Instructions

Subset, clean, and reformat the `bank_marketing.csv` dataset to create and store three new files based on the requirements detailed in the notebook.

- Split and tidy bank_marketing.csv, storing as three DataFrames called `client`, `campaign`, and `economics`, each containing the columns outlined in the notebook and formatted to the data types listed.
- Save the three DataFrames to csv files, without an index, as `client.csv`, `campaign.csv`, and `economics.csv` respectively.

Personal loans are a lucrative revenue stream for banks. The typical interest rate of a two-year loan in the UK is around 10%. This might not sound like a lot, but in September 2022 alone UK consumers borrowed around £1.5 billion, which would mean approximately £300 million in interest generated by banks over two years!

You have been asked to work with a bank to clean the data they collected as part of a recent marketing campaign, which aimed to get customers to take out a personal loan. They plan to conduct more marketing campaigns going forward so would like you to ensure it conforms to the specific structure and data types that they specify so that they can then use the cleaned data you provide to set up a PostgreSQL database, which will store this campaign's data and allow data from future campaigns to be easily imported.

They have supplied you with a csv file called `"bank_marketing.csv"`, which you will need to clean, reformat, and split the data, saving three final csv files. Specifically, the three files should have the names and contents as outlined below:

`client.csv`

column	data type	description	cleaning requirements
`client_id`	`integer`	Client ID	N/A
`age`	`integer`	Client's age in years	N/A
`job`	`object`	Client's type of job	Change `"."` to `"_"`
`marital`	`object`	Client's marital status	N/A
`education`	`object`	Client's level of education	Change `"."` to `"_"` and `"unknown"` to `np.nan`
`credit_default`	`bool`	Whether the client's credit is in default	Convert to `boolean` data type: `1` if `"yes"`, otherwise `0`
`mortgage`	`bool`	Whether the client has an existing mortgage (housing loan)	Convert to boolean data type: `1` if `"yes"`, otherwise `0`

`campaign.csv`

column	data type	description	cleaning requirements
`client_id`	`integer`	Client ID	N/A
`number_contacts`	`integer`	Number of contact attempts to the client in the current campaign	N/A
`contact_duration`	`integer`	Last contact duration in seconds	N/A
`previous_campaign_contacts`	`integer`	Number of contact attempts to the client in the previous campaign	N/A
`previous_outcome`	`bool`	Outcome of the previous campaign	Convert to boolean data type: `1` if `"success"`, otherwise `0`.
`campaign_outcome`	`bool`	Outcome of the current campaign	Convert to boolean data type: `1` if `"yes"`, otherwise `0`.
`last_contact_date`	`datetime`	Last date the client was contacted	Create from a combination of `day`, `month`, and a newly created `year` column (which should have a value of `2022`); **Format =** `"YYYY-MM-DD"`

`economics.csv`

column	data type	description	cleaning requirements
`client_id`	`integer`	Client ID	N/A
`cons_price_idx`	`float`	Consumer price index (monthly indicator)	N/A
`euribor_three_months`	`float`	Euro Interbank Offered Rate (euribor) three-month rate (daily indicator)	N/A

In [1]:
```
import pandas as pd
import numpy as np

# Read in csv
# marketing = pd.read_csv("bank_marketing.csv")
marketing = pd.read_csv("../data_raw/bank_marketing.csv")

marketing.head(20)
# marketing.tail(20)
```

Out[1]:

	client_id	age	job	marital	education	credit_default	mortgage	month	day	contact_duration	number_contacts	previous_campaign_contacts	previous_outcome	cons_price_idx
0	0	56	housemaid	married	basic.4y	no	no	may	13	261	1	0	nonexistent	93.994
1	1	57	services	married	high.school	unknown	no	may	19	149	1	0	nonexistent	93.994
2	2	37	services	married	high.school	no	yes	may	23	226	1	0	nonexistent	93.994
3	3	40	admin.	married	basic.6y	no	no	may	27	151	1	0	nonexistent	93.994
4	4	56	services	married	high.school	no	no	may	3	307	1	0	nonexistent	93.994
5	5	45	services	married	basic.9y	unknown	no	may	5	198	1	0	nonexistent	93.994
6	6	59	admin.	married	professional.course	no	no	may	3	139	1	0	nonexistent	93.994
7	7	41	blue-collar	married	unknown	unknown	no	may	12	217	1	0	nonexistent	93.994
8	8	24	technician	single	professional.course	no	yes	may	21	380	1	0	nonexistent	93.994
9	9	25	services	single	high.school	no	yes	may	5	50	1	0	nonexistent	93.994
10	10	41	blue-collar	married	unknown	unknown	no	may	8	55	1	0	nonexistent	93.994
11	11	25	services	single	high.school	no	yes	may	9	222	1	0	nonexistent	93.994
12	12	29	blue-collar	single	high.school	no	no	may	29	137	1	0	nonexistent	93.994
13	13	57	housemaid	divorced	basic.4y	no	yes	may	14	293	1	0	nonexistent	93.994
14	14	35	blue-collar	married	basic.6y	no	yes	may	29	146	1	0	nonexistent	93.994
15	15	54	retired	married	basic.9y	unknown	yes	may	29	174	1	0	nonexistent	93.994
16	16	35	blue-collar	married	basic.6y	no	yes	may	1	312	1	0	nonexistent	93.994
17	17	46	blue-collar	married	basic.6y	unknown	yes	may	9	440	1	0	nonexistent	93.994
18	18	50	blue-collar	married	basic.9y	no	yes	may	6	353	1	0	nonexistent	93.994
19	19	39	management	single	basic.9y	unknown	no	may	2	195	1	0	nonexistent	93.994

In [2]:
```
# Split into the three tables
client = marketing[["client_id", "age", "job", "marital",
                    "education", "credit_default", "mortgage"]].copy()
campaign = marketing[["client_id", "number_contacts", "month", "day",
                      "contact_duration", "previous_campaign_contacts", "previous_outcome", "campaign_outcome"]].copy()
economics = marketing[["client_id", "cons_price_idx", "euribor_three_months"]].copy()
```

In [3]:
```
## Editing the client dataset
# Clean education column
client.loc[:, "education"] = client["education"].str.replace(".", "_")
client.loc[:, "education"] = client["education"].replace("unknown", np.nan)

# Clean job column
client.loc[:, "job"] = client["job"].str.replace(".", "_")

# Clean and convert client columns to bool data type
for col in ["credit_default", "mortgage"]:
  client.loc[:, col] = client[col].map({"yes": 1,
                                        "no": 0,
                                        "unknown": 0})

  client.loc[:, col] = client[col].astype(bool)

client.head(20)
# client.tail(20)
```

Out[3]:

	client_id	age	job	marital	education	credit_default	mortgage
0	0	56	housemaid	married	basic_4y	False	False
1	1	57	services	married	high_school	False	False
2	2	37	services	married	high_school	False	True
3	3	40	admin_	married	basic_6y	False	False
4	4	56	services	married	high_school	False	False
5	5	45	services	married	basic_9y	False	False
6	6	59	admin_	married	professional_course	False	False
7	7	41	blue-collar	married	NaN	False	False
8	8	24	technician	single	professional_course	False	True
9	9	25	services	single	high_school	False	True
10	10	41	blue-collar	married	NaN	False	False
11	11	25	services	single	high_school	False	True
12	12	29	blue-collar	single	high_school	False	False
13	13	57	housemaid	divorced	basic_4y	False	True
14	14	35	blue-collar	married	basic_6y	False	True
15	15	54	retired	married	basic_9y	False	True
16	16	35	blue-collar	married	basic_6y	False	True
17	17	46	blue-collar	married	basic_6y	False	True
18	18	50	blue-collar	married	basic_9y	False	True
19	19	39	management	single	basic_9y	False	False

In [4]:
```
## Editing the campaign dataset
# Change campaign_outcome to binary values
campaign["campaign_outcome"] = campaign["campaign_outcome"].map({"yes": 1,
                                                                 "no": 0})

# Convert previous_outcome to binary values
campaign["previous_outcome"] = campaign["previous_outcome"].map({"success": 1,
                                                                 "failure": 0,
                                                                 "nonexistent": 0})

# Add year column
campaign["year"] = "2022"

# Convert day to string
campaign["day"] = campaign["day"].astype(str)

# Add last_contact_date column
campaign["last_contact_date"] = campaign["year"] + "-" + campaign["month"] + "-" + campaign["day"]

# Convert to datetime
campaign["last_contact_date"] = pd.to_datetime(campaign["last_contact_date"],
                                               format="%Y-%b-%d")
                                #  format="%Y-%m-%d")

# Clean and convert outcome columns to bool
for col in ["campaign_outcome", "previous_outcome"]:
  campaign[col] = campaign[col].astype(bool)

# Drop unnecessary columns
campaign.drop(columns=["month", "day", "year"], inplace=True)

campaign.head(20)
# campaign.tail(20)
```

Out[4]:

	client_id	number_contacts	contact_duration	previous_campaign_contacts	previous_outcome	campaign_outcome	last_contact_date
0	0	1	261	0	False	False	2022-05-13
1	1	1	149	0	False	False	2022-05-19
2	2	1	226	0	False	False	2022-05-23
3	3	1	151	0	False	False	2022-05-27
4	4	1	307	0	False	False	2022-05-03
5	5	1	198	0	False	False	2022-05-05
6	6	1	139	0	False	False	2022-05-03
7	7	1	217	0	False	False	2022-05-12
8	8	1	380	0	False	False	2022-05-21
9	9	1	50	0	False	False	2022-05-05
10	10	1	55	0	False	False	2022-05-08
11	11	1	222	0	False	False	2022-05-09
12	12	1	137	0	False	False	2022-05-29
13	13	1	293	0	False	False	2022-05-14
14	14	1	146	0	False	False	2022-05-29
15	15	1	174	0	False	False	2022-05-29
16	16	1	312	0	False	False	2022-05-01
17	17	1	440	0	False	False	2022-05-09
18	18	1	353	0	False	False	2022-05-06
19	19	1	195	0	False	False	2022-05-02

In [5]:
```
# Save tables to individual csv files
client.to_csv("../data_cleaned/client.csv", index=False)
campaign.to_csv("../data_cleaned/campaign.csv", index=False)
economics.to_csv("../data_cleaned/economics.csv", index=False)
```

In [6]:
```
df = pd.read_csv("../data_raw/bank_marketing.csv")

for col in ["credit_default", "mortgage", "previous_outcome", "campaign_outcome"]:
    # print(col)
    print(df[col].value_counts())
    print("------------------------------")
```

```
credit_default
no         32588
unknown     8597
yes            3
Name: count, dtype: int64
------------------------------
mortgage
yes        21576
no         18622
unknown      990
Name: count, dtype: int64
------------------------------
previous_outcome
nonexistent    35563
failure         4252
success         1373
Name: count, dtype: int64
------------------------------
campaign_outcome
no     36548
yes     4640
Name: count, dtype: int64
------------------------------
```