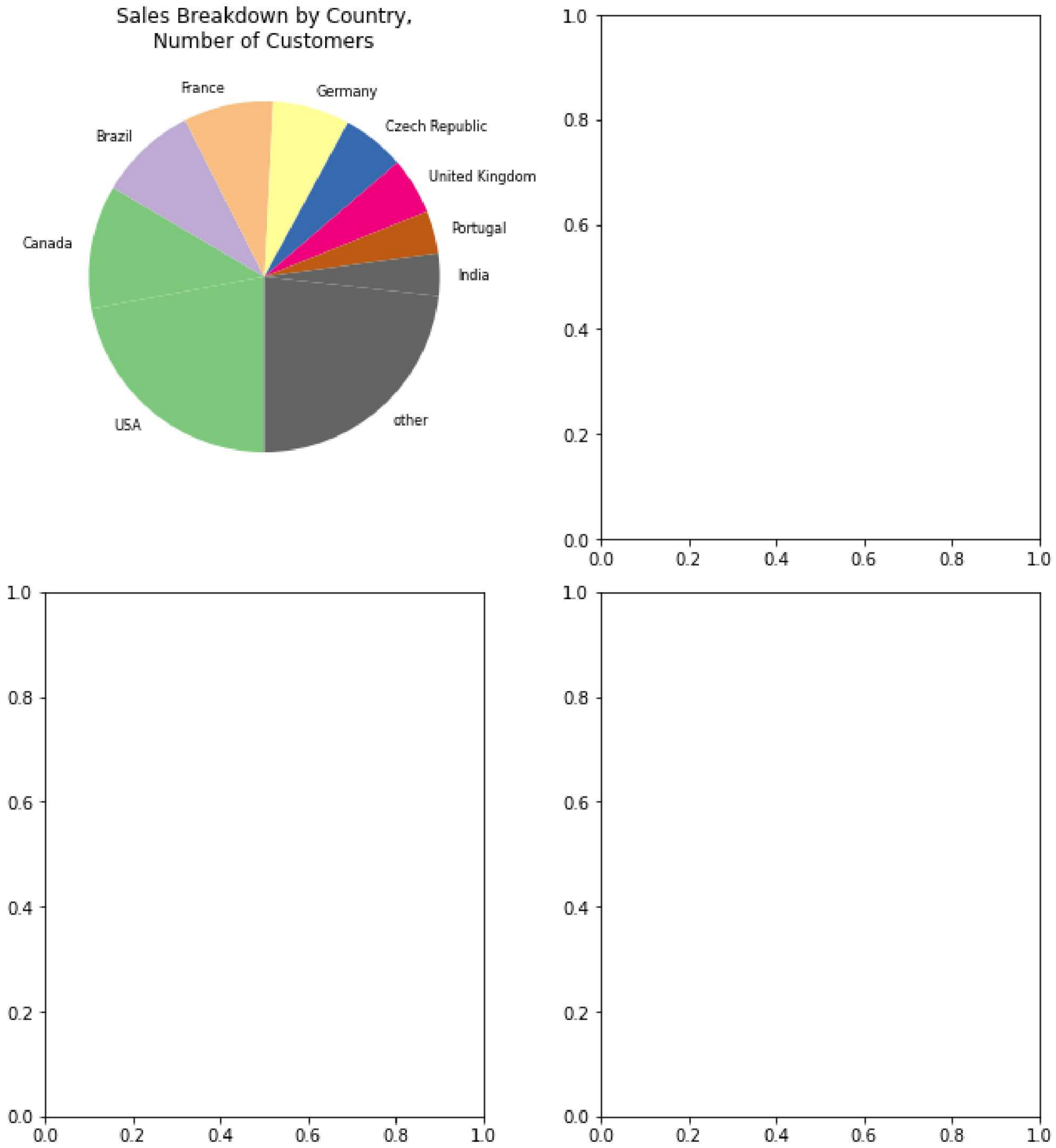
In [31]:

```
country_metrics = run_query(sales_by_country)
country_metrics.set_index("country", inplace = True, drop = True)
```

In [32]:

