# Visualization & Exploration of Airbnb User booking

For Airbnb data is an integral part of their business model, used to identify individuals without the annoyance of surveys and feedback. The Airbnb data we have access to is collected with the intention of gathering information about the customers; the tables used in this project were created for predicting the first booked destination for United States Airbnb users.

The importance of this data lies in the information it gives about the individual customers. In order to best predict an individual's travel location, one would expect the individual to make similar travel decisions to those who have analogous tendencies. The determinant of the usefulness of the data is the extent to which we can compare individuals' tendencies and group like-minded individuals.

The measure of a good prediction algorithm in this case would be the percentage of correctly identified "first booked destination" countries. If an algorithm is able to accurately predict the destination for 80% of individuals then algorithms are successful because of the potential impact. There is room for error due to the lack of detailed data that one often obtains from new users. With an algorithm success rate of 80%, individuals can have locations recommended that may impact the chances a person books a trip. Ultimately, any accuracy rate which increases company revenue, should be deemed successful, but those that create more repeat users will be the best.

Our data has a total of 16 attributes with 213,451 seperate records. The predicted value is "country_destination"

## Attribute Information

***id***

Identifies each unique user

***date_account_created***

The date an account was created

***timestamp_first_active***

Time stamp of the first activity of the user in absolute terms

***date_first_booking***

The date the first booked occurred

***gender***

The identification of an individual as Male, Female, or unidentified (unknown)

***age***

The years in which a person has been alive

***signup_method***

The method in which an account was created (Basic or Facebook)

***signup_flow***

the page from which the user came to sign up

***language***

international language preference

***affiliate_channel***

paid marketing type

***affiliate_provider***

location of marketing (Craiglist, Direct, Google, Other and Yahoo)

***first_affiliate_tracked***

first marketing interaction before sign up

***signup_app***

type of application used for sign up (Android, iOS, Moweb, Web)

***first_device_type***

The device used during sign-up

***first_browser***

The internet browser that was used to sign up

***country_destination***

Destination country for the account's first booking

For more information on the data set can be found at [https://www.kaggle.com/c/airbnb-recruiting-new-user-bookings/data (https://www.kaggle.com/c/airbnb-recruiting-new-user-bookings/data)](https://www.kaggle.com/c/airbnb-recruiting-new-user-bookings/data)

# Verify Data Quality

As mentioned in the previous section, the attributes Age and Gender have many missing values. Missing values for Age are noted as null, while missing values for Gender are noted as '-unknown-'.

In [2]:
```
# Import mathematical libraries for Python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Draw inline
%matplotlib inline

# Set figure aesthetics
sns.set_style("white", {'ytick.major.size': 10.0})
sns.set_context("poster", font_scale=1.1)

# Create our dataframes
train_users = pd.read_csv('/Users/kareemwilliams/DATAMINING/train_users_
2.csv')
test_users = pd.read_csv('/Users/kareemwilliams/DATAMINING/test_users.cs
v')
```

In [3]:
```
print 'We have', train_users.shape[0], 'users in the training set.'
```

```
We have 213451 users in the training set.
```

In [4]:
```
# Merge our 2 dataframes, train_users and test_users
#users = pd.concat((train_users, test_users), axis=0, ignore_index=True)

train_users.head()
```

Out[4]:

| | id | date_account_created | timestamp_first_active | date_first_bookir |
|---|---|---|---|---|
| **0** | gxn3p5htnn | 2010-06-28 | 20090319043255 | NaN |
| **1** | 820tgsjxq7 | 2011-05-25 | 20090523174809 | NaN |
| **2** | 4ft3gnwmtx | 2010-09-28 | 20090609231247 | 2010-08-02 |
| **3** | bjjt8pjhuk | 2011-12-05 | 20091031060129 | 2012-09-08 |
| **4** | 87mebub9p4 | 2010-09-14 | 20091208061105 | 2010-02-18 |

In [5]:

```
# Replace null with NaN for Age attribute
train_users.age.replace(np.nan, 0, inplace=True)
train_users['age'] = train_users['age'].astype('Int64')
train_users.age.replace(0, np.nan, inplace=True)

# Delete columns that we will not use
del train_users['first_affiliate_tracked']
del train_users['affiliate_channel']
del train_users['timestamp_first_active']
del train_users['id']
del train_users['language']

train_users.head()
```

Out[5]:

| | date_account_created | date_first_booking | gender | age | signup_method | si |
|---|---|---|---|---|---|---|
| 0 | 2010-06-28 | NaN | -unknown- | NaN | facebook | 0 |
| 1 | 2011-05-25 | NaN | MALE | 38 | facebook | 0 |
| 2 | 2010-09-28 | 2010-08-02 | FEMALE | 56 | basic | 3 |
| 3 | 2011-12-05 | 2012-09-08 | FEMALE | 42 | facebook | 0 |
| 4 | 2010-09-14 | 2010-02-18 | -unknown- | 41 | basic | 0 |

In [6]:

```
# Create Age Bucket variable
train_users['age_bucket'] = pd.cut(train_users.age,[0,9,19,29,39,49,59,6
9,79,89,99,1e6],3,labels=['0-9','10-19','20-29','30-39','40-49','50-5
9','60-69','70-79','80-89','90-99','100+'])

age_3039 = sum(train_users['age_bucket'] == '30-39')
total_age = train_users['age_bucket'].count()

print 'About', int(float(age_3039)/total_age * 100) ,'% our users are be
tween the ages of 30 and 39' + '\n'
print 'Here are some statistics about the Age Bucket attribute:'
print train_users.age_bucket.describe()
```

```
About 37 % our users are between the ages of 30 and 39

Here are some statistics about the Age Bucket attribute:
count     125461
unique        11
top        30-39
freq       47570
Name: age_bucket, dtype: object
```

In [7]:
```
# Replace '-unknown-' with NaN for Gender attribute
train_users.gender.replace('-unknown-', np.nan, inplace=True)
#users_nan = (users.isnull().sum() / users.shape[0]) * 100
#users_nan[users_nan > 0].drop('country_destination')

train_users.head(10)
```

Out[7]:

| | date_account_created | date_first_booking | gender | age | signup_method | si |
|---|---|---|---|---|---|---|
| 0 | 2010-06-28 | NaN | NaN | NaN | facebook | 0 |
| 1 | 2011-05-25 | NaN | MALE | 38 | facebook | 0 |
| 2 | 2010-09-28 | 2010-08-02 | FEMALE | 56 | basic | 3 |
| 3 | 2011-12-05 | 2012-09-08 | FEMALE | 42 | facebook | 0 |
| 4 | 2010-09-14 | 2010-02-18 | NaN | 41 | basic | 0 |
| 5 | 2010-01-01 | 2010-01-02 | NaN | NaN | basic | 0 |
| 6 | 2010-01-02 | 2010-01-05 | FEMALE | 46 | basic | 0 |
| 7 | 2010-01-03 | 2010-01-13 | FEMALE | 47 | basic | 0 |
| 8 | 2010-01-04 | 2010-07-29 | FEMALE | 50 | basic | 0 |
| 9 | 2010-01-04 | 2010-01-04 | NaN | 46 | basic | 0 |

In [8]:
```
# Look at Date First Booking attribute
print 'We have', int((float(train_users.date_first_booking.isnull().sum
()) / train_users.shape[0]) * 100), '% of missing values at date_first_b
ooking in the training data'
print '\n' + 'Here are some statistics about the Date First Booking attr
ibute:'
train_users.date_first_booking.describe()
```

```
We have 58 % of missing values at date_first_booking in the training da
ta

Here are some statistics about the Date First Booking attribute:
```

Out[8]:
```
count          88908
unique          1976
top       2014-05-22
freq             248
Name: date_first_booking, dtype: object
```

In [9]:
```
# Look at Age attribute
print 'Here are some statistics about the Age attribute:'
print train_users.age.describe()
```

```
Here are some statistics about the Age attribute:
count    125461.000000
mean         49.668335
std         155.666612
min           1.000000
25%          28.000000
50%          34.000000
75%          43.000000
max        2014.000000
Name: age, dtype: float64
```

In [10]:
```
print 'Users above 99:'
print train_users[train_users.age > 99]['age'].describe()
```

```
Users above 99:
count    2371.000000
mean      731.693800
std       895.193534
min       100.000000
25%       105.000000
50%       105.000000
75%      2014.000000
max      2014.000000
Name: age, dtype: float64
```

In [11]:
```
print 'Users below 13:'
print train_users[train_users.age < 13]['age'].describe()
```

```
Users below 13:
count    57.000000
mean      4.438596
std       1.195491
min       1.000000
25%       5.000000
50%       5.000000
75%       5.000000
max       5.000000
Name: age, dtype: float64
```

In [12]:
```
# Set ages greater than 99 or less then 13 to NaN
train_users.loc[train_users.age > 99, 'age'] = np.nan
train_users.loc[train_users.age < 13, 'age'] = np.nan
```

In [13]:
```
# Change to data type of our Categorical Features
categorical_features = [
    'affiliate_provider',
    'country_destination',
    'first_browser',
    'first_device_type',
    'gender',
    'signup_app',
    'signup_method'
]

for categorical_feature in categorical_features:
    train_users[categorical_feature] = train_users[categorical_feature].
astype('category')
```

In [14]:
```
# Change our dates attributes from object to datetime
train_users['date_account_created'] = pd.to_datetime(train_users['date_a
ccount_created'])
train_users['date_first_booking'] = pd.to_datetime(train_users['date_fir
st_booking'])
train_users['book_create_diff'] = train_users['date_first_booking'] - tr
ain_users['date_account_created']

print train_users.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 213451 entries, 0 to 213450
Data columns (total 13 columns):
date_account_created    213451 non-null datetime64[ns]
date_first_booking      88908 non-null datetime64[ns]
gender                  117763 non-null category
age                     123033 non-null float64
signup_method           213451 non-null category
signup_flow             213451 non-null int64
affiliate_provider      213451 non-null category
signup_app              213451 non-null category
first_device_type       213451 non-null category
first_browser           213451 non-null category
country_destination     213451 non-null category
age_bucket              125461 non-null category
book_create_diff        88908 non-null timedelta64[ns]
dtypes: category(8), datetime64[ns](2), float64(1), int64(1), timedelta
64[ns](1)
memory usage: 11.4 MB
None
```

# Simple Statistics

As mentioned earlier, the dataset that we used was primarily categorical in nature. As a result, we performed simple statistics for our two continuous variables: Age and Signup Flow

In [15]:
```
train_users.age.describe()
```

Out[15]:
```
count    123033.000000
mean         36.545805
std          11.655409
min          15.000000
25%          28.000000
50%          34.000000
75%          42.000000
max          99.000000
Name: age, dtype: float64
```

In [16]:
```
train_users.signup_flow.describe()
```

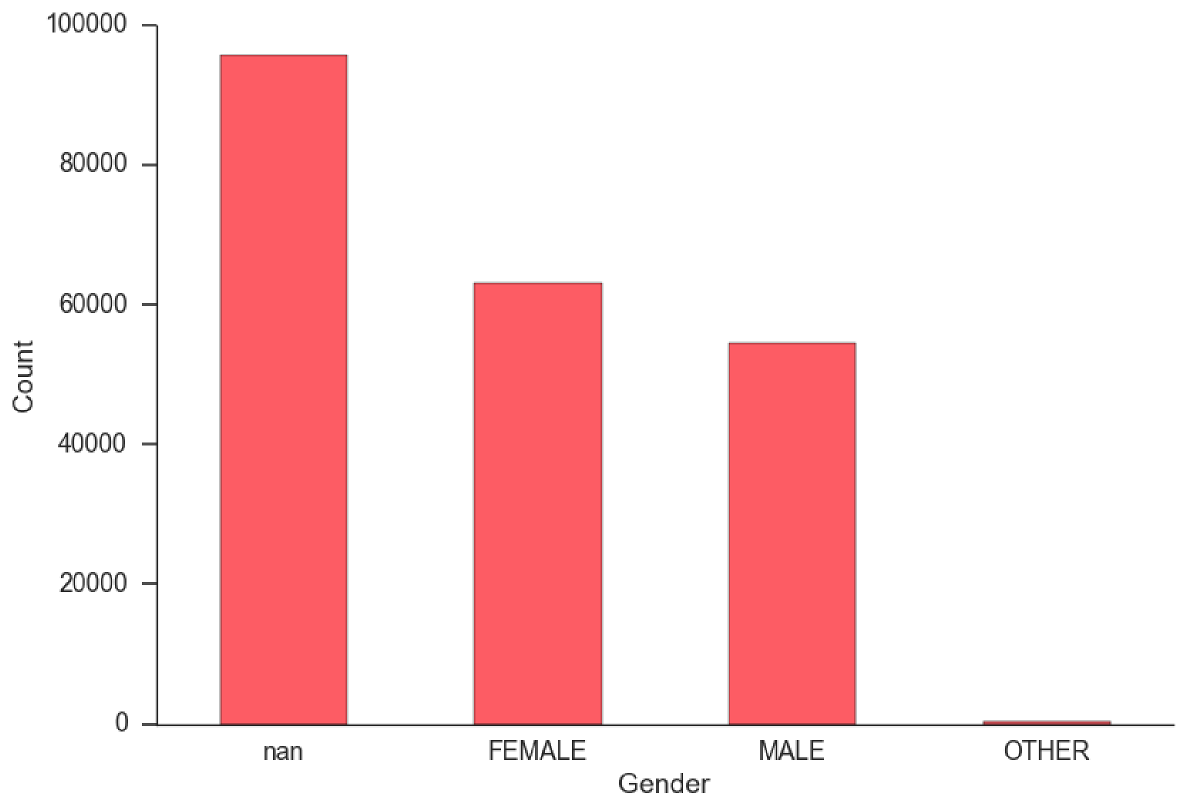
Out[16]:
```
count    213451.000000
mean          3.267387
std           7.637707
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max          25.000000
Name: signup_flow, dtype: float64
```

# Data Exploration

## Bar Graph: Gender Count

This bar graph depicts how many users we have in our data set by gender. There are 4 different gender categories such as NaN, Female, Male and Other. While we do not know the gender of many of our users, we do know that about 65,000 are Female and 55,000 are Male.

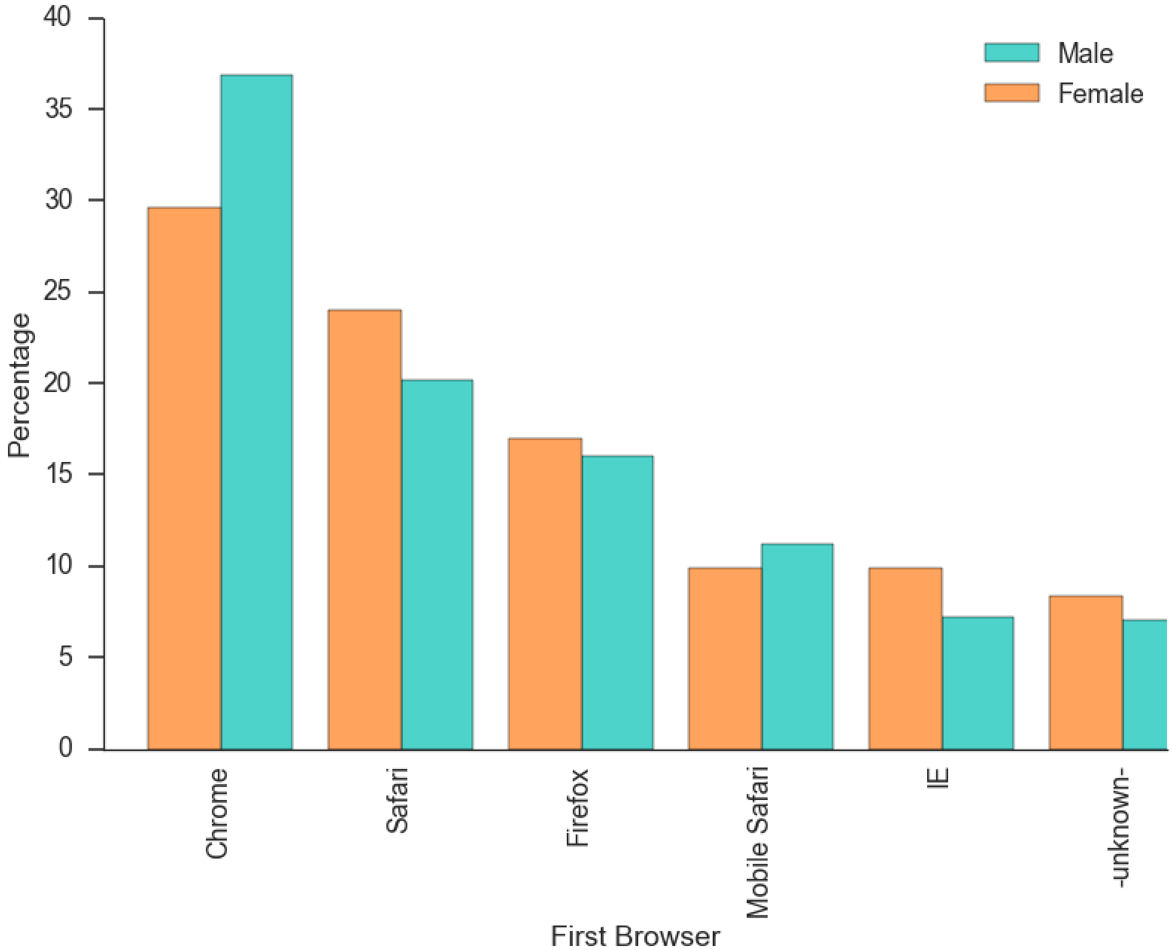
In [17]:
```
# Bar graph to show our gender counts
train_users.gender.value_counts(dropna=False).plot(kind='bar', color='#F
D5C64', rot=0)
plt.xlabel('Gender')
plt.ylabel('Count')
sns.despine()
plt.show()
```




## Bar Graph: Gender vs. First Browser

This bar graph illustrates the First Browser used by Males and Females when first being tracked by AirBnb. Since we have over 50 browsers in this category, we limited this bar graph to browsers that have at least 1% user usage.

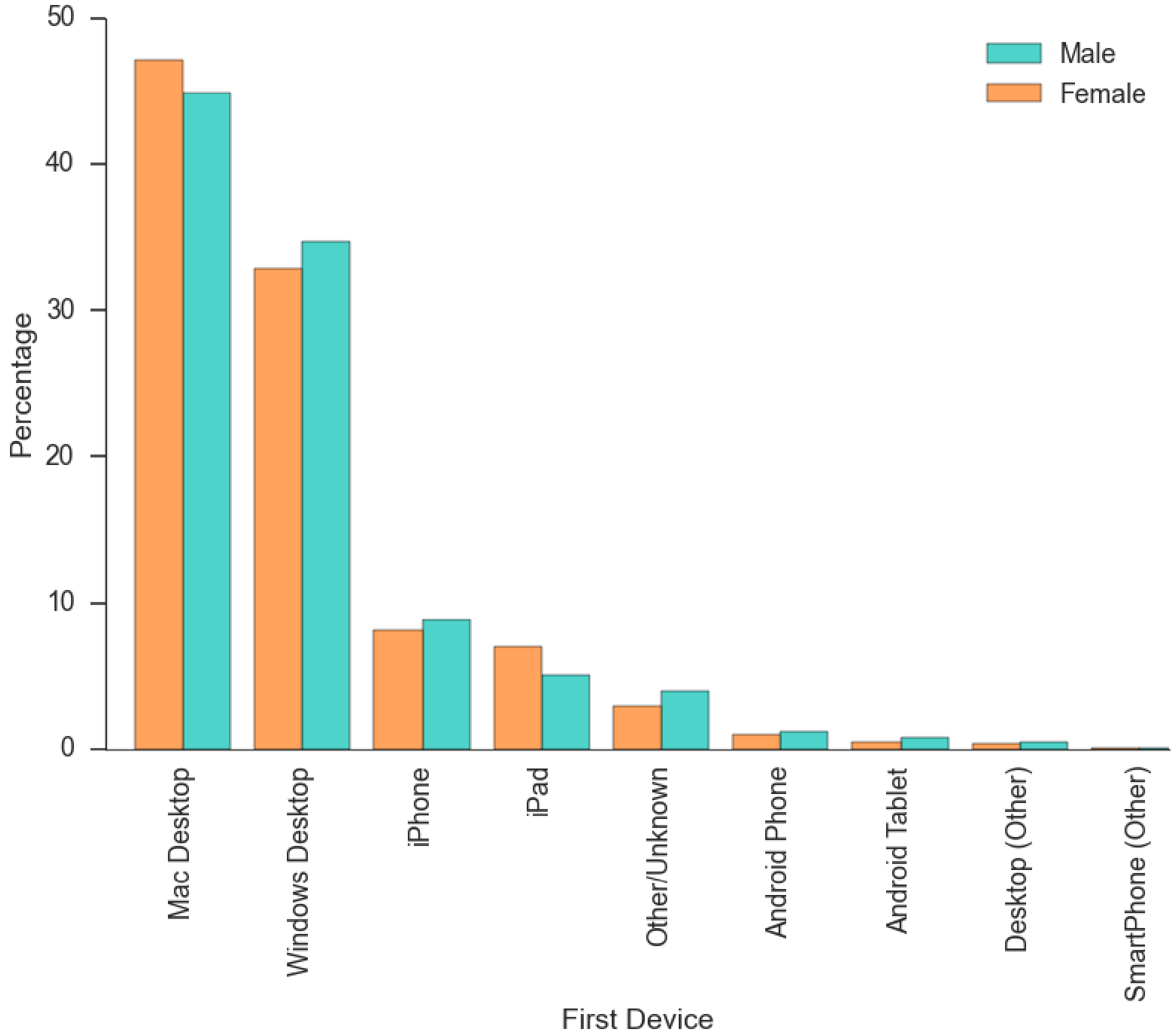
In [18]:
```
# Print Bar Graph based on Gender and Browsers
women = sum(train_users['gender'] == 'FEMALE')
men = sum(train_users['gender'] == 'MALE')

female_browsers = train_users.loc[train_users['gender'] == 'FEMALE', 'fi
rst_browser'].value_counts() / women * 100
male_browsers = train_users.loc[train_users['gender'] == 'MALE', 'first_
browser'].value_counts() / men * 100

female_browsers = female_browsers[female_browsers >= 1]
male_browsers = male_browsers[male_browsers >= 1]

# Bar width
width = 0.4

male_browsers.plot(kind='bar', width=width, color='#4DD3C9', position=0,
label='Male', rot=90)
female_browsers.plot(kind='bar', width=width, color='#FFA35D', position=
1, label='Female', rot=90)

plt.legend()
plt.xlabel('First Browser')
plt.ylabel('Percentage')

sns.despine()
plt.show()
```




## Bar Graph: Gender vs. First Device

This bar graph presents the First Device used by Males and Females when first (being tracked/signing up) for AirBnb. From this graph we learn that at least 75% of Males and Females signup for AirBnb using either a Mac Desktop or Windows Desktop.

In [19]:
```
female_device = train_users.loc[train_users['gender'] == 'FEMALE', 'firs
t_device_type'].value_counts() / women * 100
male_device = train_users.loc[train_users['gender'] == 'MALE', 'first_de
vice_type'].value_counts() / men * 100

# Bar width
width = 0.4

male_device.plot(kind='bar', width=width, color='#4DD3C9', position=0, l
abel='Male', rot=90)
female_device.plot(kind='bar', width=width, color='#FFA35D', position=1,
label='Female', rot=90)

plt.legend()
plt.xlabel('First Device')
plt.ylabel('Percentage')

sns.despine()
plt.show()
```

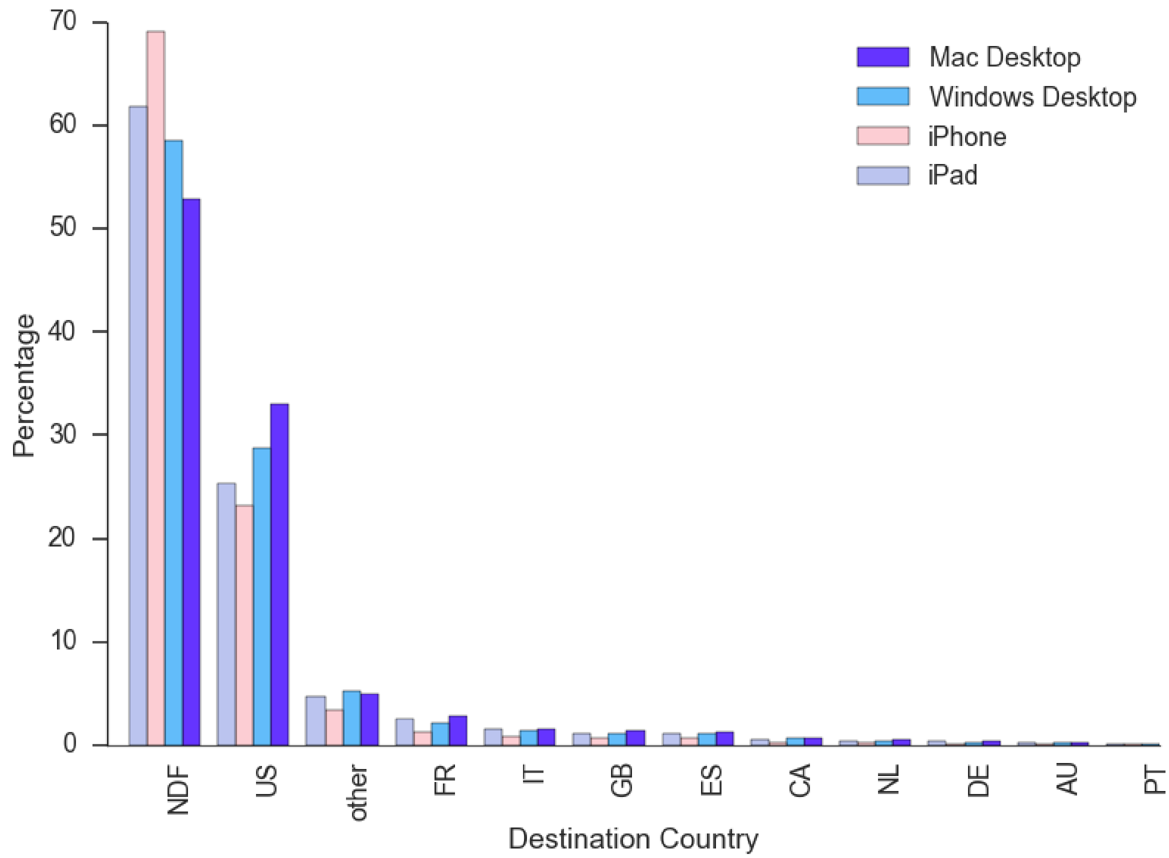



## Bar Graph: First Device vs. Destination Country

These graphs are used to get a better understanding of device types and destination countries. As seen in the other plots, the data is synonomous as all countries except for "other" are mainly Mac desktops, then Windows desktops, then Ipad, and Iphone.

In [20]:

```
MacDesk = sum(train_users['first_device_type'] == 'Mac Desktop')
WinDesk = sum(train_users['first_device_type'] == 'Windows Desktop')
iPhone = sum(train_users['first_device_type'] == 'iPhone')
iPad = sum(train_users['first_device_type'] == 'iPad')

# Print Bar Graph based on Gender and Browsers
MacDesk_Country = train_users.loc[train_users['first_device_type'] == 'M
ac Desktop', 'country_destination'].value_counts() / MacDesk * 100
WinDesk_Country = train_users.loc[train_users['first_device_type'] == 'W
indows Desktop', 'country_destination'].value_counts() / WinDesk * 100
iPhone_Country = train_users.loc[train_users['first_device_type'] == 'iP
hone', 'country_destination'].value_counts() / iPhone * 100
iPad_Country = train_users.loc[train_users['first_device_type'] == 'iPa
d', 'country_destination'].value_counts() / iPad * 100

# Bar width
width = 0.2

MacDesk_Country.plot(kind='bar', width=width, color='#6534ff', position=
0, label='Mac Desktop', rot=90)
WinDesk_Country.plot(kind='bar', width=width, color='#62bcfa', position=
1, label='Windows Desktop', rot=90)
iPhone_Country.plot(kind='bar', width=width, color='#fccdd3', position=
2, label='iPhone', rot=90)
iPad_Country.plot(kind='bar', width=width, color='#bbc4ef', position=3,
label='iPad', rot=90)

plt.legend()
plt.xlabel('Destination Country')
plt.ylabel('Percentage')

sns.despine()
plt.show()
```

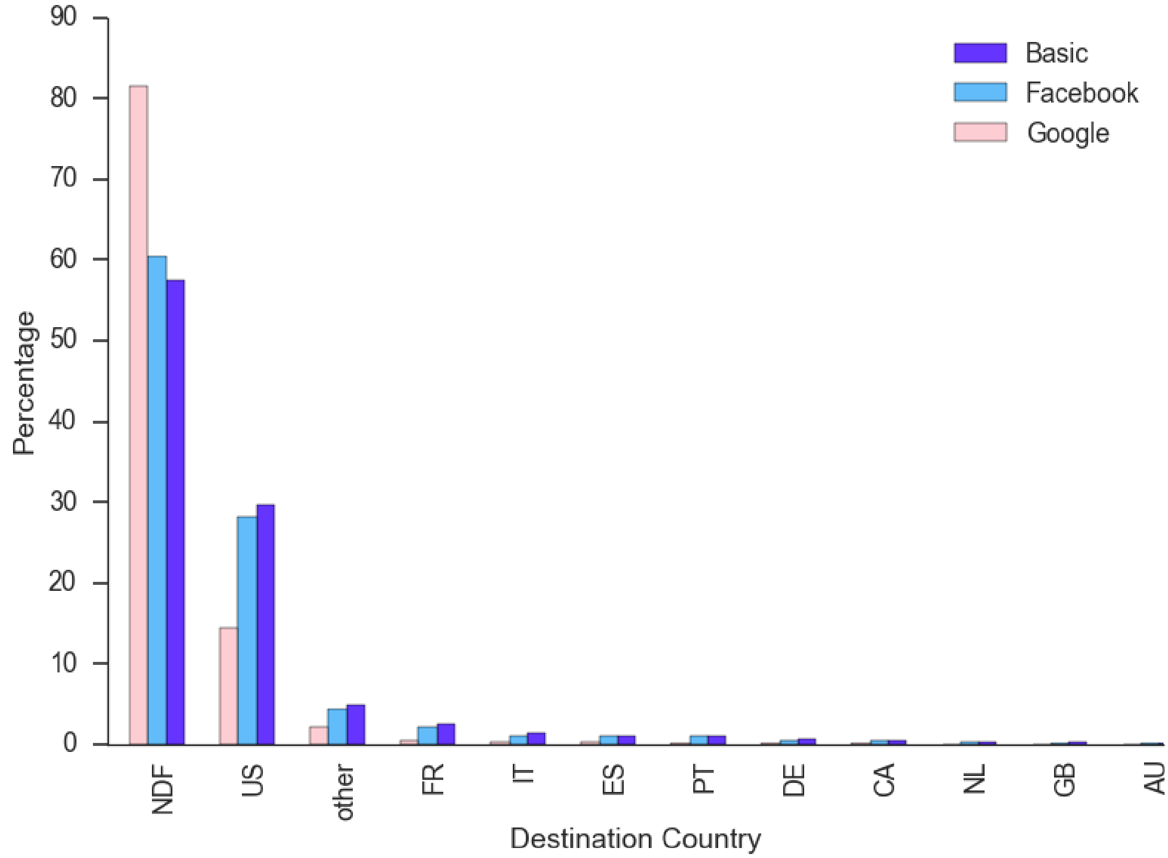



### Bar Graph: Signup Method vs. Destination Country

This bar graph depicts how the percentage of users who went to each country based on their signup method. We have 3 different categories for Signup Method; Basic, Facebook and Google. The graph shows that 58% of users who sign up through AirBnb, 60% of users who sign up through Facebook and 81% of users who signed up through Google did not choose a destination city.

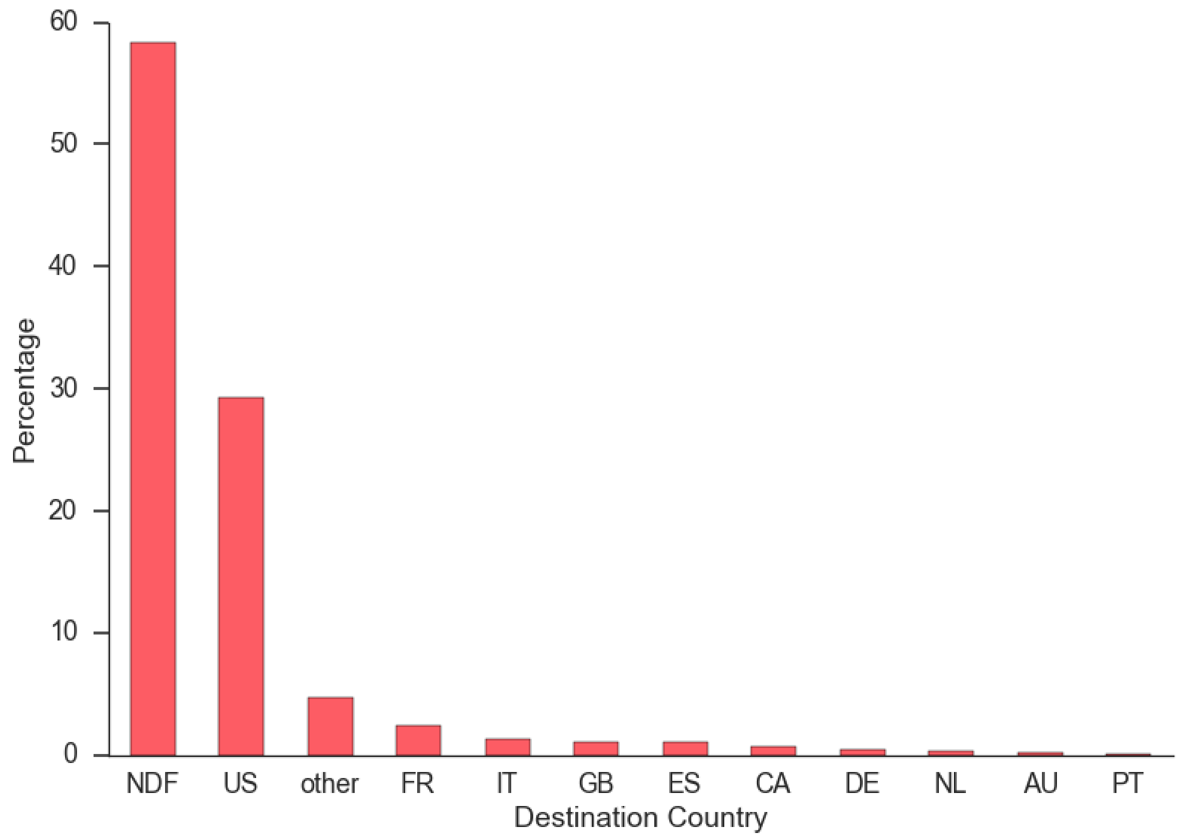
In [21]:

```
Basic = sum(train_users['signup_method'] == 'basic')
Facebook = sum(train_users['signup_method'] == 'facebook')
Google = sum(train_users['signup_method'] == 'google')

# Print Bar Graph based on Gender and Browsers
Basic_Country = train_users.loc[train_users['signup_method'] == 'basic',
'country_destination'].value_counts() / Basic * 100
Facebook_Country=  train_users.loc[train_users['signup_method'] == 'face
book', 'country_destination'].value_counts() / Facebook * 100
Google_Country = train_users.loc[train_users['signup_method'] == 'googl
e', 'country_destination'].value_counts() / Google * 100

# Bar width
width = 0.2

Basic_Country.plot(kind='bar', width=width, color='#6534ff', position=0,
label='Basic', rot=90)
Facebook_Country.plot(kind='bar', width=width, color='#62bcfa', position
=1, label='Facebook', rot=90)
Google_Country.plot(kind='bar', width=width, color='#fccdd3', position=
2, label='Google', rot=90)

plt.legend()
plt.xlabel('Destination Country')
plt.ylabel('Percentage')

sns.despine()
plt.show()
```



90
80
70
60
50
40
30
20
10
0
Percentage
Basic
Facebook
Google
NDF
US
other
FR
IT
ES
PT
DE
CA
NL
GB
AU
Destination Country

In [22]:
```
destination_percentage = train_users.country_destination.value_counts()
/ train_users.shape[0] * 100
destination_percentage.plot(kind='bar',color='#FD5C64', rot=0)
# Using seaborn can also be plotted
# sns.countplot(x="country_destination", data=users, order=list(users.co
untry_destination.value_counts().keys()))
plt.xlabel('Destination Country')
plt.ylabel('Percentage')
sns.despine()
```

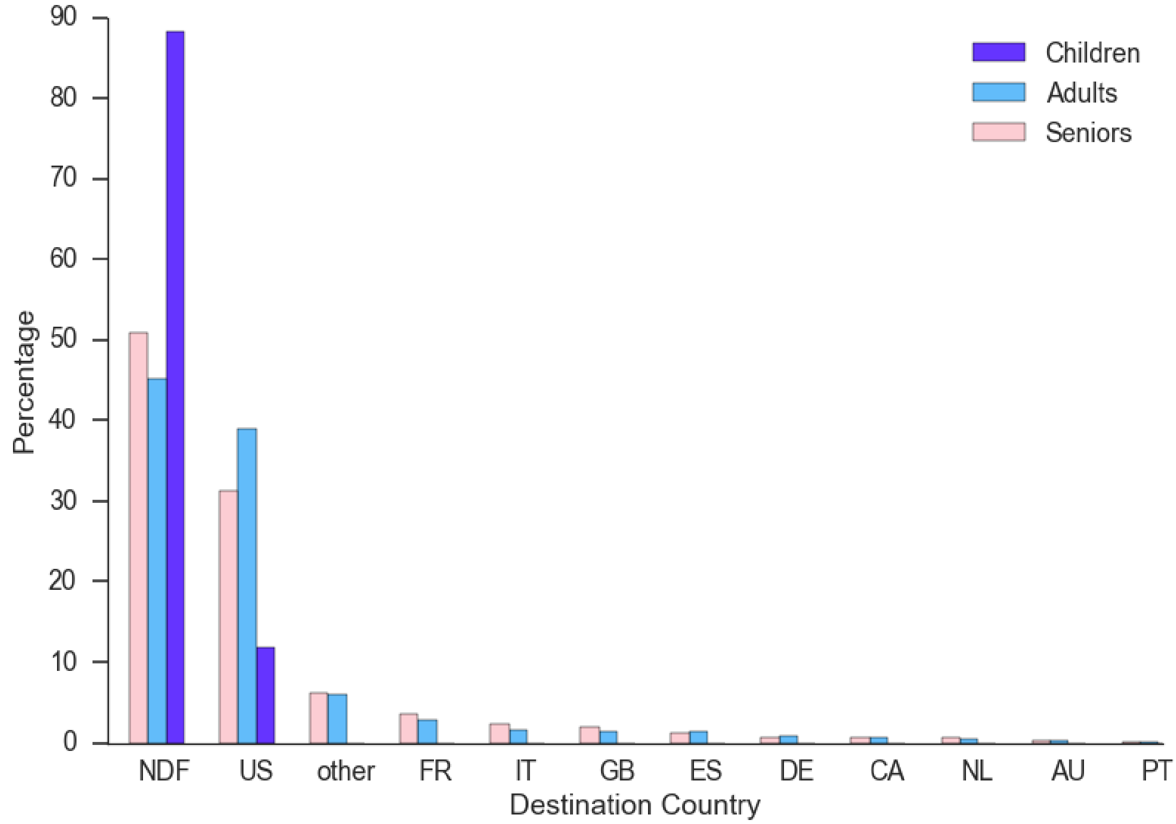



## Bar Graph: Age Range vs. Destination Country

Based on our calculated field Age Range, this bar graph presents the percentage of each Age Range that chose each Destination Country. From this graph, we observed that 88% of children do not have a first booking (NDF) and the other 12% either choose the US as their First Destination Country.

In [23]:

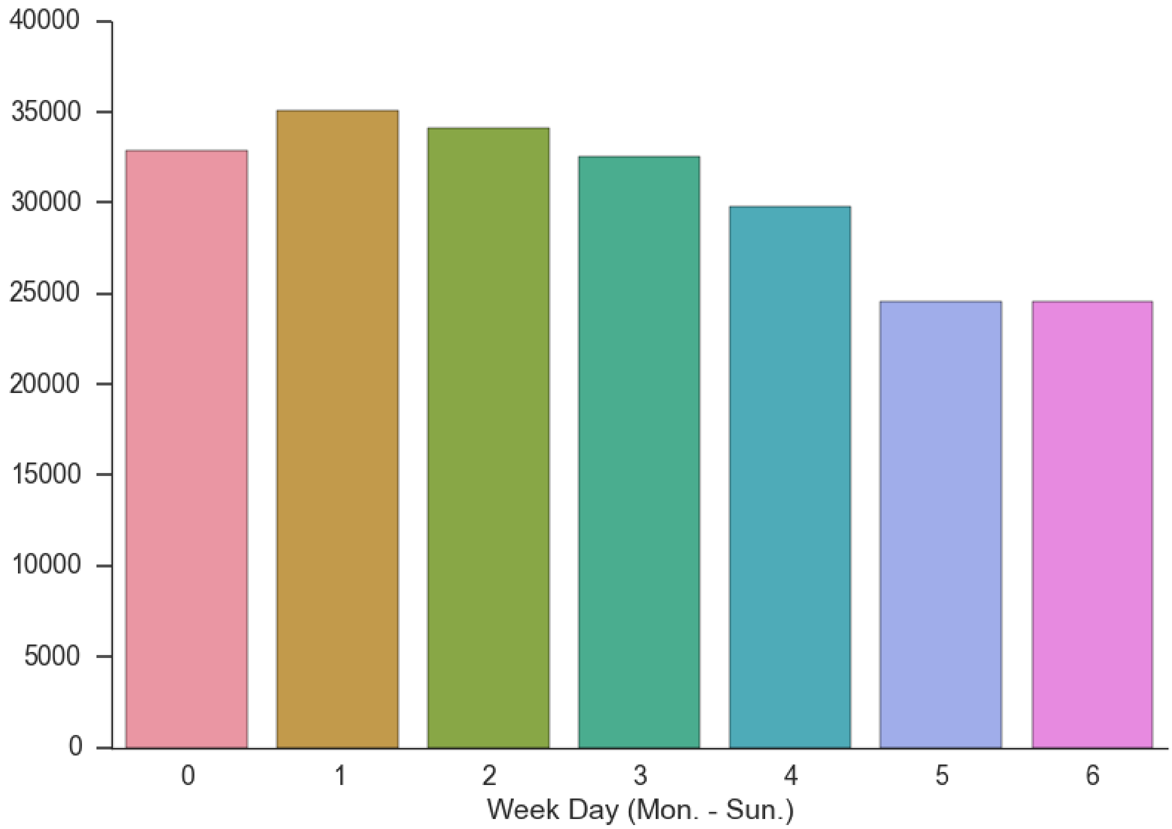
```
train_users['age_range'] = pd.cut(train_users.age,[0,16,65,1e6],3,labels
=['child','adult','senior'])
#print train_users.age_range.describe()

child = sum(train_users['age_range'] == 'child')
adult = sum(train_users['age_range'] == 'adult')
senior = sum(train_users['age_range'] == 'senior')

child_destinations = train_users.loc[train_users['age_range'] == 'chil
d', 'country_destination'].value_counts() / child * 100
adult_destinations = train_users.loc[train_users['age_range'] == 'adul
t', 'country_destination'].value_counts() / adult * 100
senior_destinations = train_users.loc[train_users['age_range'] == 'senio
r', 'country_destination'].value_counts() / senior * 100


child_destinations.plot(kind='bar', width=width, color='#6534ff', positi
on=0, label='Children', rot=0)
adult_destinations.plot(kind='bar', width=width, color='#62bcfa', positi
on=1, label='Adults', rot=0)
senior_destinations.plot(kind='bar', width=width, color='#fccdd3', posit
ion=2, label='Seniors', rot=0)

plt.legend()
plt.xlabel('Destination Country')
plt.ylabel('Percentage')

sns.despine()
plt.show()
```




### Bar Graph: User Signup per Weekday

This bar chart illustrates the number of users that signed up for Airbnb on each day of the week, Sunday through Saturday. From this chart, we can notice a spike of account creation on Monday, then a steady decrease throughout the rest of the week.

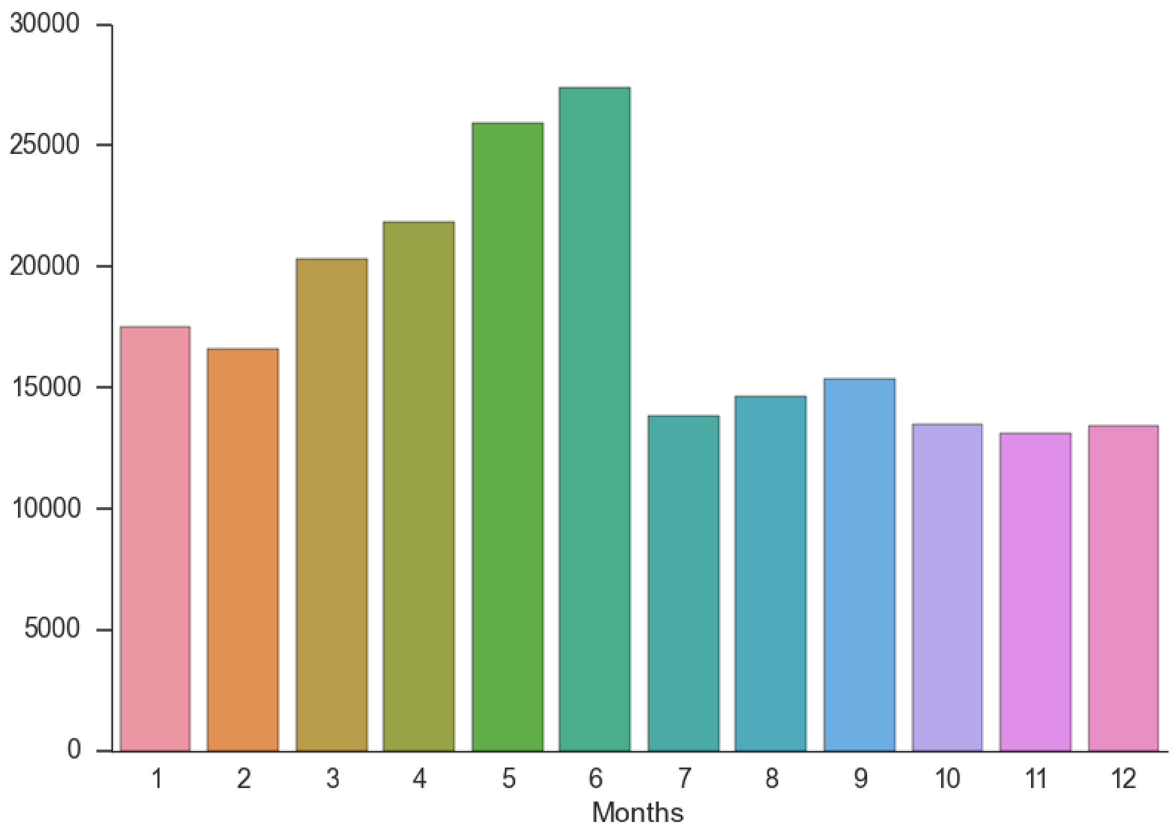
In [24]:
```
# Count of number of Users who signed up each day of the week
weekdays = []
for date in train_users.date_account_created:
    weekdays.append(date.weekday())

weekdays = pd.Series(weekdays)

sns.barplot(x = weekdays.value_counts().index, y=weekdays.value_counts
().values, order=range(0,7))
plt.xlabel('Week Day (Mon. - Sun.)')
sns.despine()
```




## Bar Graph: User Signup per Month

This bar chart illustrates the number of users that signed up for Airbnb on each month, January through December. From this chart, we notice an increase in user accounts during the first half of the month. During the 2nd half of the year, there is a sharp decline between June and July, then is steady throughout the rest of the year.

In [26]:
```
# Count of number of Users who signed up each month
months = []
for date in train_users.date_account_created:
    months.append(date.month)

months = pd.Series(months)

sns.barplot(x = months.value_counts().index, y=months.value_counts().val
ues, order=range(1,13))
plt.xlabel('Months')
sns.despine()
```

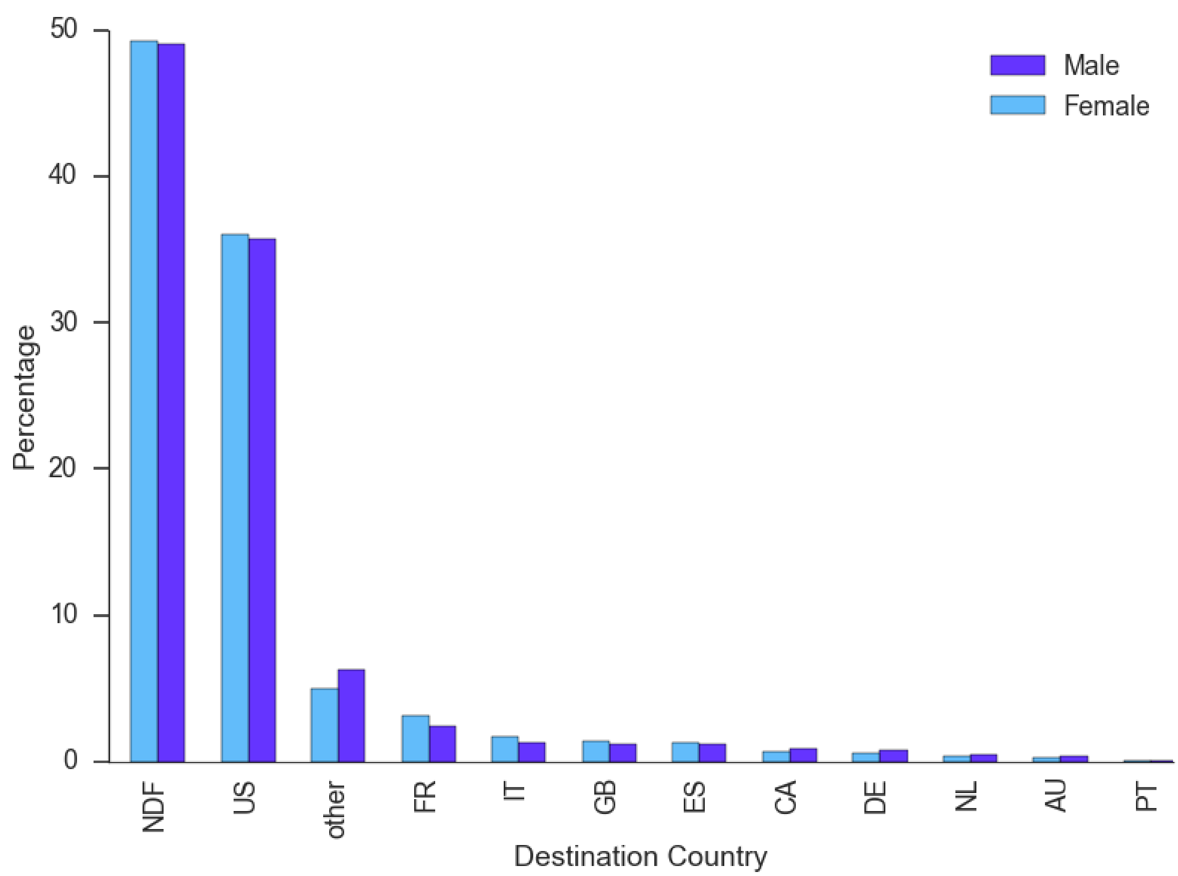



## Bar Graph: Gender vs. Destination Country

This bar chart presents the percentage of Males and Females that chose each country as their First Destination. We notice that 49% of both Males and Females do not choose First Destination Country. Also, about 35% of Males and Females choose the US as the First Destination Country.

In [27]:

```
# Print Bar Graph based on Gender and Destination
women = sum(train_users['gender'] == 'FEMALE')
men = sum(train_users['gender'] == 'MALE')

Male_Country = train_users.loc[train_users['gender'] == 'MALE', 'country
_destination'].value_counts() / men * 100
Female_Country=  train_users.loc[train_users['gender'] == 'FEMALE', 'cou
ntry_destination'].value_counts() / women * 100

# Bar width
width = 0.3

Male_Country.plot(kind='bar', width=width, color='#6534ff', position=0,
label='Male', rot=90)
Female_Country.plot(kind='bar', width=width, color='#62bcfa', position=
1, label='Female', rot=90)

plt.legend()
plt.xlabel('Destination Country')
plt.ylabel('Percentage')

sns.despine()
plt.show()
```

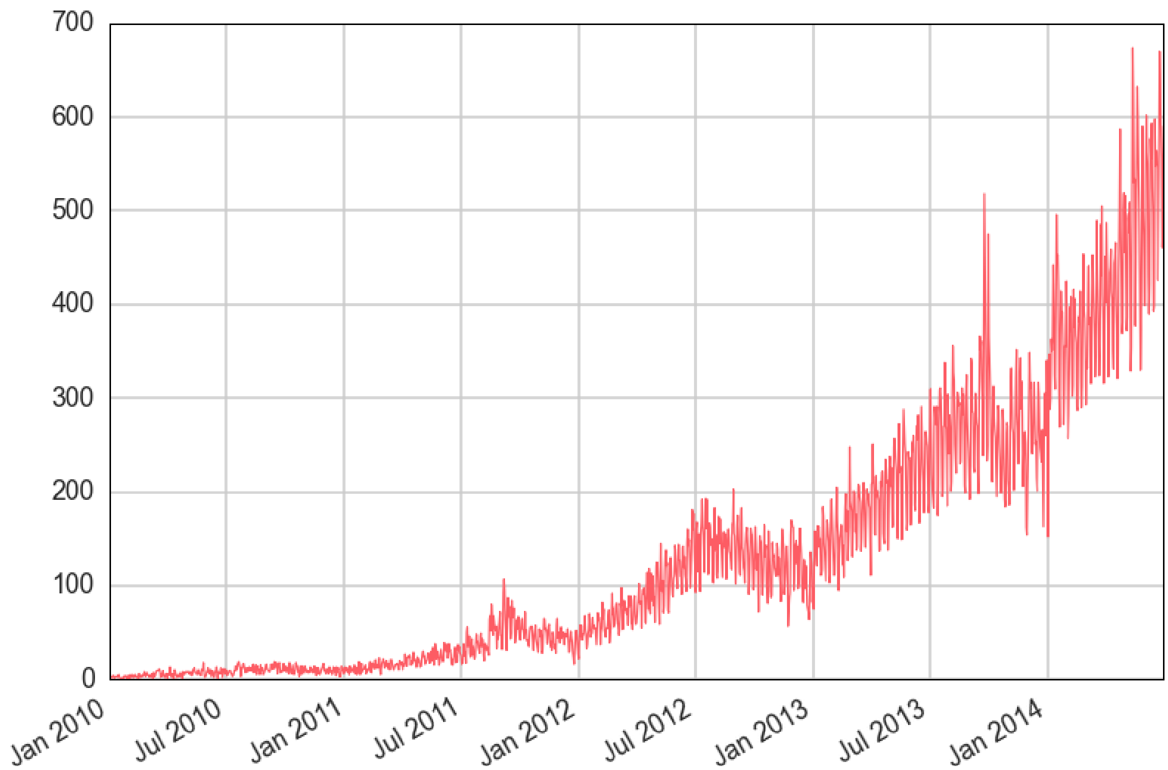


## Time Series

This series plots the signups for new Airbnb accounts on a yearly basis from 2010-2014

In [28]:
```
# Time series plot for signup dates
sns.set_style("whitegrid", {'axes.edgecolor': '0'})
sns.set_context("poster", font_scale=1.1)
train_users.date_account_created.value_counts().plot(kind='line',  linew
idth=1.2, color='#FD5C64')
```

Out[28]: <matplotlib.axes._subplots.AxesSubplot at 0x102e10a10>

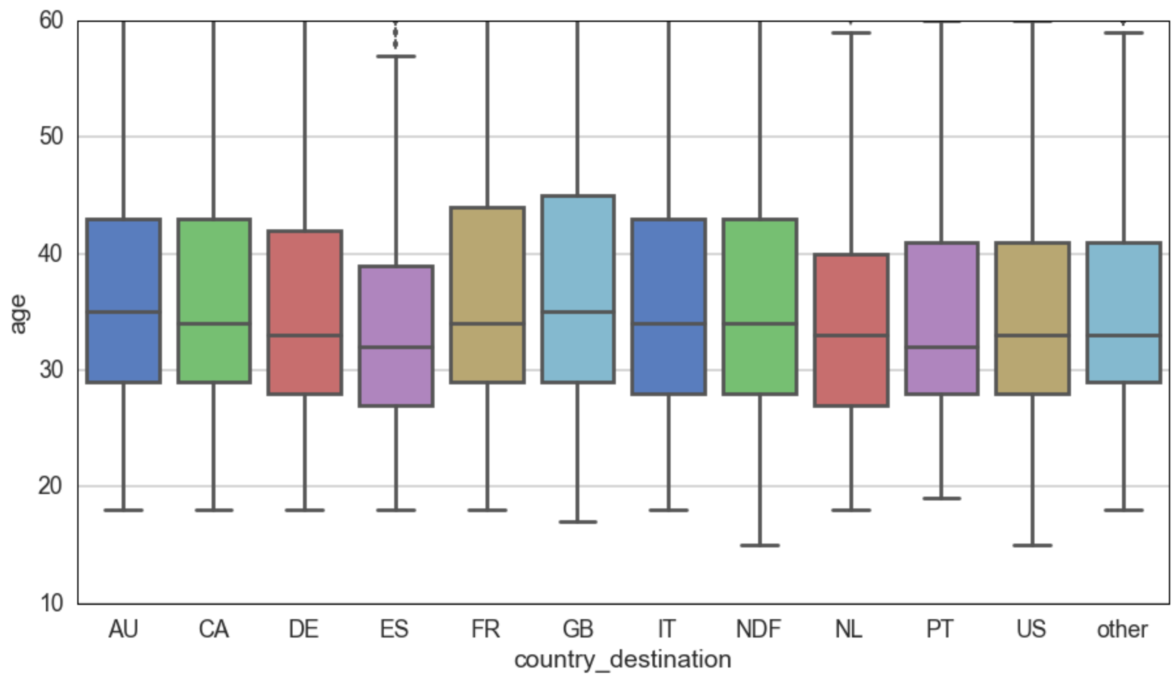



## Box & Whiskers Plot

This plot visually compares the ages of every destination country.

In [29]:
```
# plotting a box and whiskers plot that demonstrates Age to Travel Desti
nation
fig, ax = plt.subplots(nrows=1, ncols=1,figsize=(15, 8))
sns.boxplot(x='country_destination', y='age', data=train_users, palette
="muted", ax =ax)
ax.set_ylim([10, 60])
```

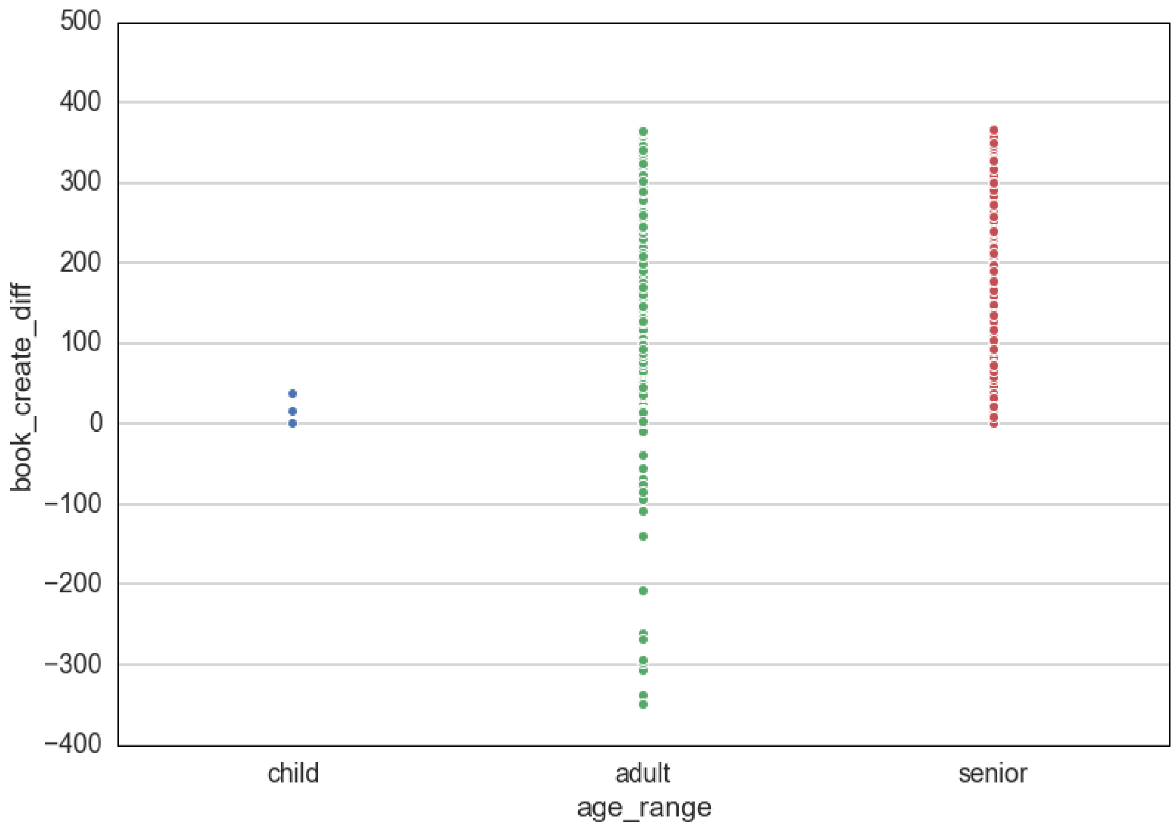
Out[29]: (10, 60)




## Scatterplot: Age Bucket vs Book Create Difference

This scatterplot represents the number of days it took a user to book a first destination from when they signed up based on our created variable Age Buckets.

In [30]:
```
# Scatterplot for Age Range vs Book Create Difference
sns.stripplot(x='age_range', y=train_users.book_create_diff.astype('time
delta64[D]'), data=train_users);
```

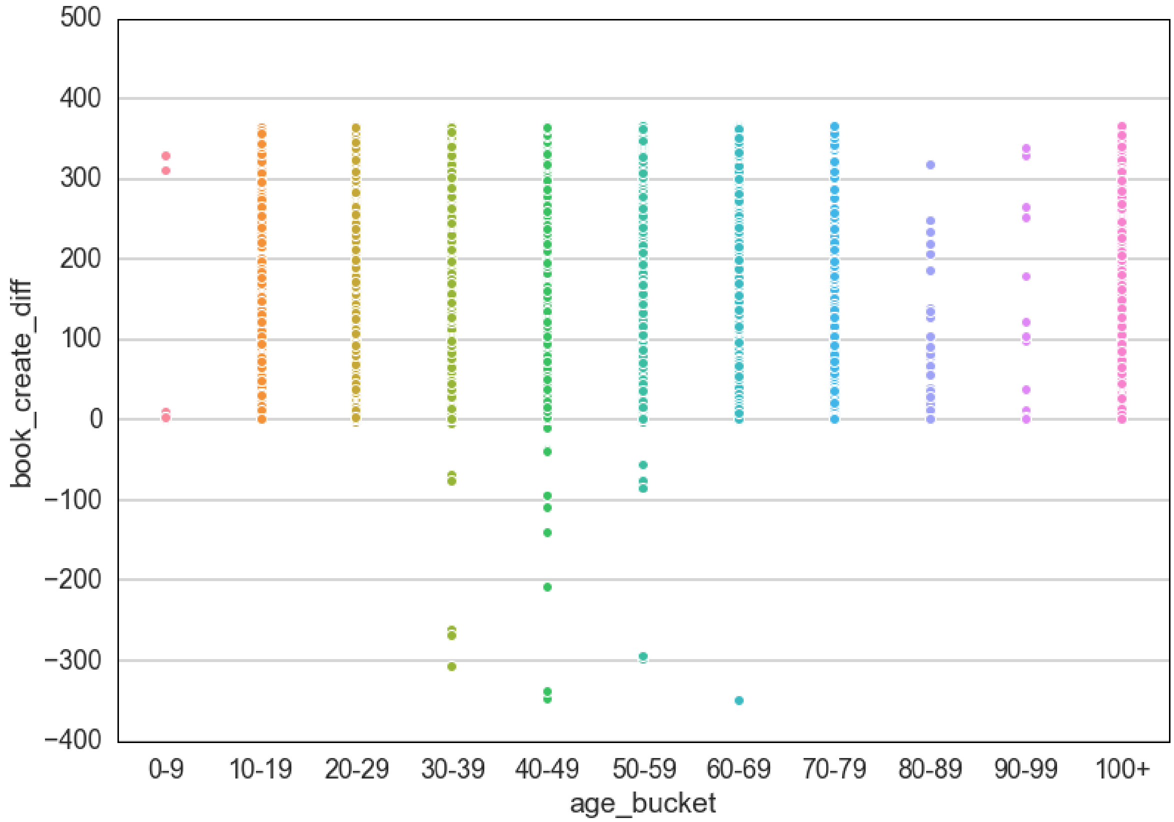



## Scatterplot: Age Range vs Book Create Difference

This scatterplot represents the number of days it took a user to book a first destination from when they signed up based on our created variable Age Range. We are still looking into why we receive negative values for some of the users. An interesting take away from this graph is that children, who do book a first destination, do so in the first 80 days after creating an account.

In [31]:

```
# Scatterplot for Age Bucket vs Book Create Difference
sns.stripplot(x='age_bucket', y=train_users.book_create_diff.astype('tim
edelta64[D]'), data=train_users);
```




# Interesting Features

The most interesting feature is the exponential growth of user signups for Airbnb. As seen in the time series above, the peak signup month is June for each new year. Users typically had the account creation traffic on Tuesdays and in June.

We found that the children accounts (ages =< 15) were one of the highest demographics for creating an account but not actually booking a room. Additionally, for children users, they only booked US destinations.

# Other Features that Could Be Added

We did add two feature. One was 'age bucket' and the other 'date-diff' (the difference of creation date minus date booked)

Additionally some features that could get added would be the other csv's attributes. We originally concatnated all of the csv files to have a master file, however we ran into some import errors. As a result we focused on the train_users dataset. Moving forwards with this project, it would definitely be worthwhile to properly add in the other features such as population, country data, latitude, longitude, etc.