In [2]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import os
```

In [3]:

```
activity = pd.read_csv('FitBit data.csv') # importing the dataset
```

In [8]:

```
activity.shape # checking the number of rows and columns in the dataset
```

Out[8]:

```
(457, 15)
```

In [9]:

```
activity.isnull().sum() # checking the number of missing values in the dataset
```

Out[9]:

```
Id                          0
ActivityDate                0
TotalSteps                  0
TotalDistance               0
TrackerDistance             0
LoggedActivitiesDistance    0
VeryActiveDistance          0
ModeratelyActiveDistance    0
LightActiveDistance         0
SedentaryActiveDistance     0
VeryActiveMinutes           0
FairlyActiveMinutes         0
LightlyActiveMinutes        0
SedentaryMinutes            0
Calories                    0
dtype: int64
```

In [11]:

```
activity.head(10) # seeing a sample of 10 rows from the dataset
```

Out[11]:

	Id	ActivityDate	TotalSteps	TotalDistance	TrackerDistance	LoggedActivitiesDista
0	1503960366	3/25/2016	11004	7.11	7.11	
1	1503960366	3/26/2016	17609	11.55	11.55	
2	1503960366	3/27/2016	12736	8.53	8.53	
3	1503960366	3/28/2016	13231	8.93	8.93	
4	1503960366	3/29/2016	12041	7.85	7.85	
5	1503960366	3/30/2016	10970	7.16	7.16	
6	1503960366	3/31/2016	12256	7.86	7.86	
7	1503960366	4/1/2016	12262	7.87	7.87	
8	1503960366	4/2/2016	11248	7.25	7.25	
9	1503960366	4/3/2016	10016	6.37	6.37	

In [14]:

```
activity1 = activity.copy() # copying the datset to activity1
```

In [15]:

```
activity1['ActivityDate'].unique() # checking out the unique activity dates in the datas
```

Out[15]:

```
array(['3/25/2016', '3/26/2016', '3/27/2016', '3/28/2016', '3/29/2016',
       '3/30/2016', '3/31/2016', '4/1/2016', '4/2/2016', '4/3/2016',
       '4/4/2016', '4/5/2016', '4/6/2016', '4/7/2016', '4/8/2016',
       '4/9/2016', '4/10/2016', '4/11/2016', '4/12/2016', '3/12/2016',
       '3/13/2016', '3/14/2016', '3/15/2016', '3/16/2016', '3/17/2016',
       '3/18/2016', '3/19/2016', '3/20/2016', '3/21/2016', '3/22/2016',
       '3/23/2016', '3/24/2016'], dtype=object)
```

In [13]:

```
activity1['ActivityDate'].head(10)  # cheking out the datset before transformation
```

Out[13]:

```
0    3/25/2016
1    3/26/2016
2    3/27/2016
3    3/28/2016
4    3/29/2016
5    3/30/2016
6    3/31/2016
7     4/1/2016
8     4/2/2016
9     4/3/2016
Name: ActivityDate, dtype: object
```

In [16]:

```
# adding the yearm month and date columns to the dataset
activity1['year'] = pd.DatetimeIndex(activity1['ActivityDate']).year
activity1['month'] = pd.DatetimeIndex(activity1['ActivityDate']).month
activity1['date'] = pd.DatetimeIndex(activity1['ActivityDate']).day
```

In [17]:

```
activity1.head(10) # cheking out the datset after transformation
```

Out[17]:

	Id	ActivityDate	TotalSteps	TotalDistance	TrackerDistance	LoggedActivitiesDista
0	1503960366	3/25/2016	11004	7.11	7.11	
1	1503960366	3/26/2016	17609	11.55	11.55	
2	1503960366	3/27/2016	12736	8.53	8.53	
3	1503960366	3/28/2016	13231	8.93	8.93	
4	1503960366	3/29/2016	12041	7.85	7.85	
5	1503960366	3/30/2016	10970	7.16	7.16	
6	1503960366	3/31/2016	12256	7.86	7.86	
7	1503960366	4/1/2016	12262	7.87	7.87	
8	1503960366	4/2/2016	11248	7.25	7.25	
9	1503960366	4/3/2016	10016	6.37	6.37	

In [19]:

```
activity1=activity1.drop(['TrackerDistance'],axis=1)  #dropping the TrackerDistance colu
```

In [20]:

```
activity1.head(200) # cheking out the first 200 rows of the datset after transformation
```

Out[20]:

	Id	ActivityDate	TotalSteps	TotalDistance	LoggedActivitiesDistance	VeryActive
0	1503960366	3/25/2016	11004	7.11	0.0	
1	1503960366	3/26/2016	17609	11.55	0.0	
2	1503960366	3/27/2016	12736	8.53	0.0	
3	1503960366	3/28/2016	13231	8.93	0.0	
4	1503960366	3/29/2016	12041	7.85	0.0	
5	1503960366	3/30/2016	10970	7.16	0.0	
6	1503960366	3/31/2016	12256	7.86	0.0	
7	1503960366	4/1/2016	12262	7.87	0.0	
8	1503960366	4/2/2016	11248	7.25	0.0	
9	1503960366	4/3/2016	10016	6.37	0.0	
10	1503960366	4/4/2016	14557	9.80	0.0	
11	1503960366	4/5/2016	14844	9.73	0.0	
12	1503960366	4/6/2016	11974	7.67	0.0	
13	1503960366	4/7/2016	10198	6.44	0.0	
14	1503960366	4/8/2016	12521	7.94	0.0	
15	1503960366	4/9/2016	12432	8.10	0.0	
16	1503960366	4/10/2016	10057	6.98	0.0	
17	1503960366	4/11/2016	10990	7.26	0.0	
18	1503960366	4/12/2016	224	0.14	0.0	
19	1624580081	3/25/2016	1810	1.18	0.0	
20	1624580081	3/26/2016	815	0.53	0.0	
21	1624580081	3/27/2016	1985	1.29	0.0	
22	1624580081	3/28/2016	1905	1.24	0.0	
23	1624580081	3/29/2016	1552	1.01	0.0	
24	1624580081	3/30/2016	1675	1.09	0.0	
25	1624580081	3/31/2016	4506	2.93	0.0	
26	1624580081	4/1/2016	9218	5.99	0.0	
27	1624580081	4/2/2016	1556	1.01	0.0	
28	1624580081	4/3/2016	2910	1.89	0.0	
29	1624580081	4/4/2016	18464	12.00	0.0	
...	...	...	...	...	...	
170	4020332650	3/17/2016	8940	6.41	0.0	
171	4020332650	3/18/2016	368	0.26	0.0	
172	4020332650	3/19/2016	5702	4.09	0.0	
173	4020332650	3/20/2016	10330	7.41	0.0	
174	4020332650	3/21/2016	8778	6.29	0.0	
175	4020332650	3/22/2016	6662	4.78	0.0	

	Id	ActivityDate	TotalSteps	TotalDistance	LoggedActivitiesDistance	VeryActive
176	4020332650	3/23/2016	6309	4.52	0.0	
177	4020332650	3/24/2016	1951	1.41	0.0	
178	4020332650	3/25/2016	5563	3.99	0.0	
179	4020332650	3/26/2016	4370	3.13	0.0	
180	4020332650	3/27/2016	7144	5.12	0.0	
181	4020332650	3/28/2016	2106	1.51	0.0	
182	4020332650	3/29/2016	4152	2.98	0.0	
183	4020332650	3/30/2016	5400	3.87	0.0	
184	4020332650	3/31/2016	7428	5.33	0.0	
185	4020332650	4/1/2016	5351	3.84	0.0	
186	4020332650	4/2/2016	4299	3.10	0.0	
187	4020332650	4/3/2016	6107	4.38	0.0	
188	4020332650	4/4/2016	6429	4.60	0.0	
189	4020332650	4/5/2016	6880	4.93	0.0	
190	4020332650	4/6/2016	7476	5.36	0.0	
191	4020332650	4/7/2016	6581	4.72	0.0	
192	4020332650	4/8/2016	10480	7.51	0.0	
193	4020332650	4/9/2016	7734	5.55	0.0	
194	4020332650	4/10/2016	5129	3.68	0.0	
195	4020332650	4/11/2016	2993	2.15	0.0	
196	4020332650	4/12/2016	8	0.01	0.0	
197	4057192912	3/12/2016	0	0.00	0.0	
198	4057192912	3/13/2016	0	0.00	0.0	
199	4057192912	3/14/2016	8433	6.23	0.0	

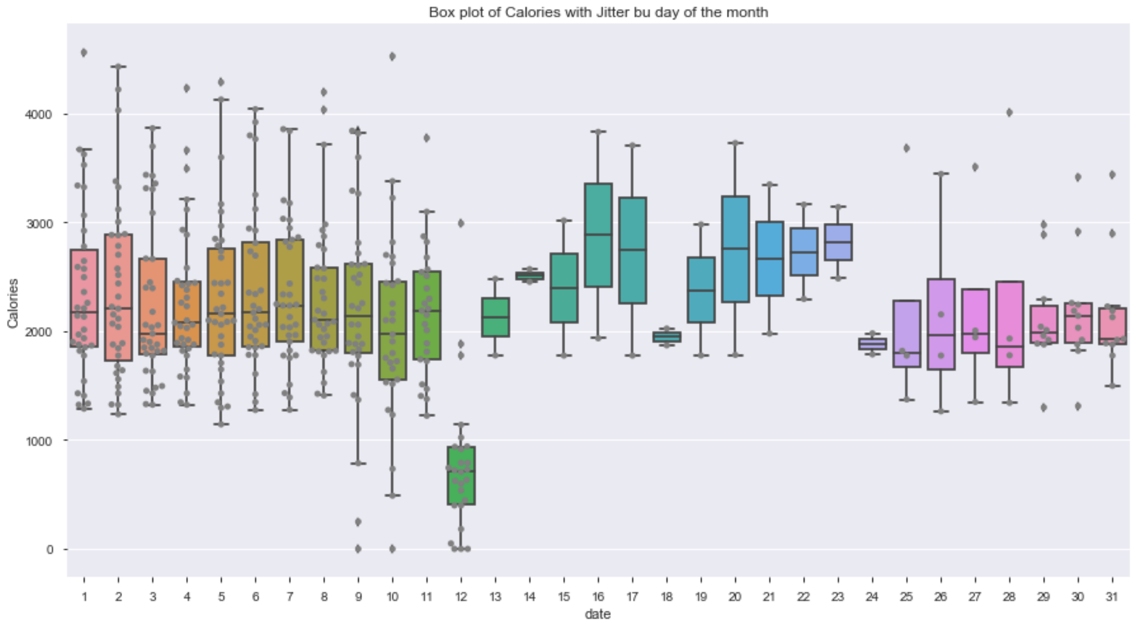
200 rows × 17 columns

In [21]:

```
### Groupby the day of the month and make a boxplot of calories burnt
import matplotlib.pyplot as plt
# figure size
plt.figure(figsize=(15,8))

# Usual boxplot
ax = sns.boxplot(x='date', y='Calories', data=activity1)

# Add jitter with the swarmplot function.
ax = sns.swarmplot(x='date', y='Calories', data=activity1, color="grey")

ax.set_title('Box plot of Calories with Jitter bu day of the month')
```

Out[21]:

```
Text(0.5, 1.0, 'Box plot of Calories with Jitter bu day of the month')
```



In [22]:

```
# converting the datatype to datetime
activity1['Week'] = pd.to_datetime(activity1.ActivityDate).dt.week
activity1['Year'] = pd.to_datetime(activity1.ActivityDate).dt.year
```

In [23]:

```
activity1.head()  # cheking out the datset after transformation
```

Out[23]:

	Id	ActivityDate	TotalSteps	TotalDistance	LoggedActivitiesDistance	VeryActiveD
0	1503960366	3/25/2016	11004	7.11	0.0	
1	1503960366	3/26/2016	17609	11.55	0.0	
2	1503960366	3/27/2016	12736	8.53	0.0	
3	1503960366	3/28/2016	13231	8.93	0.0	
4	1503960366	3/29/2016	12041	7.85	0.0	

In [25]:

```
activity1.ActivityDate.dtype # cheking the datatype of ActivityDate field
```

Out[25]:

```
dtype('O')
```

In [26]:

```
activity1['ActivityDate'] = pd.to_datetime(activity1['ActivityDate']) # converting it to
```

In [27]:

```
activity1['day'] = activity1['ActivityDate'].dt.weekday_name # converting the day of the
```

In [28]:

```
activity1.head(10) # cheking out the datset after transformation
```

Out[28]:

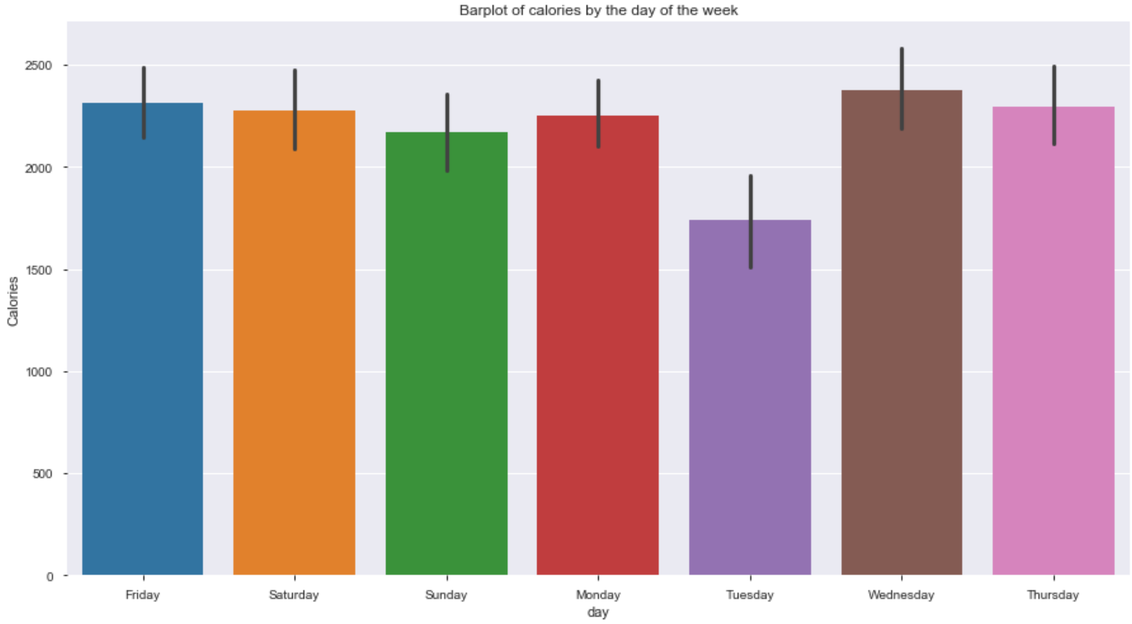
	Id	ActivityDate	TotalSteps	TotalDistance	LoggedActivitiesDistance	VeryActiveD
0	1503960366	2016-03-25	11004	7.11	0.0	
1	1503960366	2016-03-26	17609	11.55	0.0	
2	1503960366	2016-03-27	12736	8.53	0.0	
3	1503960366	2016-03-28	13231	8.93	0.0	
4	1503960366	2016-03-29	12041	7.85	0.0	
5	1503960366	2016-03-30	10970	7.16	0.0	
6	1503960366	2016-03-31	12256	7.86	0.0	
7	1503960366	2016-04-01	12262	7.87	0.0	
8	1503960366	2016-04-02	11248	7.25	0.0	
9	1503960366	2016-04-03	10016	6.37	0.0	

In [29]:

```
# figure size
plt.figure(figsize=(15,8))

# simple barplot
ax = sns.barplot(x='day', y='Calories',  data=activity1)

ax.set_title('Barplot of calories by the day of the week')
```

Out[29]:

```
Text(0.5, 1.0, 'Barplot of calories by the day of the week')
```

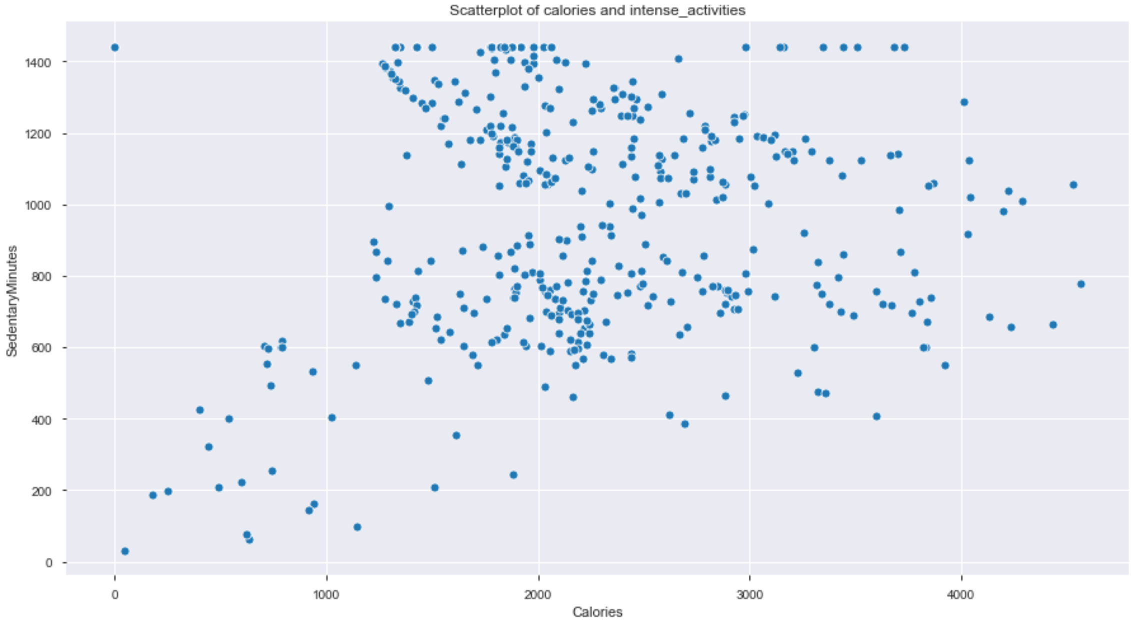


In [30]:

```
# figure size
plt.figure(figsize=(15,8))

# Simple scatterplot
ax = sns.scatterplot(x='Calories', y='SedentaryMinutes', data=activity1)

ax.set_title('Scatterplot of calories and intense_activities')
```

Out[30]:

```
Text(0.5, 1.0, 'Scatterplot of calories and intense_activities')
```

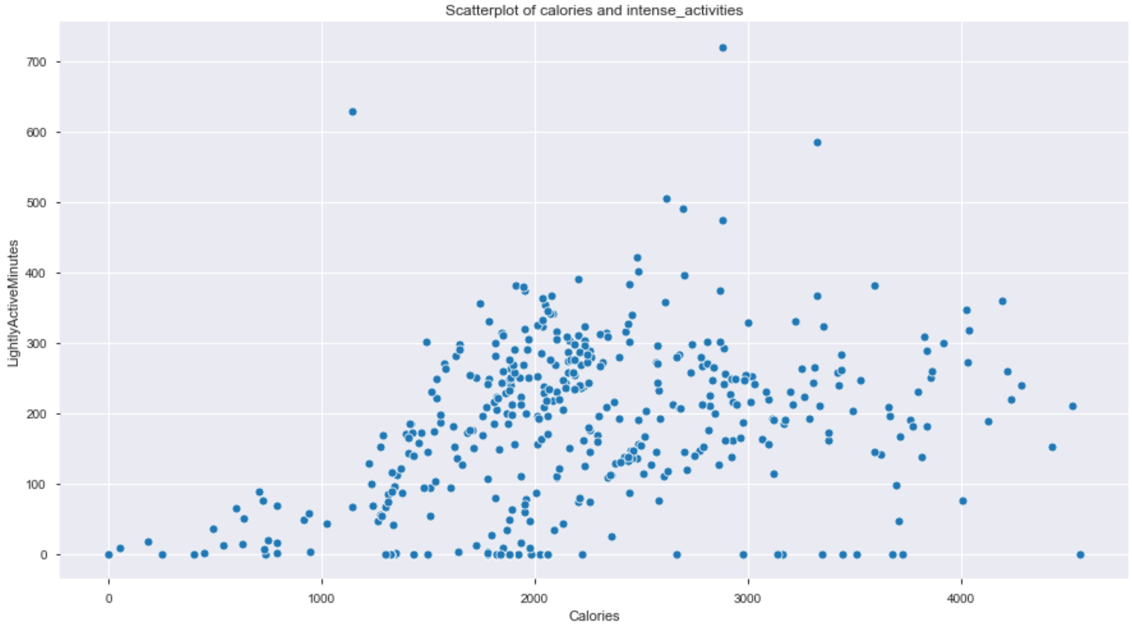


In [32]:

```
# figure size
plt.figure(figsize=(15,8))

# Simple scatterplot
ax = sns.scatterplot(x='Calories', y='LightlyActiveMinutes', data=activity1)

ax.set_title('Scatterplot of calories and intense_activities')
```

Out[32]:

```
Text(0.5, 1.0, 'Scatterplot of calories and intense_activities')
```

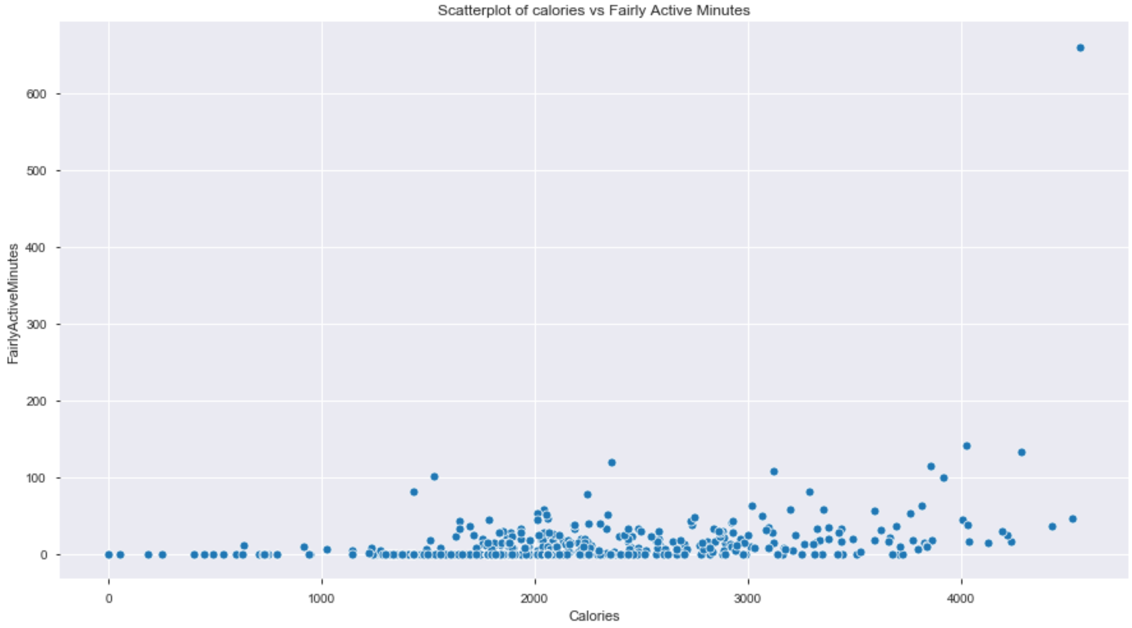


In [34]:

```
# figure size
plt.figure(figsize=(15,8))

# Simple scatterplot between calories burnt in the moderately active minutes
ax = sns.scatterplot(x='Calories', y='FairlyActiveMinutes', data=activity1)

ax.set_title('Scatterplot of calories vs Fairly Active Minutes')
```

Out[34]:

```
Text(0.5, 1.0, 'Scatterplot of calories vs Fairly Active Minutes')
```

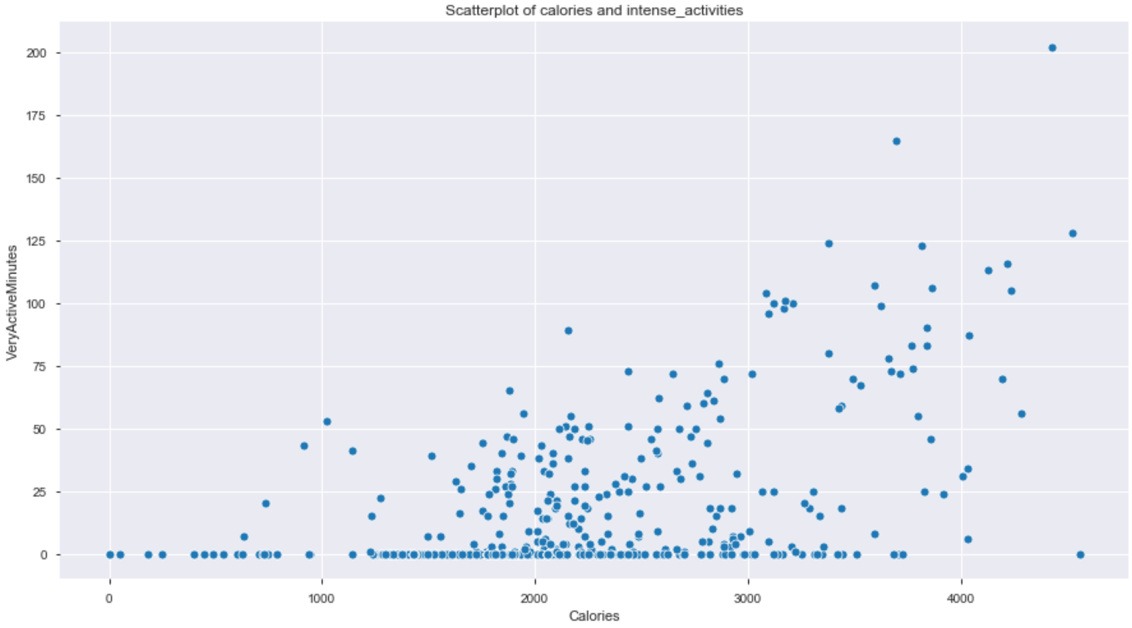


In [35]:

```
# figure size
plt.figure(figsize=(15,8))

# Simple scatterplot between calories burnt in the intensely active minutes
ax = sns.scatterplot(x='Calories', y='VeryActiveMinutes', data=activity1)

ax.set_title('Scatterplot of calories and intense_activities')
```

Out[35]:

```
Text(0.5, 1.0, 'Scatterplot of calories and intense_activities')
```



In [36]:

```
activity1.head(10) # cheking out the datset before transformation
```

Out[36]:

	Id	ActivityDate	TotalSteps	TotalDistance	LoggedActivitiesDistance	VeryActiveD
0	1503960366	2016-03-25	11004	7.11	0.0	
1	1503960366	2016-03-26	17609	11.55	0.0	
2	1503960366	2016-03-27	12736	8.53	0.0	
3	1503960366	2016-03-28	13231	8.93	0.0	
4	1503960366	2016-03-29	12041	7.85	0.0	
5	1503960366	2016-03-30	10970	7.16	0.0	
6	1503960366	2016-03-31	12256	7.86	0.0	
7	1503960366	2016-04-01	12262	7.87	0.0	
8	1503960366	2016-04-02	11248	7.25	0.0	
9	1503960366	2016-04-03	10016	6.37	0.0	

In [37]:

```
activity1=activity1.drop(['Week','Year'],axis=1) # dropping the columns week and year
```

In [38]:

```
activity1.head(10) # cheking out the datset after transformation
```

Out[38]:

	Id	ActivityDate	TotalSteps	TotalDistance	LoggedActivitiesDistance	VeryActiveD
0	1503960366	2016-03-25	11004	7.11	0.0	
1	1503960366	2016-03-26	17609	11.55	0.0	
2	1503960366	2016-03-27	12736	8.53	0.0	
3	1503960366	2016-03-28	13231	8.93	0.0	
4	1503960366	2016-03-29	12041	7.85	0.0	
5	1503960366	2016-03-30	10970	7.16	0.0	
6	1503960366	2016-03-31	12256	7.86	0.0	
7	1503960366	2016-04-01	12262	7.87	0.0	
8	1503960366	2016-04-02	11248	7.25	0.0	
9	1503960366	2016-04-03	10016	6.37	0.0	

In [39]:

```
activity1.shape # cheking the number of rows and columns in the transformed  dataset
```

Out[39]:

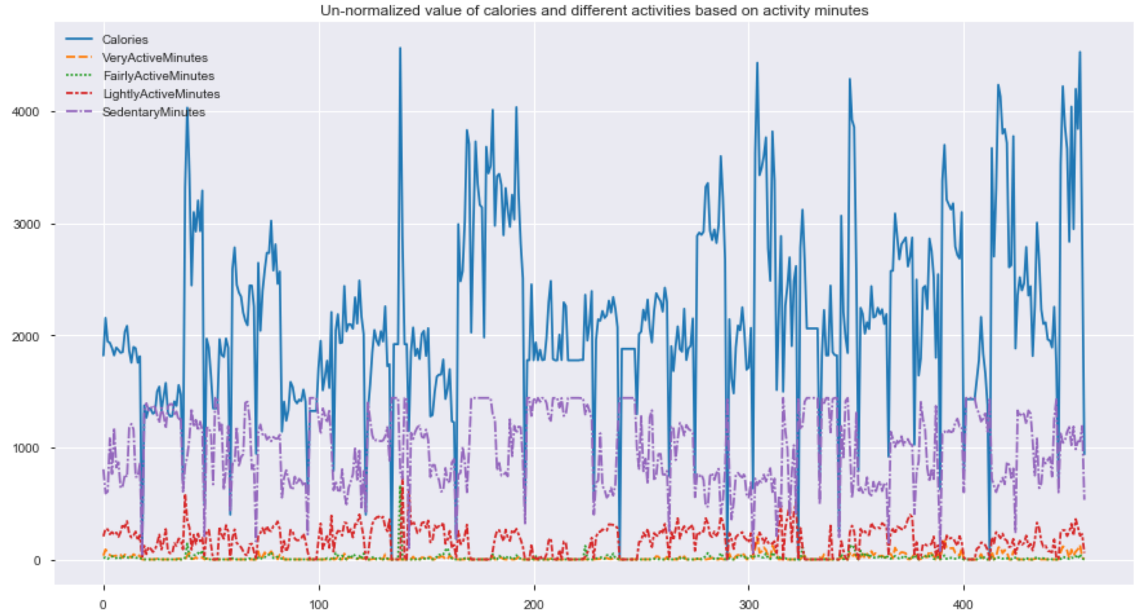
```
(457, 18)
```

In [45]:

```
## plot the raw values

col_select = ['Calories','VeryActiveMinutes','FairlyActiveMinutes','LightlyActiveMinutes
wide_df = activity1[col_select]

# figure size
plt.figure(figsize=(15,8))

# timeseries plot using lineplot
ax = sns.lineplot(data=wide_df)

ax.set_title('Un-normalized value of calories and different activities based on activity
```

Out[45]:

```
Text(0.5, 1.0, 'Un-normalized value of calories and different activities b
ased on activity minutes')
```

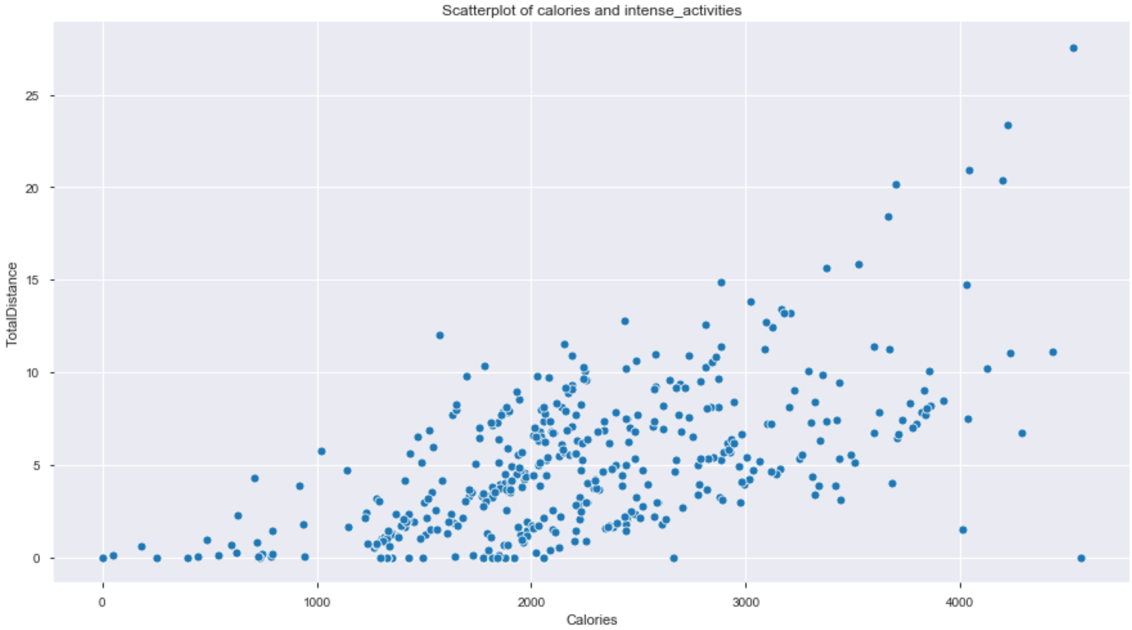


In [41]:

```
# figure size
plt.figure(figsize=(15,8))

# Simple scatterplot between  calories burnt and total distance covered
ax = sns.scatterplot(x='Calories', y='TotalDistance', data=activity1)

ax.set_title('Scatterplot of calories and intense_activities')
```

Out[41]:

```
Text(0.5, 1.0, 'Scatterplot of calories and intense_activities')
```

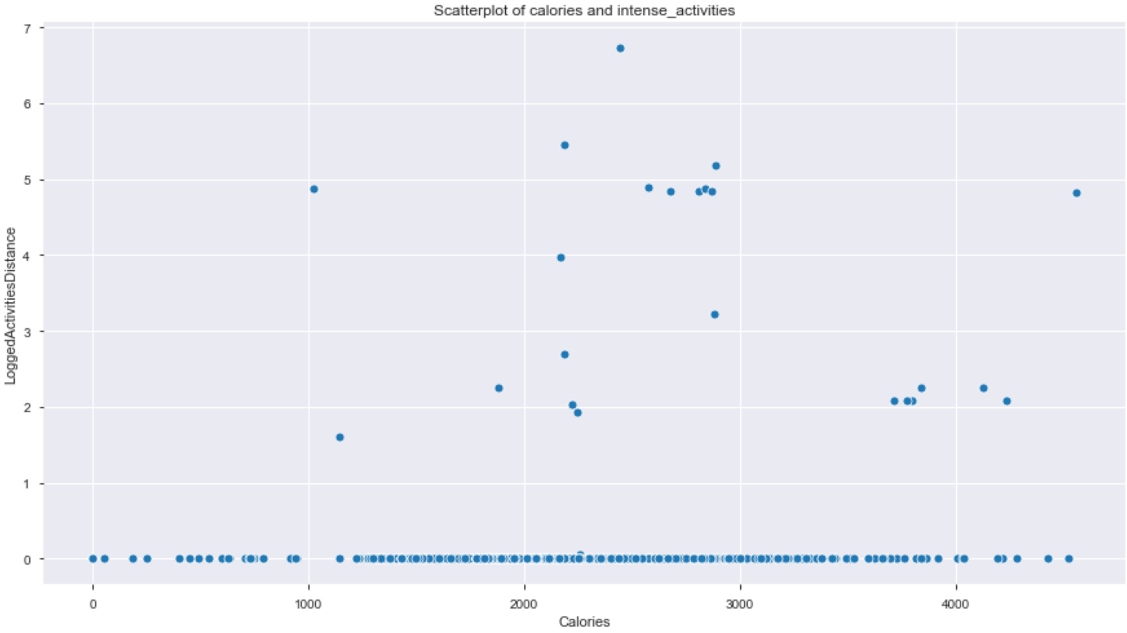


In [42]:

```
# figure size
plt.figure(figsize=(15,8))

# Simple scatterplot between calories burnt and the loggged activities distance
ax = sns.scatterplot(x='Calories', y='LoggedActivitiesDistance', data=activity1)

ax.set_title('Scatterplot of calories and intense_activities')
```

Out[42]:

```
Text(0.5, 1.0, 'Scatterplot of calories and intense_activities')
```

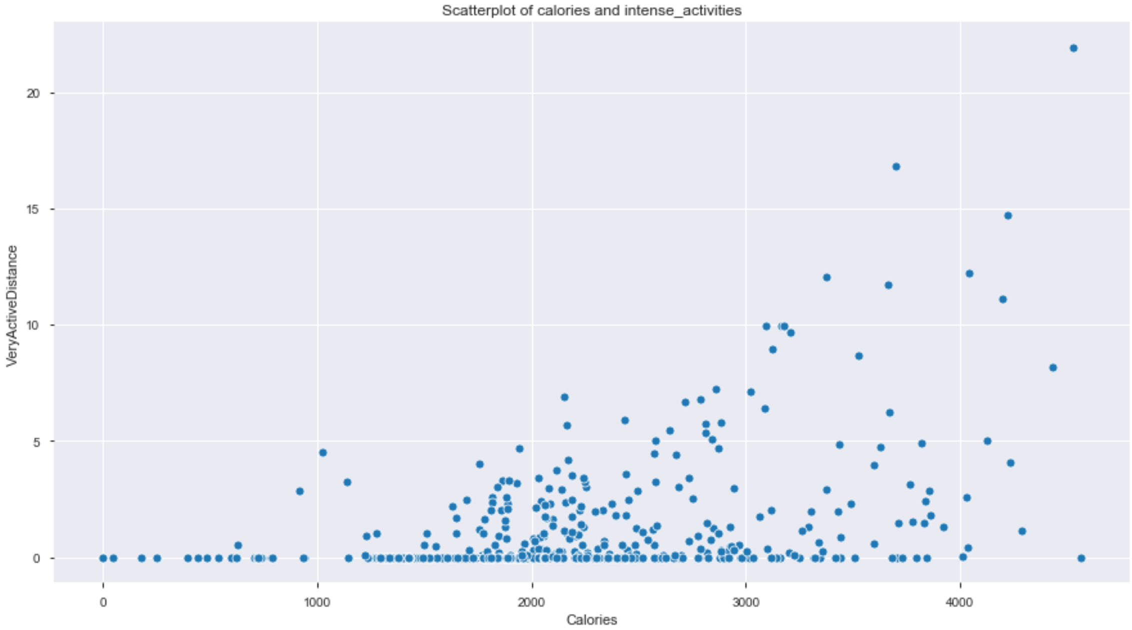


In [43]:

```
# figure size
plt.figure(figsize=(15,8))

# Simple scatterplot between calories burnt and the distance of intense activies
ax = sns.scatterplot(x='Calories', y='VeryActiveDistance', data=activity1)

ax.set_title('Scatterplot of calories and intense_activities')
```

Out[43]:

```
Text(0.5, 1.0, 'Scatterplot of calories and intense_activities')
```

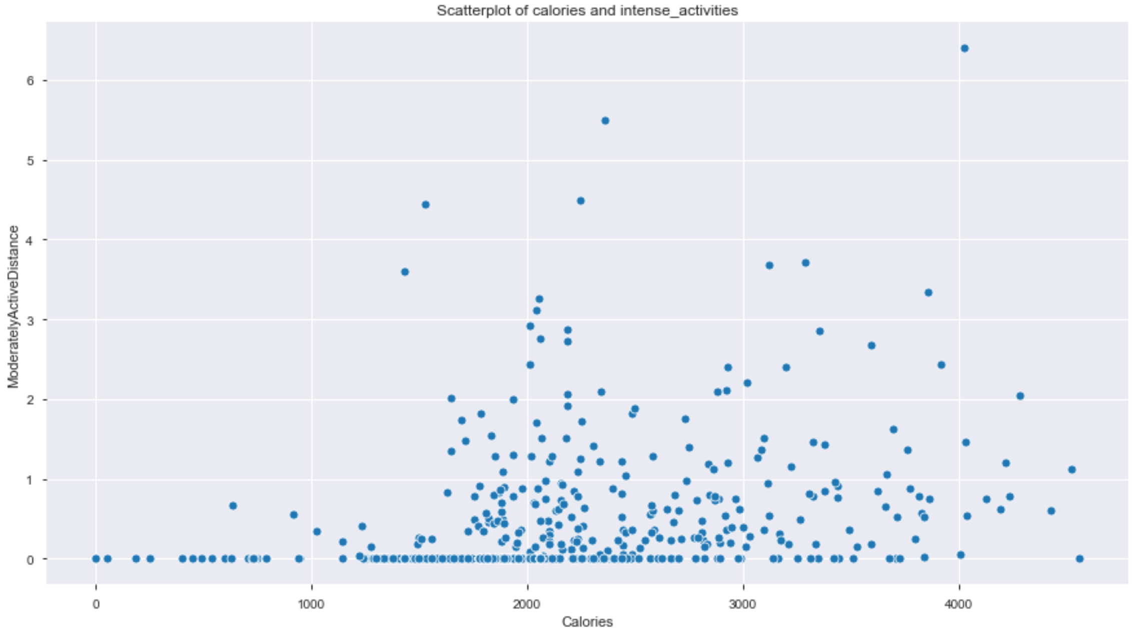


In [44]:

```
# figure size
plt.figure(figsize=(15,8))

# Simple scatterplot between calories burnt and the distance of moderate activies
ax = sns.scatterplot(x='Calories', y='ModeratelyActiveDistance', data=activity1)

ax.set_title('Scatterplot of calories and intense_activities')
```

Out[44]:

```
Text(0.5, 1.0, 'Scatterplot of calories and intense_activities')
```

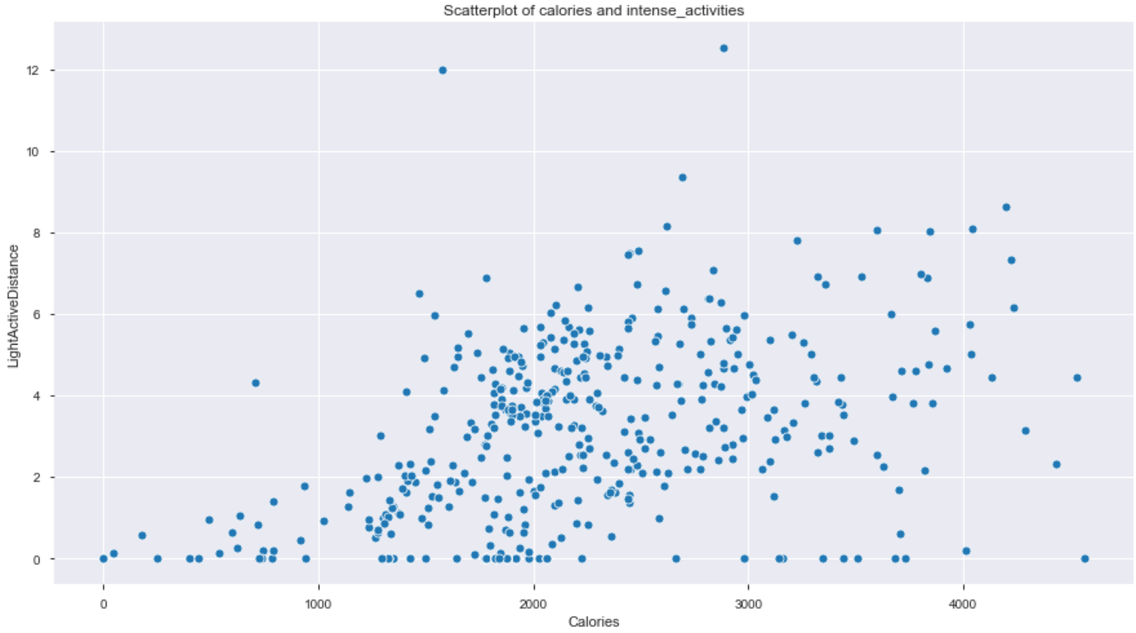


In [40]:

```
# figure size
plt.figure(figsize=(15,8))

# Simple scatterplot
ax = sns.scatterplot(x='Calories', y='LightActiveDistance', data=activity1)

ax.set_title('Scatterplot of calories and intense_activities')
```

Out[40]:

```
Text(0.5, 1.0, 'Scatterplot of calories and intense_activities')
```

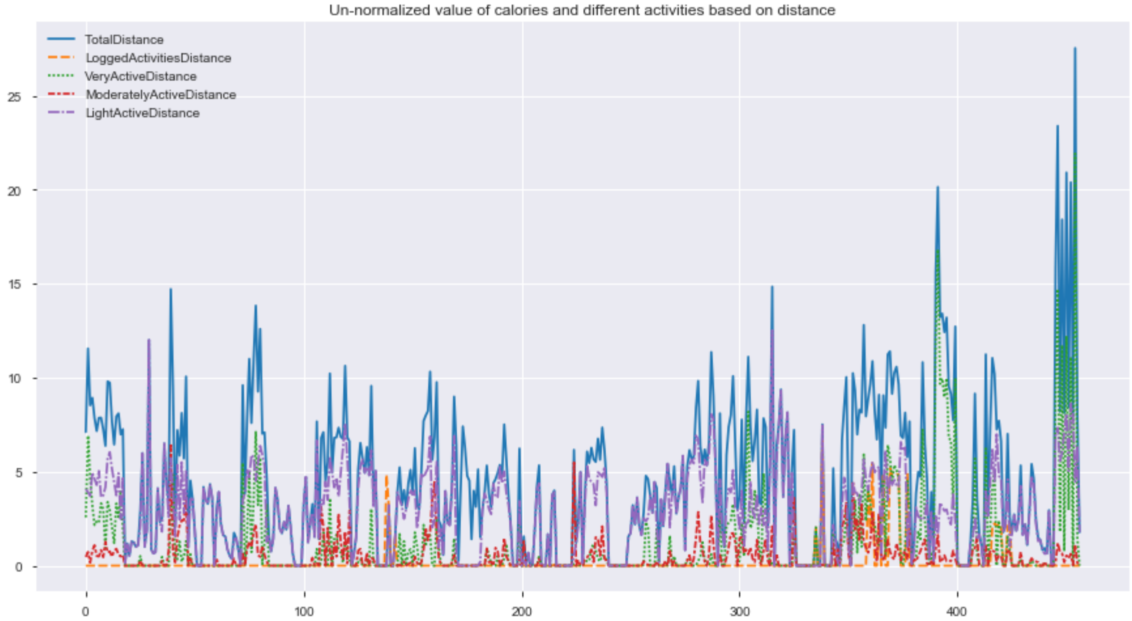


In [46]:

```
## plot the raw values

rol_select = ['TotalDistance','LoggedActivitiesDistance','VeryActiveDistance','Moderatel
wide_df1 = activity1[rol_select]

# figure size
plt.figure(figsize=(15,8))

# timeseries plot using lineplot
ax = sns.lineplot(data=wide_df1)

ax.set_title('Un-normalized value of calories and different activities based on distance
```

Out[46]:

```
Text(0.5, 1.0, 'Un-normalized value of calories and different activities b
ased on distance')
```



The EDA here gives us the insight about the relation between the active hours, the distance for which the user has moderate and intense activity and the calories burnt during that period.