Challenge

Which EU countries has the highest average takings per customer?
Create and export a chart and a csv/ excel

In [1]:
```
# EU countries
eu_countries = ['Austria', 'Belgium', 'Bulgaria', 'Croatia', 'Czech Republic', 'Denmark', 'Estonia', 'Finlar
                'Germany', 'Greece', 'Hungary', 'Ireland', 'Italy', 'Latvia', 'Lithuania', 'Luxembourg', 'Ma
                'Poland', 'Portugal', 'Romania', 'Slovakia', 'Slovenia', 'Spain', 'Sweden']
```

In [2]:
```
import pandas as pd
```

In [3]:
```
payment = pd.read_csv('payment.csv')
payment.head()
```

Out[3]:

	payment_id	customer_id	staff_id	rental_id	amount	payment_date
0	16050	269	2	7	1.99	2007-01-24 21:40:19.996577
1	16051	269	1	98	0.99	2007-01-25 15:16:50.996577
2	16052	269	2	678	6.99	2007-01-28 21:44:14.996577
3	16053	269	2	703	0.99	2007-01-29 00:58:02.996577
4	16054	269	1	750	4.99	2007-01-29 08:10:06.996577

In [4]:
```
# group payments table by customer_id and sum amount as total_amount
customer_payments = payment[['customer_id', 'amount']].groupby('customer_id').agg(total_sales = ('amount',
```

In [5]:
```
customer_payments.head()
```

Out[5]:

customer_id	total_sales
1	118.68
2	128.73
3	135.74
4	81.78
5	144.62

We will need some other tables to allow us to find out which country each customer is in

In [6]:
```
address = pd.read_csv('address.csv')
address.head()
```

Out[6]:

	address_id	address	address2	district	city_id	postal_code	phone	last_update
0	1	47 MySakila Drive	NaN	Alberta	300	NaN	NaN	2006-02-15 09:45:30
1	2	28 MySQL Boulevard	NaN	QLD	576	NaN	NaN	2006-02-15 09:45:30
2	3	23 Workhaven Lane	NaN	Alberta	300	NaN	1.403334e+10	2006-02-15 09:45:30
3	4	1411 Lillydale Drive	NaN	QLD	576	NaN	6.172236e+09	2006-02-15 09:45:30
4	5	1913 Hanoi Way	NaN	Nagasaki	463	35200.0	2.830338e+10	2006-02-15 09:45:30

In [7]:
```
city = pd.read_csv('city.csv')
city.head()
```

Out[7]:

	city_id	city	country_id	last_update
0	1	A Corua (La Corua)	87	2006-02-15 09:45:25
1	2	Abha	82	2006-02-15 09:45:25
2	3	Abu Dhabi	101	2006-02-15 09:45:25
3	4	Acua	60	2006-02-15 09:45:25
4	5	Adana	97	2006-02-15 09:45:25

In [8]:
```
country = pd.read_csv('country.csv')
country.head()
```

Out[8]:

	country_id	country	last_update
0	1	Afghanistan	2006-02-15 09:44:00
1	2	Algeria	2006-02-15 09:44:00
2	3	American Samoa	2006-02-15 09:44:00
3	4	Angola	2006-02-15 09:44:00
4	5	Anguilla	2006-02-15 09:44:00

In [9]:
```
customer = pd.read_csv('customer.csv')
```

In [10]:
```
customer_payment_details_full = customer.merge(right = customer_payments,
                                               how = 'left', left_on = 'customer_id', right_on = 'customer_id'
                                               ).merge(right = address,
                                                       how = 'left', left_on = 'address_id', right_on = 'address_id
                                                       ).merge(right = city,
                                                               how = 'left', left_on = 'city_id', right_on = 'city_i
                                                               ).merge(right = country,
                                                                       how = 'left', left_on = 'country_id', right_on
```

```
C:\Users\udgar\AppData\Local\Temp\ipykernel_7912\573947076.py:1: FutureWarning: Passing 'suffixes' which ca
use duplicate columns {'last_update_x'} in the result is deprecated and will raise a MergeError in a future
version.
  customer_payment_details_full = customer.merge(right = customer_payments,
```