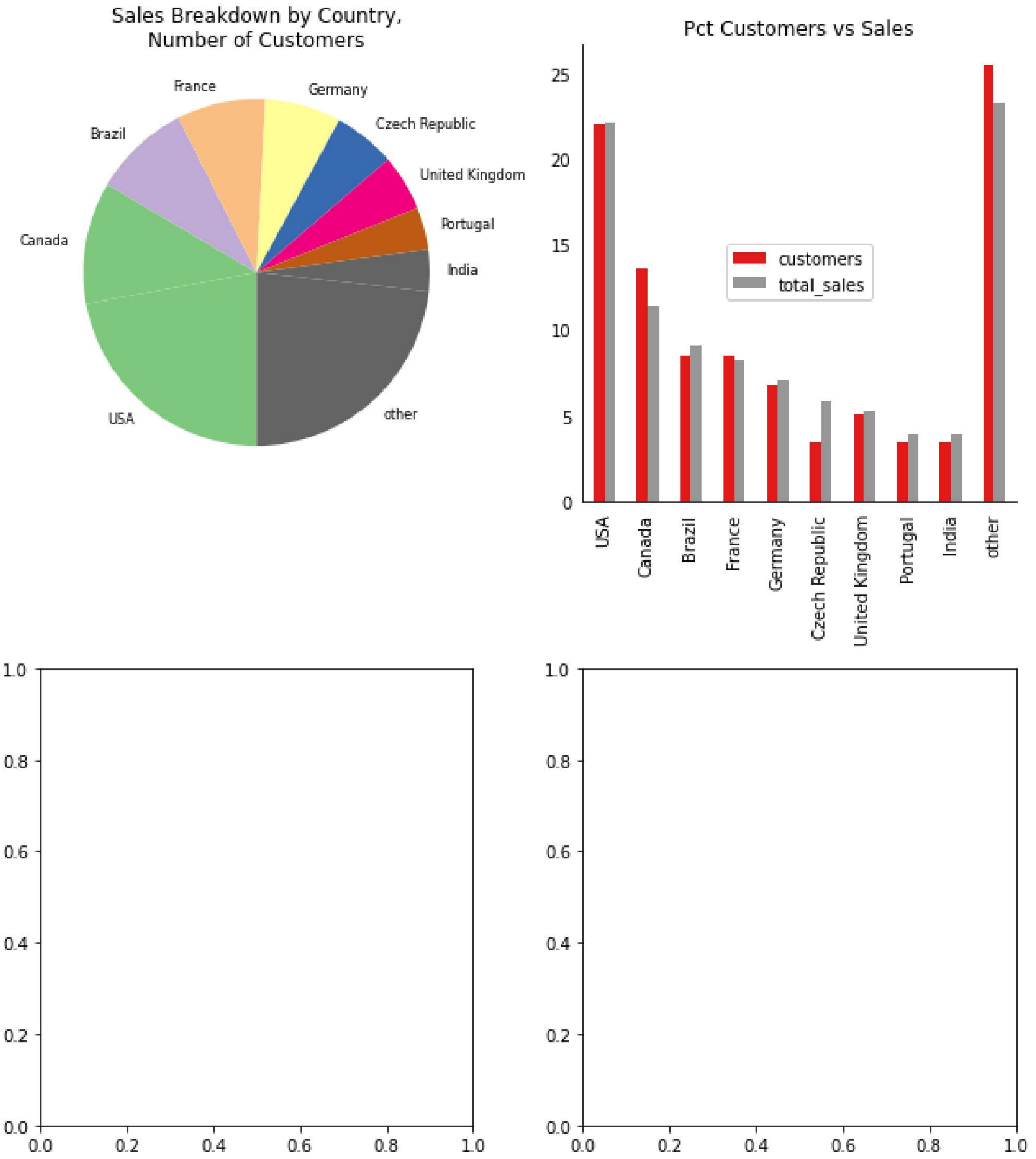
```
fig, ax = plt.subplots(2,2,figsize = (9,10))
ax1, ax2, ax3, ax4 = ax.flatten()

# top left
```

```
Sales_breakdown =country_metrics["total_sales"].rename("")
Sales_breakdown.plot.pie(ax=ax1, title = "Sales Breakdown by Country,\nNumber of Customers",
                         colormap=plt.cm.Accent, startangle = -90,
                         counterclock = False, fontsize = 8)
plt.tight_layout()

plt.show()
```


Sales Breakdown by Country,
Number of Customers
France
Germany
Czech Republic
United Kingdom
Portugal
India
other
USA
Canada
Brazil
0.0
0.2
0.4
0.6
0.8
1.0

