Import Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Load the input files

```
user = pd.read_csv('USER_TAKEHOME.csv')
user.head()
```

	ID	CREATED_DATE	BIRTH_DATE	STATE	LANGUAGE	GENDER
0	5ef3b4f17053ab141787697d	2020-06-24 20:17:54.000 Z	2000-08-11 00:00:00.000 Z	CA	es-419	female
1	5ff220d383fcfc12622b96bc	2021-01-03 19:53:55.000 Z	2001-09-24 04:00:00.000 Z	PA	en	female
2	6477950aa55bb77a0e27ee10	2023-05-31 18:42:18.000 Z	1994-10-28 00:00:00.000 Z	FL	es-419	female
3	658a306e99b40f103b63ccf8	2023-12-26 01:46:22.000 Z	NaN	NC	en	NaN
4	653cf5d6a225ea102b7ecdc2	2023-10-28 11:51:50.000 Z	1972-03-19 00:00:00.000 Z	PA	en	female

Next steps: Generate code with `user` View recommended plots New interactive sheet

```
user.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100000 entries, 0 to 99999
Data columns (total 6 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   ID            100000 non-null  object
 1   CREATED_DATE  100000 non-null  object
 2   BIRTH_DATE    96325 non-null   object
 3   STATE         95188 non-null   object
 4   LANGUAGE      69492 non-null   object
 5   GENDER        94108 non-null   object
dtypes: object(6)
memory usage: 4.6+ MB
```

Converting the data types

```
user['CREATED_DATE'] = pd.to_datetime(user['CREATED_DATE'], errors='coerce')
user['BIRTH_DATE'] = pd.to_datetime(user['BIRTH_DATE'], errors='coerce').dt.date
user.head()
```

	ID	CREATED_DATE	BIRTH_DATE	STATE	LANGUAGE	GENDER
0	5ef3b4f17053ab141787697d	2020-06-24 20:17:54+00:00	2000-08-11	CA	es-419	female
1	5ff220d383fcfc12622b96bc	2021-01-03 19:53:55+00:00	2001-09-24	PA	en	female
2	6477950aa55bb77a0e27ee10	2023-05-31 18:42:18+00:00	1994-10-28	FL	es-419	female
3	658a306e99b40f103b63ccf8	2023-12-26 01:46:22+00:00	NaT	NC	en	NaN
4	653cf5d6a225ea102b7ecdc2	2023-10-28 11:51:50+00:00	1972-03-19	PA	en	female

Next steps: Generate code with `user` View recommended plots New interactive sheet

Converting the data type for CREATED_DATE, BIRTH_DATE into date-time, date formats respectively.

Missing data count

```
user.isnull().sum()
```

	0
ID	0
CREATED_DATE	0
BIRTH_DATE	3675
STATE	4812
LANGUAGE	30508
GENDER	5892

dtype: int64

Percentage of missing data in each column

```
user.isnull().sum() / len(user) * 100
```

	0
ID	0.000
CREATED_DATE	0.000
BIRTH_DATE	3.675
STATE	4.812
LANGUAGE	30.508
GENDER	5.892

dtype: float64

LANGUAGE column in missing 30 percent of data.

Checking for duplicates

```
user.duplicated().sum()
```

```
0
```

No Duplicate rows were observed in USER dataset

Unique values in columns

```
user['ID'].is_unique
```

```
True
```

This confirms that all the values in ID column are unique and can be used as primary key for this dataset

```
user['STATE'].unique()
```

```
array(['CA', 'PA', 'FL', 'NC', 'NY', 'IN', nan, 'OH', 'TX', 'NM', 'PR',
       'CO', 'AZ', 'RI', 'MO', 'NJ', 'MA', 'TN', 'LA', 'NH', 'WI', 'IA',
       'GA', 'VA', 'DC', 'KY', 'SC', 'MN', 'WV', 'DE', 'MI', 'IL', 'MS',
       'WA', 'KS', 'CT', 'OR', 'UT', 'MD', 'OK', 'NE', 'NV', 'AL', 'AK',
       'AR', 'HI', 'ME', 'ND', 'ID', 'WY', 'MT', 'SD', 'VT'], dtype=object)
```

```
user['LANGUAGE'].unique()
```

```
array(['es-419', 'en', nan], dtype=object)
```

```
user['GENDER'].unique()
```

```
array(['female', nan, 'male', 'non_binary', 'transgender',
       'prefer_not_to_say', 'not_listed', 'Non-Binary', 'unknown',
       'not_specified', "My gender isn't listed", 'Prefer not to say'],
      dtype=object)
```

Attached below are the data quality issues observed in USER DATASET:

- Several issues are observed in GENDER column due to inconsistent formatting of the values primarily due to:

  1) Case-sensitive ('non_binary' and 'Non-Binary')

  2) Underscores instead of spaces ('prefer_not_to_say' and 'Prefer not to say')

  3) Missing values and different representation of similar values ('unknown' and 'not_specified', 'not_listed' and 'My gender isn't listed').

- LANGUAGE columnn is missing 30 percent of data.

All the columns in this dataset are easy to understand and to use this dataset in further steps, the above mentioned changes are to be implemented.