In [11]:
```
customer_payment_details_full.head()
```

Out[11]:

	customer_id	store_id	first_name	last_name	email	address_id	activebool	create_date	last_u
0	1	1	MARY	SMITH	MARY.SMITH@sakilacustomer.org	5	True	2006-02-14	20
1	2	1	PATRICIA	JOHNSON	PATRICIA.JOHNSON@sakilacustomer.org	6	True	2006-02-14	20
2	3	1	LINDA	WILLIAMS	LINDA.WILLIAMS@sakilacustomer.org	7	True	2006-02-14	20
3	4	2	BARBARA	JONES	BARBARA.JONES@sakilacustomer.org	8	True	2006-02-14	20
4	5	1	ELIZABETH	BROWN	ELIZABETH.BROWN@sakilacustomer.org	9	True	2006-02-14	20

5 rows × 23 columns

In [12]:
```
# column names of new table
customer_payment_details_full.columns
```

Out[12]:
```
Index(['customer_id', 'store_id', 'first_name', 'last_name', 'email',
       'address_id', 'activebool', 'create_date', 'last_update_x', 'active',
       'total_sales', 'address', 'address2', 'district', 'city_id',
       'postal_code', 'phone', 'last_update_y', 'city', 'country_id',
       'last_update_x', 'country', 'last_update_y'],
      dtype='object')
```

In [13]:
```
customer_payment_details = customer_payment_details_full[['customer_id', 'address_id','activebool','total_sa
customer_payment_details.head()
```

Out[13]:

	customer_id	address_id	activebool	total_sales	city_id	city	country_id	country
0	1	5	True	118.68	463	Sasebo	50	Japan
1	2	6	True	128.73	449	San Bernardino	103	United States
2	3	7	True	135.74	38	Athenai	39	Greece
3	4	8	True	81.78	349	Myingyan	64	Myanmar
4	5	9	True	144.62	361	Nantou	92	Taiwan

In [14]:
```
usefull_cols = ['total_sales', 'country', 'country_id']

customer_payment_details_grouped = customer_payment_details[usefull_cols].groupby('country_id')

avg_country_customer = customer_payment_details_grouped.agg({'total_sales':'mean', 'country':'max'})

avg_country_customer.sort_values('total_sales', ascending = False).head()
```

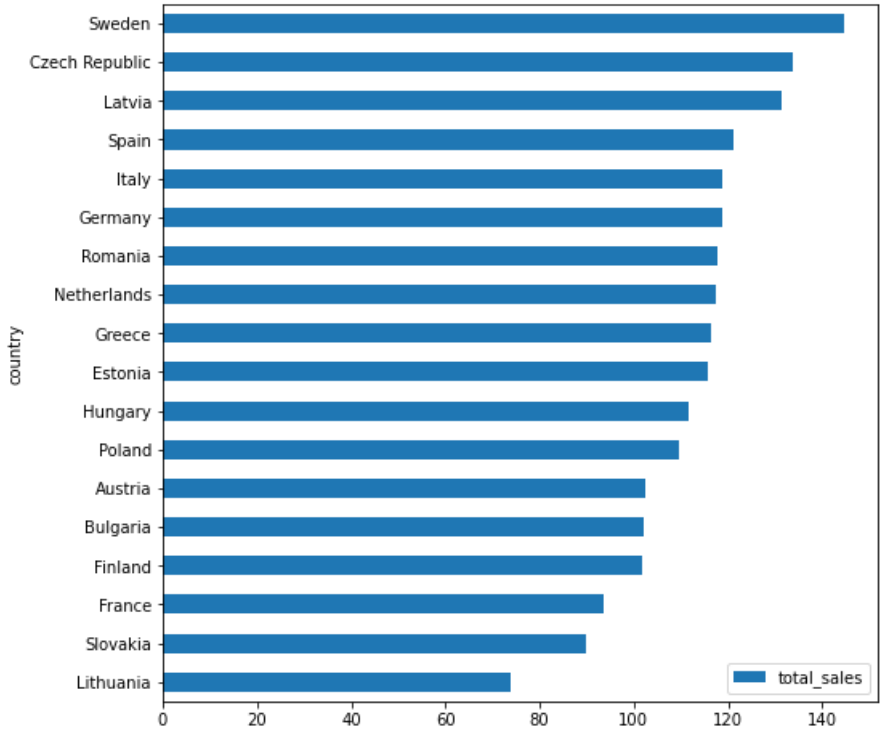
Out[14]:

country_id	total_sales	country
79	216.54	Runion
41	152.66	Holy See (Vatican City State)
65	148.69	Nauru
90	144.66	Sweden
42	142.70	Hong Kong