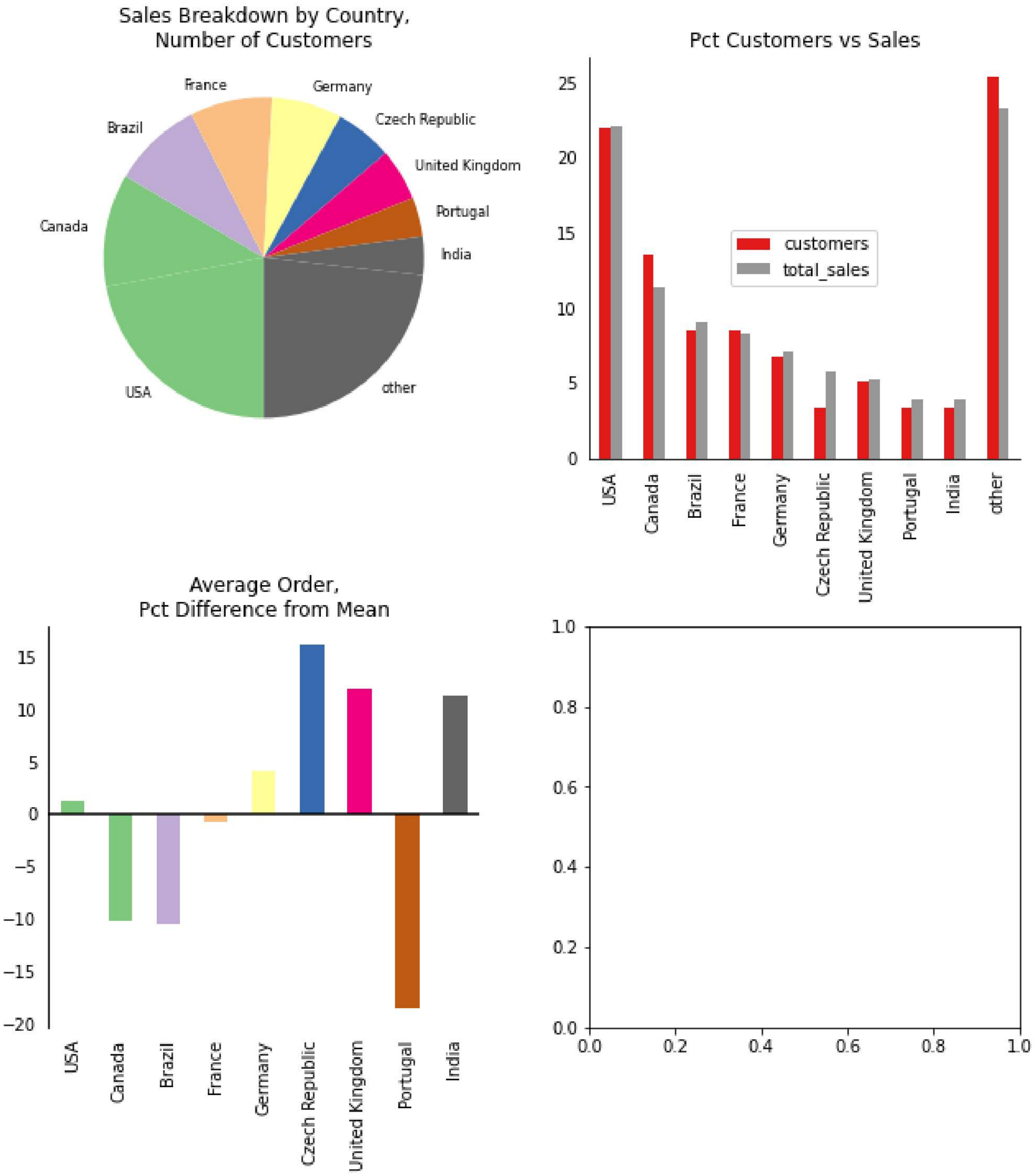
In [33]:

```
fig, ax = plt.subplots(2,2,figsize = (9,10))
ax1, ax2, ax3, ax4 = ax.flatten()

# top left
Sales_breakdown =country_metrics["total_sales"].rename("")
Sales_breakdown.plot.pie(ax=ax1, title = "Sales Breakdown by Country,\nNumber of Customers",
                         colormap=plt.cm.Accent, startangle = -90,
                         counterclock = False, fontsize = 8,
                         )

# top right
cvd_cols = ["customers","total_sales"]
custs_vs_dollars = country_metrics[cvd_cols]
custs_vs_dollars.index.name = ""
custs_vs_dollars/= (custs_vs_dollars.sum())/100

custs_vs_dollars.plot.bar(ax = ax2, colormap = plt.cm.Set1,
                          title ="Pct Customers vs Sales")
ax2.tick_params(top = "off", right = "off", left = "off", bottom = "off")
ax2.spines["top"].set_visible(False)
ax2.spines["right"].set_visible(False)
ax2.tick_params(top=False, right=False, left=False, bottom=False)

ax2.legend(loc = "center")

plt.tight_layout()

plt.show()
```


Sales Breakdown by Country,
Number of Customers
France
Germany
Czech Republic
Brazil
United Kingdom
Portugal
Canada
India
USA
other
Pct Customers vs Sales
25
20
15
10
5
0
customers
total_sales
USA
Canada
Brazil
France
Germany
Czech Republic
United Kingdom
Portugal
India
other
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

In [34]:

```
fig, ax = plt.subplots(2,2,figsize = (9,10))
ax1, ax2, ax3, ax4 = ax.flatten()
color = [plt.cm.Accent(i) for i in np.linspace(0, 1, country_metrics.shape[0])]

# top left
Sales_breakdown =country_metrics["total_sales"].rename("")
Sales_breakdown.plot.pie(ax=ax1, title = "Sales Breakdown by Country,\nNumber of Customers",
                         colormap=plt.cm.Accent, startangle = -90,
                         counterclock = False, fontsize = 8,
                         )

# top right
cvd_cols = ["customers","total_sales"]
custs_vs_dollars = country_metrics[cvd_cols]
custs_vs_dollars.index.name = ""
custs_vs_dollars/= (custs_vs_dollars.sum())/100

custs_vs_dollars.plot.bar(ax = ax2, colormap = plt.cm.Set1,
                          title ="Pct Customers vs Sales")
ax2.tick_params(top = "off", right = "off", left = "off", bottom = "off")
ax2.spines["top"].set_visible(False)
ax2.spines["right"].set_visible(False)
ax2.tick_params(top=False, right=False, left=False, bottom=False)

ax2.legend(loc = "center")

# bottom left
avg_order = country_metrics["average_order"].copy()
avg_order.index.name = ""
difference_from_avg = avg_order * 100 / avg_order.mean() - 100
difference_from_avg.drop("other", inplace = True)

difference_from_avg.plot.bar(ax=ax3,
                             title = "Average Order,\nPct Difference from Mean",
                             color = color)
ax3.tick_params(top=False, right=False, left=False, bottom=False)
ax3.axhline(0, color = "k")
ax3.spines["top"].set_visible(False)
ax3.spines["right"].set_visible(False)
ax3.spines["bottom"].set_visible(False)

plt.tight_layout()

plt.show()
```

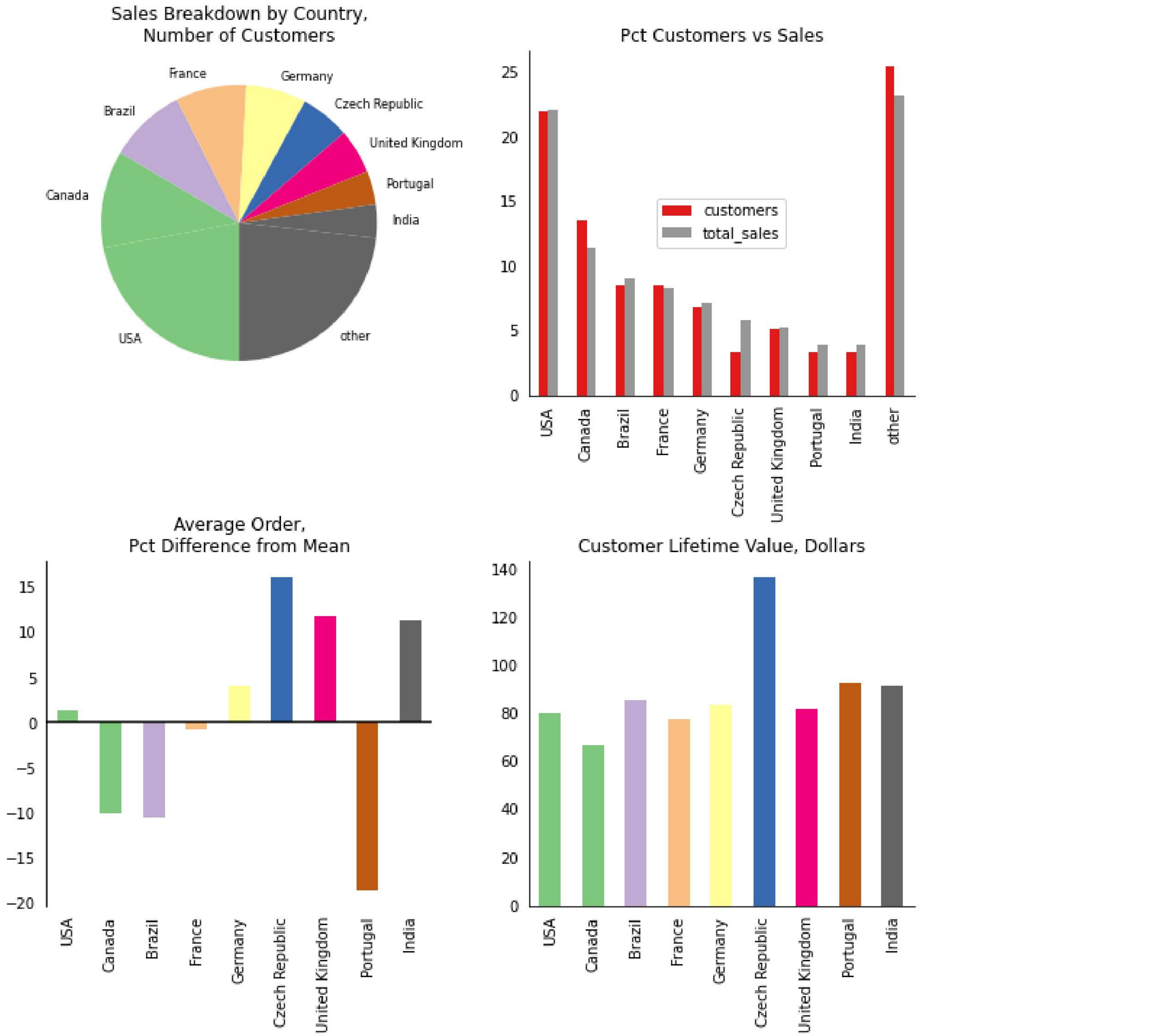

Sales Breakdown by Country,
Number of Customers

Pct Customers vs Sales

Average Order,
Pct Difference from Mean

In [35]:

```
fig, ax = plt.subplots(2,2,figsize = (9,10))
ax1, ax2, ax3, ax4 = ax.flatten()
color = [plt.cm.Accent(i) for i in np.linspace(0, 1, country_metrics.shape[0])]

# top left
Sales_breakdown =country_metrics["total_sales"].rename("")
Sales_breakdown.plot.pie(ax=ax1, title = "Sales Breakdown by Country,\nNumber of Customers",
                         colormap=plt.cm.Accent, startangle = -90,
                         counterclock = False, fontsize = 8,
                         )

# top right
cvd_cols = ["customers","total_sales"]
custs_vs_dollars = country_metrics[cvd_cols]
custs_vs_dollars.index.name = ""
custs_vs_dollars/= (custs_vs_dollars.sum())/100

custs_vs_dollars.plot.bar(ax = ax2, colormap = plt.cm.Set1,
                          title ="Pct Customers vs Sales")
ax2.tick_params(top = "off", right = "off", left = "off", bottom = "off")
ax2.spines["top"].set_visible(False)
ax2.spines["right"].set_visible(False)
ax2.tick_params(top=False, right=False, left=False, bottom=False)

ax2.legend(loc = "center")

# bottom left
avg_order = country_metrics["average_order"].copy()
avg_order.index.name = ""
difference_from_avg = avg_order * 100 / avg_order.mean() - 100
difference_from_avg.drop("other", inplace = True)

difference_from_avg.plot.bar(ax=ax3,
                             title = "Average Order,\nPct Difference from Mean",
                             color = color)
ax3.tick_params(top=False, right=False, left=False, bottom=False)
ax3.axhline(0, color = "k")
ax3.spines["top"].set_visible(False)
ax3.spines["right"].set_visible(False)
ax3.spines["bottom"].set_visible(False)

# bottom right
ltv = country_metrics["customer_lifetime_value"].copy()
ltv.index.name = ''
ltv.drop("other", inplace = True)

ltv.plot.bar(ax = ax4, title = "Customer Lifetime Value, Dollars",
             color = color)
ax4.tick_params(top=False, right=False, left=False, bottom=False)
ax4.spines["top"].set_visible(False)
ax4.spines["right"].set_visible(False)

plt.tight_layout()

plt.show()
```


Sales Breakdown by Country,
Number of Customers
France
Germany
Czech Republic
Brazil
United Kingdom
Portugal
Canada
India
USA
other
Pct Customers vs Sales
customers
total_sales
25
20
15
10
5
0
USA
Canada
Brazil
France
Germany
Czech Republic
United Kingdom
Portugal
India
other
Average Order,
Pct Difference from Mean
15
10
5
0
−5
−10
−15
−20
USA
Canada
Brazil
France
Germany
Czech Republic
United Kingdom
Portugal
India
Customer Lifetime Value, Dollars
140
120
100
80
60
40
20
0
USA
Canada
Brazil
France
Germany
Czech Republic
United Kingdom
Portugal
India

In [ ]:

In [ ]: