Data Viz Challenge

Agenda:

1. Raw Data Ingestion & Preliminary Analysis
2. Data Quality Checks & Data Cleaning
3. Dashboard Creation
4. Next Steps

1. Raw Data Ingestion & Preliminary Analysis

- Retrieve data from CSV datasets.
- Analysis of datasets structure

```
#get info regarding datatypes
print(dataset_csv.info())
print(meta_browser.info())
print(meta_device.info())
print(meta_os.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 27 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   ?uuid                   10000 non-null  int64
 1   giorno_min              10000 non-null  datetime64[ns]
 2   giorno_max              10000 non-null  datetime64[ns]
 3   durata_vita             10000 non-null  int64
 4   conteggio_giorni_totale 10000 non-null  int64
 5   device_type             9814 non-null   float64
 6   browser_type            9591 non-null   float64
 7   os_type                 9657 non-null   float64
 8   site_giallozafferano    10000 non-null  int64
 9   site_repubblica         10000 non-null  int64
 10  site subito             10000 non-null  int64
```

- Corrective actions:
  - ✓ Import date fields in a *datetime* format
  - ✓ Split “prima_citta” columns in two different fields in order to separate country code from city code
  - ✓ Trasform float datatype into an integer datatype

2. Data Quality Checks & Data Cleaning (1/2)

- Scan for columns containing Null values

```
# create a method that checks for each dataset which columns has null values
def null_value_check(df):
    # loop through each column in the DataFrame
    for col in df.columns:
        # check if there are any null values in the column
        if df[col].isnull().any():
            print(f"for dataset {df} Column {col} has null values.".format(df=df))
        else:
            pass

#check for NaN values through dataset
datasets = [dataset_csv,meta_os,meta_device,meta_browser]
for i in datasets:
    print( null_value_check(i))
```

- Replace Null values with a dummy value (“*”)

```
#replace null with dummy value "*"
dataset_csv["device_type"].fillna("*",inplace = True)
dataset_csv["browser_type"].fillna("*",inplace = True)
dataset_csv["os_type"].fillna("*",inplace = True)
dataset_csv["prima_citta_stato_cod"].fillna("*",inplace = True)
dataset_csv["prima_citta_cod"].fillna("*",inplace = True)
```

- Remove rows with at least a null value and put them into an Error Tabel (*dataset_csv_errors)*

```
#remove rows that contains null values from the final fact table, create an error table for further analysis

q = '''select * FROM dataset_csv where device_type = '*'
or browser_type = '*' or os_type = '*' or prima_citta_stato_cod = '*' or prima_citta_cod = '*'''''
dataset_csv_errors = sqldf(q, globals())

dataset_csv = dataset_csv[~dataset_csv.user_id.isin(dataset_csv_errors["user_id"])]
```

2. Data Quality Checks & Data Cleaning

- Perform primary key checks

```
#primary key checks on meta_browser
q = "SELECT id, count(*) FROM meta_browser group by id having count(*)>1"
meta_browser_pk_check = sqldf(q, globals())
pk_checks.append(meta_browser_pk_check)

#primary key checks on meta_device
q = "SELECT id, count(*) FROM meta_device group by id having count(*)>1"
meta_device_pk_check = sqldf(q, globals())
pk_checks.append(meta_device_pk_check)

#primary key checks on meta_os
q = "SELECT id, count(*) FROM meta_os group by id having count(*)>1"
meta_os_pk_check = sqldf(q, globals())
pk_checks.append(meta_os_pk_check)

#primary key checks on dataset_csv
q = "SELECT user_id, count(user_id) FROM dataset_csv group by user_id having count(*)>1"
dataset_csv_pk_check = sqldf(q, globals())
pk_checks.append(dataset_csv_pk_check)
```

- Perform foreign key checks

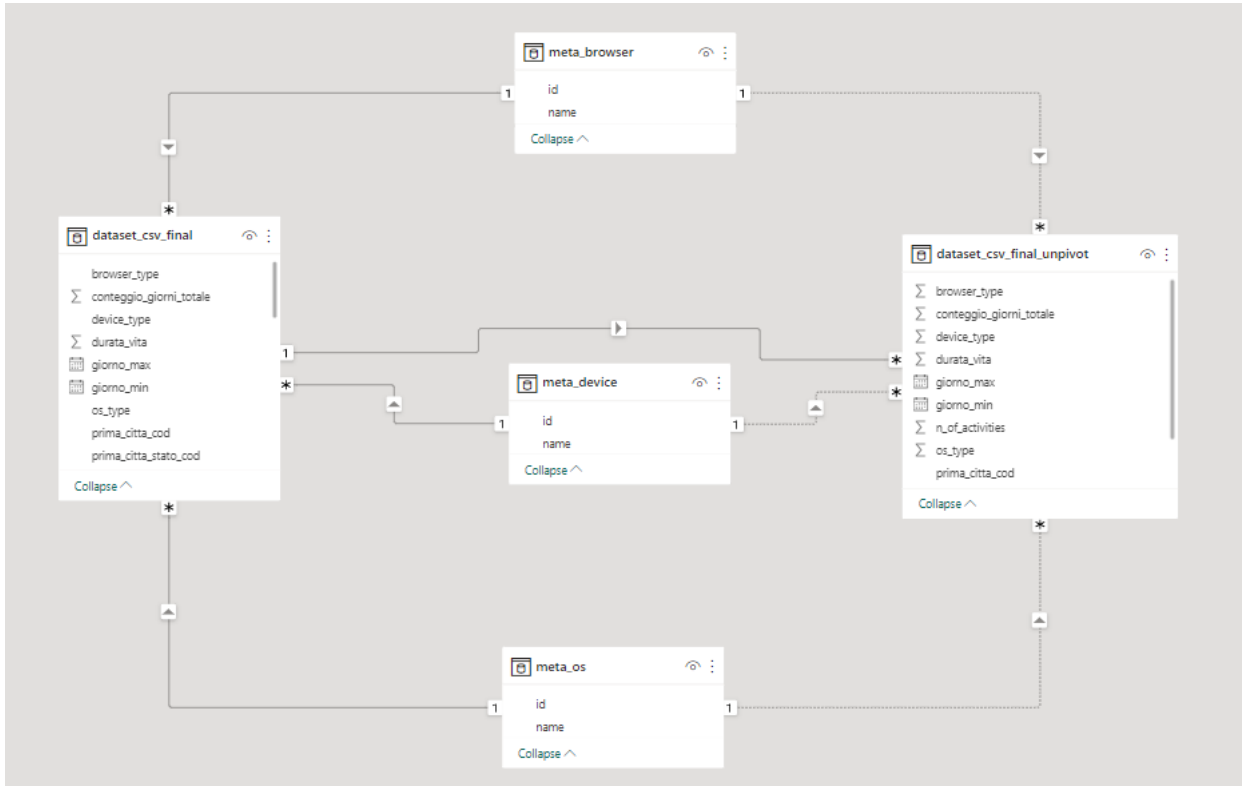
```
#foreign key checks on device_type
q = "select * FROM (SELECT * FROM dataset_csv as a left join meta_device as b  on a.device_type=b.id) where device_type is null"
meta_device_fk_check = sqldf(q, globals())
fk_checks.append(meta_device_fk_check)

#foreign key checks on browser_type
q = "select * FROM (SELECT * FROM dataset_csv as a left join meta_browser as b  on a.browser_type=b.id) where browser_type is null"
meta_browser_fk_check = sqldf(q, globals())
fk_checks.append(meta_browser_fk_check)

#foreign key checks on os_type
q = "select * FROM (SELECT * FROM dataset_csv as a left join meta_os as b  on a.os_type=b.id) where os_type is null"
meta_os_fk_check = sqldf(q, globals())
fk_checks.append(meta_os_fk_check)
```

```
dataset_csv[ os_type ] = datas
PK checks ended with success!!
FK checks ended with success!!
```

- Download final datasets

3. Dashboard Creation (1/5)

- Import datasets on power BI
- Create Relations and Data Model

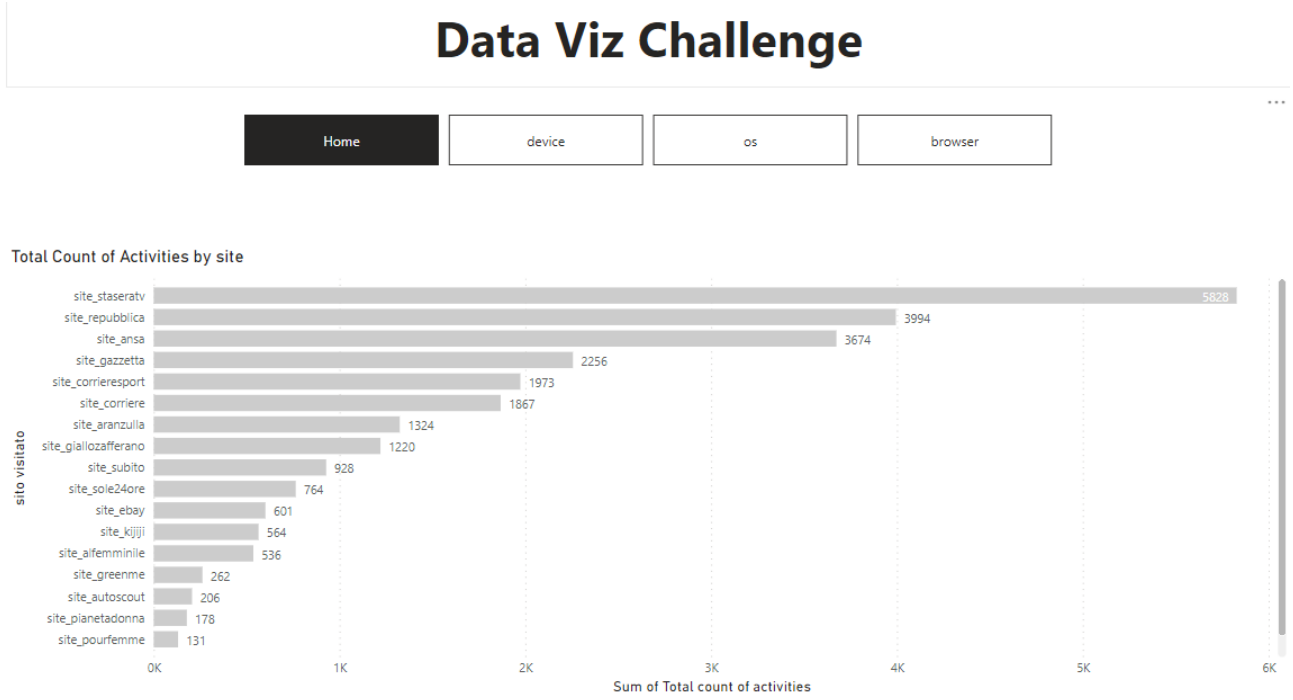


3. Dashboard Creation (2/5)

- Create dashboards structure: the dashboard will be divided into three dimension of analysis (Device, Os, Browser)
  - **Device**: We would like to know how users' activities are distributed through different devices. We would also know how active days and life duration changes from a device to another
  - **OS**: We would like to know how activities are distributed through different OS
  - **Browser**: We would like to know how activities are distributed through different browsers
- Create measures and calculated columns

3. Dashboard Creation (3/5)

- Create visualizations
  - Home Page

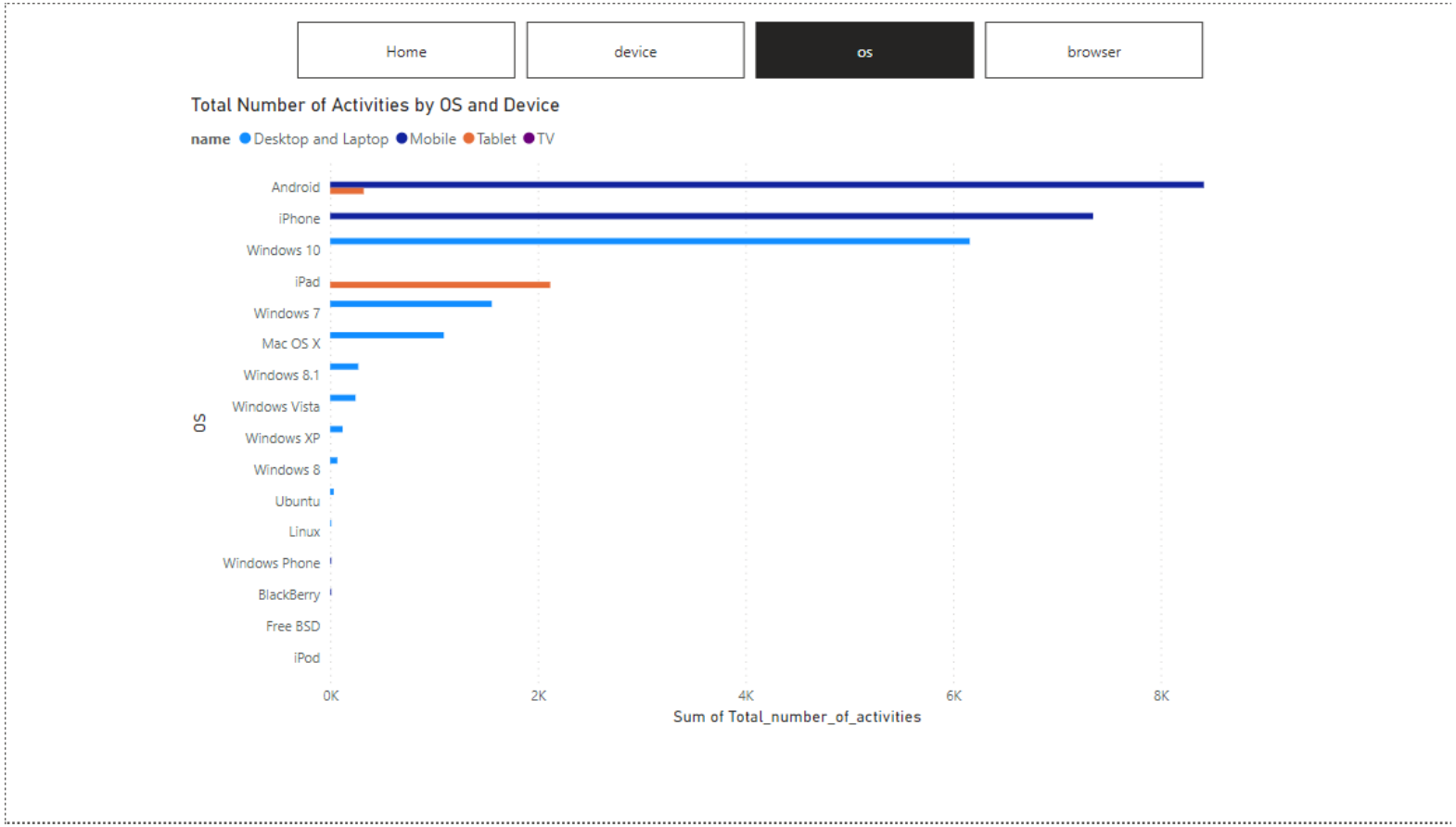


3. Dashboard Creation (4/5)

- Create visualizations
  - Device

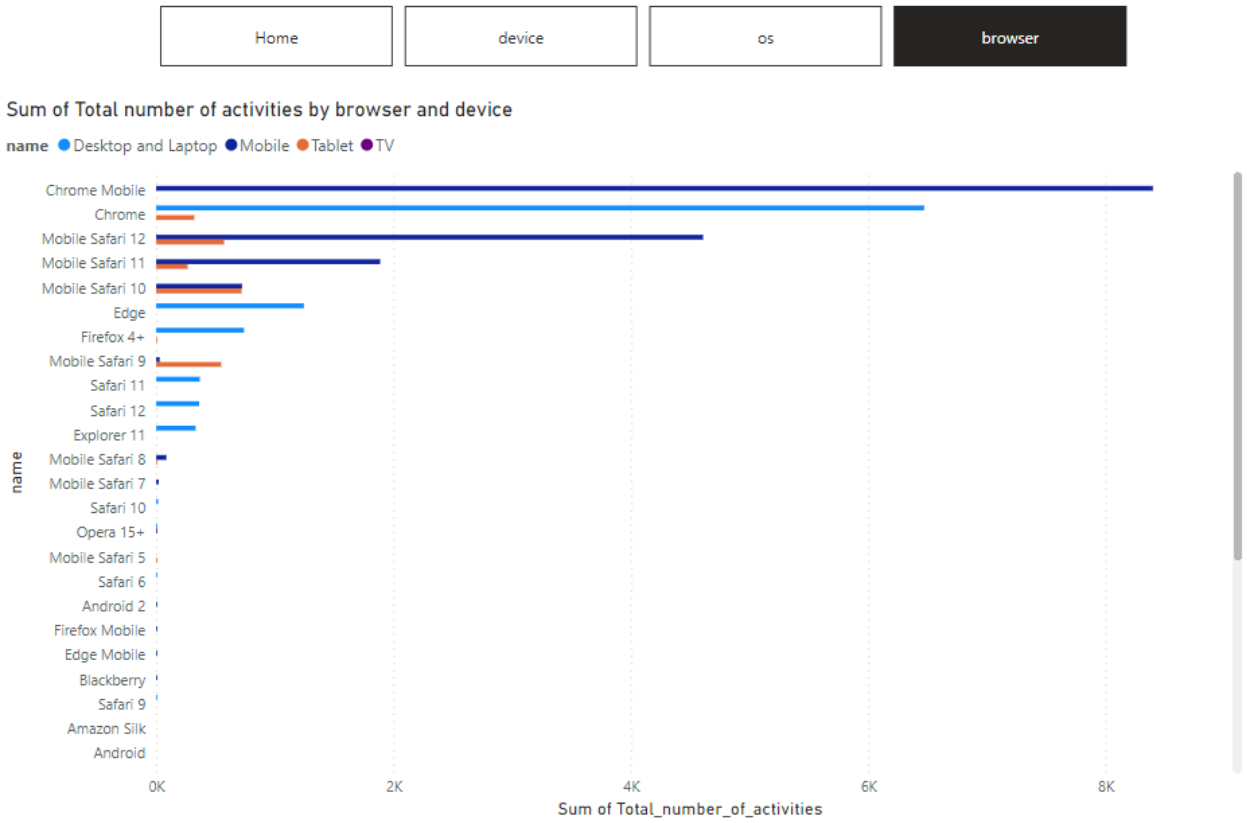
3. Dashboard Creation (5/5)

- Create visualizations
  - OS



3. Dashboard Creation (5/5)

- Create visualizations
  - Browser



4. Next Steps

- We can include these features in the second release:
  - Since we have the city detail for each user, we can create a visualization that distribute users/actions on a map.
  - We can create more KPIs for OS and Browser dimension.