In [15]:
```
# sort values
avg_country_customer = avg_country_customer.sort_values('total_sales')
```

In [16]:
```
# selecting only EU contries
avg_country_customer_eu = avg_country_customer[avg_country_customer['country'].isin(eu_countries)]
```

In [17]:
```
# plotting Eu hbar
avg_country_customer_eu.plot.barh(x='country', y='total_sales', figsize = (8,8));
```



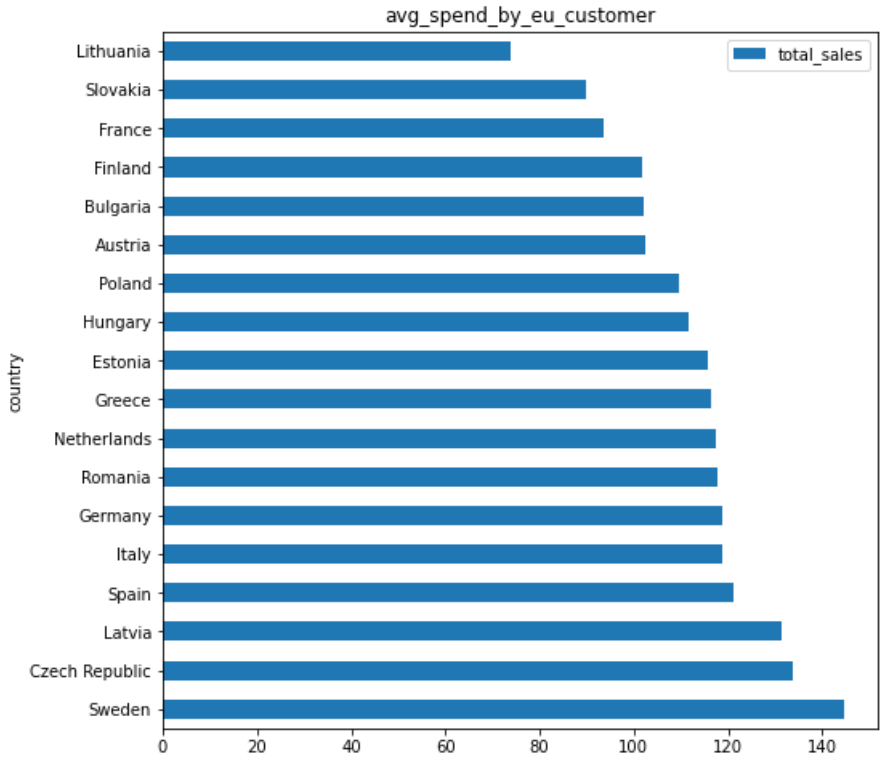
In [18]:
```
avg_country_customer_eu = avg_country_customer_eu.sort_values('total_sales', ascending = False)
avg_country_customer_eu.head()
```

Out[18]:

country_id	total_sales	country
90	144.660	Sweden
26	133.710	Czech Republic
54	131.200	Latvia
87	121.316	Spain
49	118.730	Italy

In [19]:
```
# Export chart
# save the plot

plot = avg_country_customer_eu.plot.barh(x='country', y='total_sales', figsize = (8,8))
plot.set_title('avg_spend_by_eu_customer')
plot.get_figure().savefig('avg_eu_sales.pdf', format='pdf')
```



Export data to a CSV or Excel

In [20]:
```
avg_country_customer_eu.to_csv('avg_country_customer_eu.csv', index = False)
```

Excel spreadsheet

In [21]:
```
avg_country_customer_eu.to_excel('avg_country_customer_eu.xlsx', sheet_name = 'avg_country_customer_eu')
```

In [ ]: