In [1]:
```
#importing data
import pandas as pd
import numpy as np
from datetime import datetime, timedelta
import matplotlib.pyplot as plt
```

In [2]:
```
pwd
```

Out[2]:
```
'C:\\Users\\Deepanshu Dutta\\datascience\\work\\data'
```

In [3]:
```
#reading data
df=pd.read_csv("C:\\Users\\Deepanshu Dutta\\datascience\\work\\data\\order_detail
```

# basic EDA

In [4]:
```
#shape of the data
df.shape
```

Out[4]:
```
(1457, 29)
```

In [5]:
```
df.columns
```

Out[5]:
```
Index(['_id__$oid', 'last_modified__$date', 'delivery_lat',
       'assigned_delivery_boy_id', 'trash', 'created_at__$date',
       'delivery_lon', 'outlet_id', 'order_status', 'status', 'order_id',
       'order_status_id', 'preparation_time', 'committed_time', 'eta',
       'time_of_delivery', 'arrival_time_duration',
       'delivery_boy_assigning_time__$date', 'actual_preparaed_time',
       'pickup_time', 'approve_time', 'reaching_time__$date',
       'arrival_time__$date', 'outlet_distance_from_delivery_person',
       'assigning_lat_lon__001', 'assigning_lat_lon__002',
       'destination_km_distance', 'destination_journey_duration',
       'total_distance_travel_for_delivery'],
      dtype='object')
```

In [6]:
```
df.dtypes
```

Out[6]:
```
_id__$oid                                 object
last_modified__$date                      object
delivery_lat                             float64
assigned_delivery_boy_id                  object
trash                                      int64
created_at__$date                         object
delivery_lon                             float64
outlet_id                                  int64
order_status                              object
status                                    object
order_id                                   int64
order_status_id                          float64
preparation_time                          object
committed_time                            object
eta                                       object
time_of_delivery                          object
arrival_time_duration                     object
delivery_boy_assigning_time__$date        object
actual_preparaed_time                     object
pickup_time                               object
approve_time                              object
reaching_time__$date                      object
arrival_time__$date                       object
outlet_distance_from_delivery_person      object
assigning_lat_lon__001                   float64
assigning_lat_lon__002                   float64
destination_km_distance                   object
destination_journey_duration              object
total_distance_travel_for_delivery        object
dtype: object
```

In [7]:
```
#Checking the presence of missing values
val = df.isnull().values.any()
if val == True :
    print("Missing values present : ", df.isnull().values.sum() )
else:
    print("No missing values Present")
```

```
Missing values present :  16698
```

In [8]:
```
#missing values by columns
df.isnull().sum()
```

Out[8]:
```
_id__$oid                                  0
last_modified__$date                       0
delivery_lat                               0
assigned_delivery_boy_id                   4
trash                                      0
created_at__$date                          0
delivery_lon                               0
outlet_id                                  0
order_status                               5
status                                     0
order_id                                   0
order_status_id                            5
preparation_time                           7
committed_time                           899
eta                                      899
time_of_delivery                        1043
arrival_time_duration                    901
delivery_boy_assigning_time__$date       917
actual_preparaed_time                    951
pickup_time                             1036
approve_time                             948
reaching_time__$date                    1132
arrival_time__$date                     1151
outlet_distance_from_delivery_person    1129
assigning_lat_lon__001                  1130
assigning_lat_lon__002                  1130
destination_km_distance                 1137
destination_journey_duration            1137
total_distance_travel_for_delivery      1137
dtype: int64
```

In [9]:
```
#sample data
#df.head(10)
df.status.value_counts()
```

Out[9]:
```
New                     892
delivery boy assigned   565
Name: status, dtype: int64
```

In [10]:
```
#converting date to datetime object for easy analysis
df['last_modified__$date']=pd.to_datetime(df['last_modified__$date'])
df.head()
```

Out[10]:

| | _id__$oid | last_modified__$date | delivery_lat | assigned_delivery_boy_id | trash | c |
|---|---|---|---|---|---|---|
| 0 | 5f2549e3d1dfb3348735a34d | 2020-08-01 11:49:07.775000+00:00 | 22.485330 | 21 | 0 | |
| 1 | 5f24f749e63cb52d5f17df46 | 2020-08-01 05:07:38.903000+00:00 | 22.585257 | 1 | 0 | |
| 2 | 5f25372956c63423201653f7 | 2020-08-01 10:40:21.208000+00:00 | 22.492368 | 21 | 1 | |
| 3 | 5f257163d97253b6141c6471 | 2020-08-01 15:28:58.058000+00:00 | 22.509654 | 24 | 0 | |
| 4 | 5f25dc1f5ed457c026f875d9 | 2020-08-01 21:21:30.505000+00:00 | 22.585755 | 10 | 0 | |

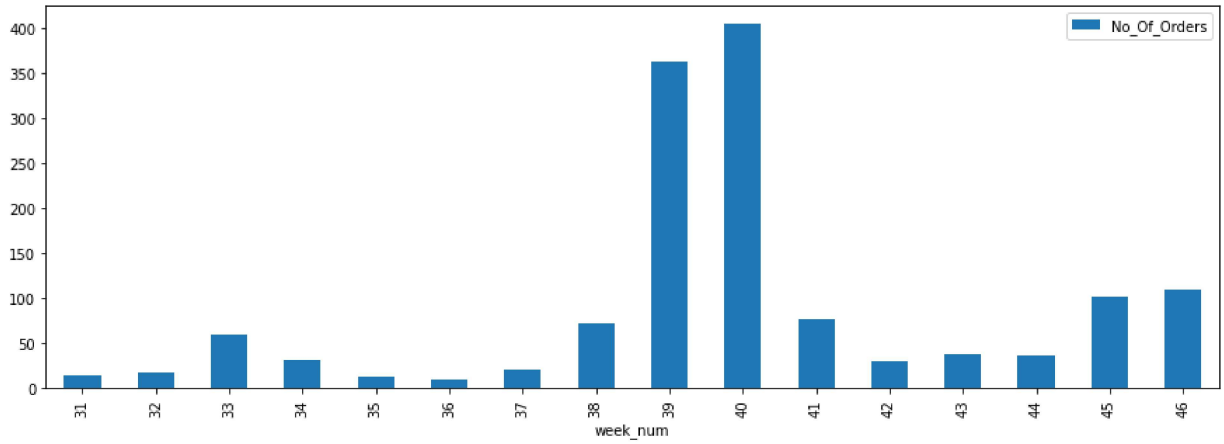
5 rows × 29 columns

In [11]:
```
#extracting the week of the data
weeks=[]
for i in range(0,len(df['last_modified__$date'])):
    dt=df['last_modified__$date'][i]
    weeks.append(dt.isocalendar()[1])
df['week_number']=weeks
```

In [12]:
```
#removing the null values
df['order_status_id'].value_counts()
#we see that the maximum rows of order status id has the value 1 so we replace th
df['order_status_id']=df['order_status_id'].fillna(1)
#df.isnull().sum()
```

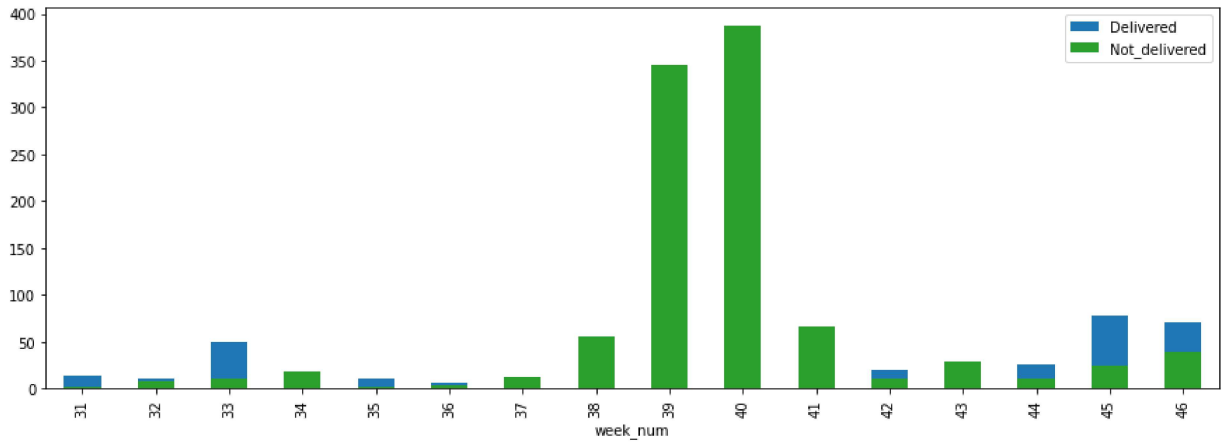
# No Of Orders Per Week

In [13]:
```
data=[df['week_number'],df['order_id']]
Q1 = pd.DataFrame(data=data)
Q1=Q1.T
weeknum=[]
no_of_orders=[]
for i in range(int(Q1['week_number'].iloc[0]),int(Q1['week_number'].iloc[-1])):
    weeknum.append((i))
    no_of_orders.append(int(Q1['week_number'][Q1['week_number']==i].value_counts(
data=[weeknum,no_of_orders]
Q1 = pd.DataFrame(data=data)
Q1=Q1.T
headers = ["week_num","No_Of_Orders"]
Q1.columns = headers
#Q1.head()
ax = Q1.plot(x="week_num", y="No_Of_Orders", kind="bar",figsize=[15,5])
plt.show()
```




# Delivered vs Not Delivered Per Week

In [14]:
```
#saving the week number and order status in a different data frame
data=[df['week_number'],df['order_status_id']]
Q2 = pd.DataFrame(data=data)
Q2=Q2.T
Q2['is-Delivered']=Q2['order_status_id'].apply(lambda x:'Delivered' if x==6 else
#Q2['is-Delivered'].value_counts()
#counting the number of items delivered vs not delivered per week
delivered=[]
weeknum=[]
notdelivered=[]
for i in range(int(Q2['week_number'].iloc[0]),int(Q2['week_number'].iloc[-1])):
    weeknum.append((i))
    delivered.append(Q2['is-Delivered'][Q2['week_number']==i].value_counts()['Del
    notdelivered.append(Q2['is-Delivered'][Q2['week_number']==i].value_counts()['
data=[weeknum,delivered,notdelivered]
Q2 = pd.DataFrame(data=data)
Q2=Q2.T
headers = ["week_num","Delivered","Not_delivered"]
Q2.columns = headers
#Q2.head()
ax = Q2.plot(x="week_num", y="Delivered", kind="bar",figsize=[15,5])
Q2.plot(x="week_num", y="Not_delivered", kind="bar", ax=ax, color="C2")
plt.show()
```




In [15]:
```
Q1.head()
```

Out[15]:

| | week_num | No_Of_Orders |
|---|---|---|
| **0** | 31 | 15 |
| **1** | 32 | 17 |
| **2** | 33 | 60 |
| **3** | 34 | 32 |
| **4** | 35 | 13 |

In [16]:
```
Q2.head()
```

Out[16]:

| | week_num | Delivered | Not_delivered |
|---|---|---|---|
| **0** | 31 | 14 | 1 |
| **1** | 32 | 10 | 7 |
| **2** | 33 | 49 | 11 |
| **3** | 34 | 14 | 18 |
| **4** | 35 | 11 | 2 |

In [17]:
```
Q2['No_Of_Orders']=Q1['No_Of_Orders']
```

In [18]:
```
Q2[Q2['week_num']==43]
```

Out[18]:

| | week_num | Delivered | Not_delivered | No_Of_Orders |
|---|---|---|---|---|
| **12** | 43 | 8 | 29 | 37 |

In [19]:
```
Q2.head()
```

Out[19]:

| | week_num | Delivered | Not_delivered | No_Of_Orders |
|---|---|---|---|---|
| **0** | 31 | 14 | 1 | 15 |
| **1** | 32 | 10 | 7 | 17 |
| **2** | 33 | 49 | 11 | 60 |
| **3** | 34 | 14 | 18 | 32 |
| **4** | 35 | 11 | 2 | 13 |

### Convert DataFrame to Csv file

In [20]:
```
Q2.to_csv(r"C:\\Users\\Deepanshu Dutta\\datascience\\work\\data\\Delivered_vs_not
```

# Find Avg Delivery Distance And Total_km_Travelled per Week

In [21]:
```
df1=df.copy()
```

In [22]:
```
df1.columns
```

Out[22]:
```
Index(['_id__$oid', 'last_modified__$date', 'delivery_lat',
       'assigned_delivery_boy_id', 'trash', 'created_at__$date',
       'delivery_lon', 'outlet_id', 'order_status', 'status', 'order_id',
       'order_status_id', 'preparation_time', 'committed_time', 'eta',
       'time_of_delivery', 'arrival_time_duration',
       'delivery_boy_assigning_time__$date', 'actual_preparaed_time',
       'pickup_time', 'approve_time', 'reaching_time__$date',
       'arrival_time__$date', 'outlet_distance_from_delivery_person',
       'assigning_lat_lon__001', 'assigning_lat_lon__002',
       'destination_km_distance', 'destination_journey_duration',
       'total_distance_travel_for_delivery', 'week_number'],
      dtype='object')
```

In [23]:
```
df1['destination_km_distance'].isnull().sum()
```

Out[23]:
```
1137
```

In [24]:
```
## basic data clean using "trash" column
## we need to take only 0 trash value because 0 means not canceled order and 1 me
##df1_clean=df1.loc[df['trash']==0]
##df1_clean.shape
```

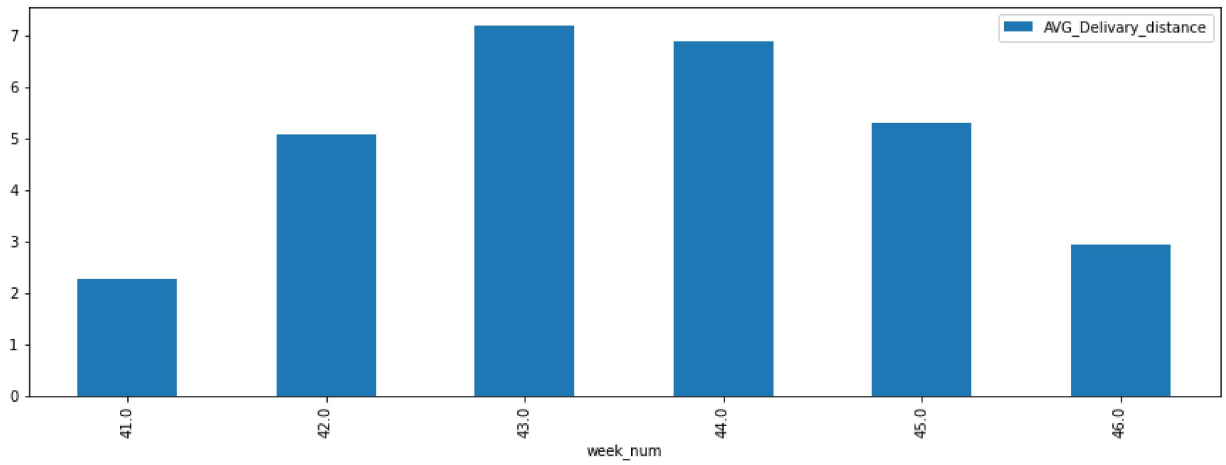
In [25]:
```
#converting date to datetime object for easy analysis
df['last_modified__$date']=pd.to_datetime(df['last_modified__$date'])
df.head()
```

Out[25]:

| | _id__$oid | last_modified__$date | delivery_lat | assigned_delivery_boy_id | trash | c |
|---|---|---|---|---|---|---|
| **0** | 5f2549e3d1dfb3348735a34d | 2020-08-01 11:49:07.775000+00:00 | 22.485330 | 21 | 0 | |
| **1** | 5f24f749e63cb52d5f17df46 | 2020-08-01 05:07:38.903000+00:00 | 22.585257 | 1 | 0 | |
| **2** | 5f25372956c63423201653f7 | 2020-08-01 10:40:21.208000+00:00 | 22.492368 | 21 | 1 | |
| **3** | 5f257163d97253b6141c6471 | 2020-08-01 15:28:58.058000+00:00 | 22.509654 | 24 | 0 | |
| **4** | 5f25dc1f5ed457c026f875d9 | 2020-08-01 21:21:30.505000+00:00 | 22.585755 | 10 | 0 | |

5 rows × 30 columns

In [26]:
```
#saving the week number and destination_km_distance in a different data frame
df2=df.copy()
df2=df2[df2['trash']!=1]
data=[df2['week_number'],df2['destination_km_distance']]
Q3 = pd.DataFrame(data=data)
Q3=Q3.T
#change object to float
Q3["destination_km_distance"] = Q3["destination_km_distance"].str.split(" ", n =
Q3['destination_km_distance']=Q3['destination_km_distance'].astype('float')
#second basic data cleanning
Q3=Q3.dropna()
#Q3.head()
week_num=[]
AVG_Delivary_distance=[]
Total_km_travelled=[]
for i in range(int(Q3['week_number'].iloc[0]),int(Q3['week_number'].iloc[-1])):
    week_num.append(i)
    AVG_Delivary_distance.append((Q3['destination_km_distance'][Q3['week_number']
    Total_km_travelled.append(Q3['destination_km_distance'][Q3['week_number']==i]
data=[week_num,AVG_Delivary_distance,Total_km_travelled]
Q3 = pd.DataFrame(data=data)
Q3=Q3.T
headers = ["week_num","AVG_Delivary_distance",'Total_km_travelled']
Q3.columns = headers
##Q3.head()
##plotting the graph of week number vs
ax = Q3.plot(x="week_num", y="AVG_Delivary_distance", kind="bar",figsize=[15,5])
```



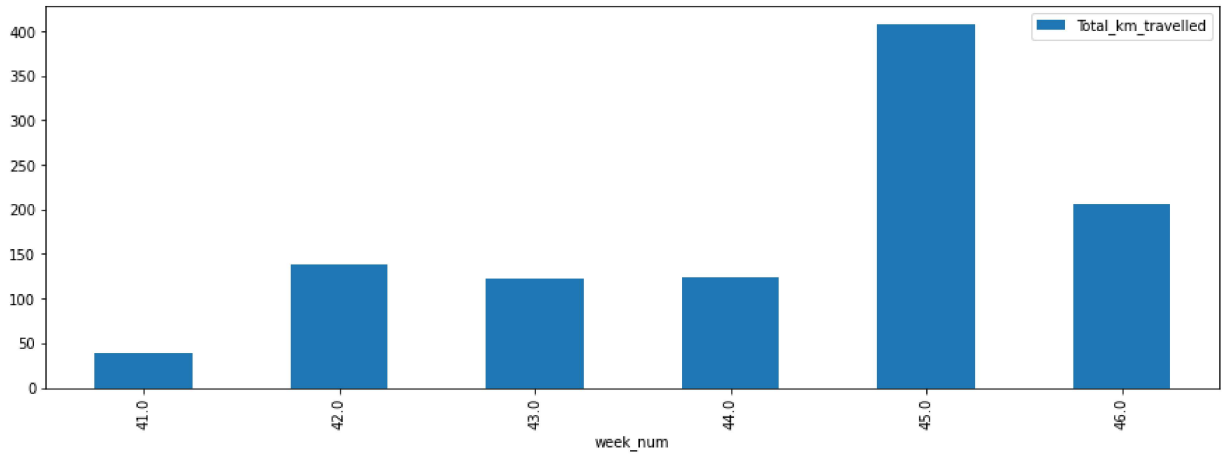
In [27]:
```
##basic EDA of week_num vs AVG_Delivary_distance
Q3.head(20)
```

Out[27]:

| | week_num | AVG_Delivary_distance | Total_km_travelled |
|---|---|---|---|
| **0** | 41.0 | 2.276471 | 38.70 |
| **1** | 42.0 | 5.088889 | 137.40 |
| **2** | 43.0 | 7.200000 | 122.40 |
| **3** | 44.0 | 6.894444 | 124.10 |
| **4** | 45.0 | 5.306494 | 408.60 |
| **5** | 46.0 | 2.937429 | 205.62 |

**plot for Week_num vs Total_km_travelled per week**

In [28]:
```
ax = Q3.plot(x="week_num", y="Total_km_travelled", kind="bar",figsize=[15,5])
```




# Avg Pickup Distance Per Week

In [29]:
```
df['outlet_distance_from_delivery_person']
```

Out[29]:
```
0          NaN
1          NaN
2          NaN
3          NaN
4          NaN
          ...
1452    4.4 km
1453    1.2 km
1454    2.8 km
1455    1.1 km
1456    1.7 km
Name: outlet_distance_from_delivery_person, Length: 1457, dtype: object
```

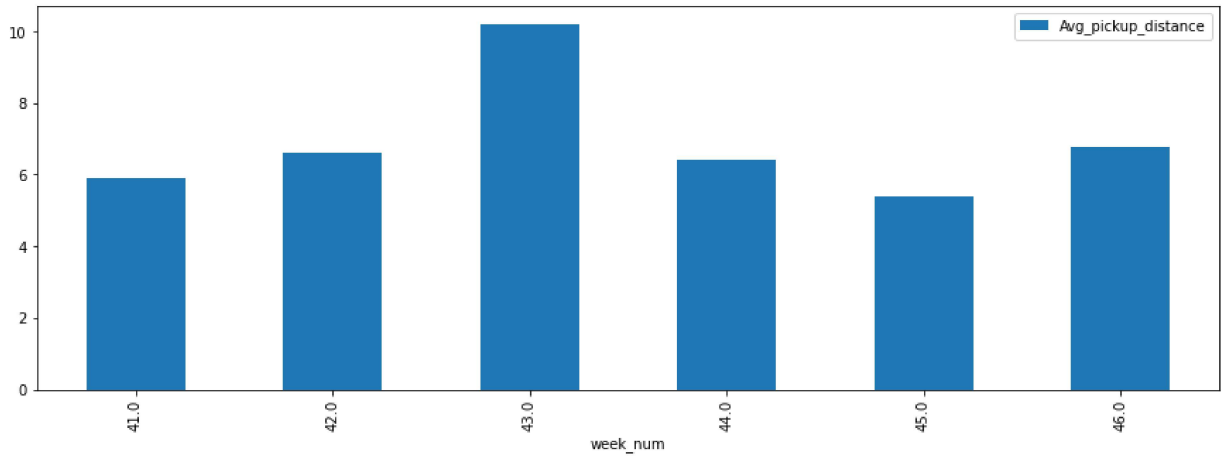
In [30]:
```
#saving the week number and destination_km_distance in a different data frame
df1=df.copy()

df1=df1[df1['outlet_id']!=3]## we dont want to take rows of outlet_id=3
data=[df1['week_number'],df1['outlet_distance_from_delivery_person']]
Q4 = pd.DataFrame(data=data)
Q4=Q4.T
#change object to float
Q4["outlet_distance_from_delivery_person"] = Q4["outlet_distance_from_delivery_pe
Q4['outlet_distance_from_delivery_person']=Q4['outlet_distance_from_delivery_pers
#second basic data cleanning
Q4=Q4.dropna()

week_num=[]
Avg_pickup_distance=[]
for i in range(int(Q4['week_number'].iloc[0]),int(Q4['week_number'].iloc[-1])):
        week_num.append(i)
        Avg_pickup_distance.append((Q4['outlet_distance_from_delivery_person'][Q4

data=[week_num,Avg_pickup_distance]
Q4 = pd.DataFrame(data=data)
Q4=Q4.T
headers=['week_num','Avg_pickup_distance']
Q4.columns=headers
Q4.head()
```

Out[30]:

| | week_num | Avg_pickup_distance |
|---|---|---|
| **0** | 41.0 | 5.918750 |
| **1** | 42.0 | 6.600000 |
| **2** | 43.0 | 10.216667 |
| **3** | 44.0 | 6.428000 |
| **4** | 45.0 | 5.391463 |

**Plotting Graph Of week_num Vs Avg_pickup_distance**

In [31]:
```
ax =Q4.plot(x="week_num", y="Avg_pickup_distance", kind="bar",figsize=[15,5])
```




# Delta Of ETD And ATD

In [32]:
```
data=[df['eta'],df['time_of_delivery']]
Q5 = pd.DataFrame(data=data)
Q5=Q5.T
Q5['eta']=Q5['eta'].fillna("null")
Q5['time_of_delivery']=pd.to_datetime(Q5['time_of_delivery'])
Q5['time_of_delivery']=Q5['time_of_delivery'].fillna("null")
Ehr=0
ETD=[]
timediff=[]

length=0
for i in range(0,len(df['time_of_delivery'])):
    if(Q5['eta'][i] != "null" and Q5['time_of_delivery'][i] != "null"):
        Arr_T=Q5['time_of_delivery'][i].time()
        ETD=Q5['eta'][i].split(':')
        if(ETD[1].split()[1]=='PM'):
            Ehr=int(ETD[0])+12
        diff = -((int(Arr_T.hour)*60+int(Arr_T.minute))-((Ehr*60)+int(ETD[1].spli
        timediff.append(diff)
        length+=1
#print(diff/length)
##int(len(timediff))
##int(sum(timediff)/length)
```

In [33]:
```
data=[df['eta'],df['time_of_delivery']]
Q5 = pd.DataFrame(data=data)
Q5=Q5.T
```

In [34]:
```
Q5.head()
```

Out[34]:

| | eta | time_of_delivery |
|---|---|---|
| **0** | 4:57 PM | 11:49 AM |
| **1** | 9:48 AM | 5:07 AM |
| **2** | 1:00 PM | NaN |
| **3** | 7:26 PM | 3:28 PM |
| **4** | 3:11 AM | 9:21 PM |

In [35]:
```
Q5['eta']=Q5['eta'].fillna("null")
Q5['time_of_delivery']=pd.to_datetime(Q5['time_of_delivery'])
```

In [36]:
```
Q5.head()
```

Out[36]:

| | eta | time_of_delivery |
|---|---|---|
| **0** | 4:57 PM | 2020-11-25 11:49:00 |
| **1** | 9:48 AM | 2020-11-25 05:07:00 |
| **2** | 1:00 PM | NaT |
| **3** | 7:26 PM | 2020-11-25 15:28:00 |
| **4** | 3:11 AM | 2020-11-25 21:21:00 |

In [37]:
```
Q5['time_of_delivery']=Q5['time_of_delivery'].fillna("null")
```

In [38]:
```
Q5.head()
```

Out[38]:

| | eta | time_of_delivery |
|---|---|---|
| **0** | 4:57 PM | 2020-11-25 11:49:00 |
| **1** | 9:48 AM | 2020-11-25 05:07:00 |
| **2** | 1:00 PM | null |
| **3** | 7:26 PM | 2020-11-25 15:28:00 |
| **4** | 3:11 AM | 2020-11-25 21:21:00 |

In [46]:
```
### First Part
data=[df['eta'],df['time_of_delivery']]
Q5 = pd.DataFrame(data=data)
Q5=Q5.T
Q5['eta']=Q5['eta'].fillna("null")
Q5['time_of_delivery']=pd.to_datetime(Q5['time_of_delivery'])
Q5['time_of_delivery']=Q5['time_of_delivery'].fillna("null")
Ehr=0
ETD=[]
timediff=[]

length=0
for i in range(0,len(df['time_of_delivery'])):
    if(Q5['eta'][i] != "null" and Q5['time_of_delivery'][i] != "null"):
        Arr_T=Q5['time_of_delivery'][i].time()
        ETD=Q5['eta'][i].split(':')
        if(ETD[1].split()[1]=='PM'):
            Ehr=int(ETD[0])+12
        diff = -((int(Arr_T.hour)*60+int(Arr_T.minute))-((Ehr*60)+int(ETD[1].spli
        timediff.append(diff)
        length+=1

data=[timediff]
Q5=pd.DataFrame(data=data)
Q5=Q5.T
header=["Delta_of_ETD_And_ATD"]
Q5.columns=header
Q5.shape
```

Out[46]: (380, 1)

In [47]:
```
##Second Part
data=[df['eta'],df['time_of_delivery'],df['week_number']]
Q6=pd.DataFrame(data=data)
Q6=Q6.T
Q6['eta']=Q6['eta'].fillna("null")
Q6['time_of_delivery']=pd.to_datetime(Q6['time_of_delivery'])
Q6['time_of_delivery']=Q6['time_of_delivery'].fillna("null")
Q7=Q6[(Q6['eta'] != "null") & (Q6['time_of_delivery'] != "null")]
Q7['Delta_of_ETD_And_ATD']=Q5['Delta_of_ETD_And_ATD']
```

```
<ipython-input-47-ec3d0cbb1113>:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  Q7['Delta_of_ETD_And_ATD']=Q5['Delta_of_ETD_And_ATD']
```

In [48]:
```
Q7.head()
```

Out[48]:

| | eta | time_of_delivery | week_number | Delta_of_ETD_And_ATD |
|---|---|---|---|---|
| **0** | 4:57 PM | 2020-11-25 11:49:00 | 31 | 308.0 |
| **1** | 9:48 AM | 2020-11-25 05:07:00 | 31 | 701.0 |
| **3** | 7:26 PM | 2020-11-25 15:28:00 | 31 | -130.0 |
| **4** | 3:11 AM | 2020-11-25 21:21:00 | 31 | -30.0 |
| **5** | 1:18 AM | 2020-11-25 19:48:00 | 31 | -79.0 |

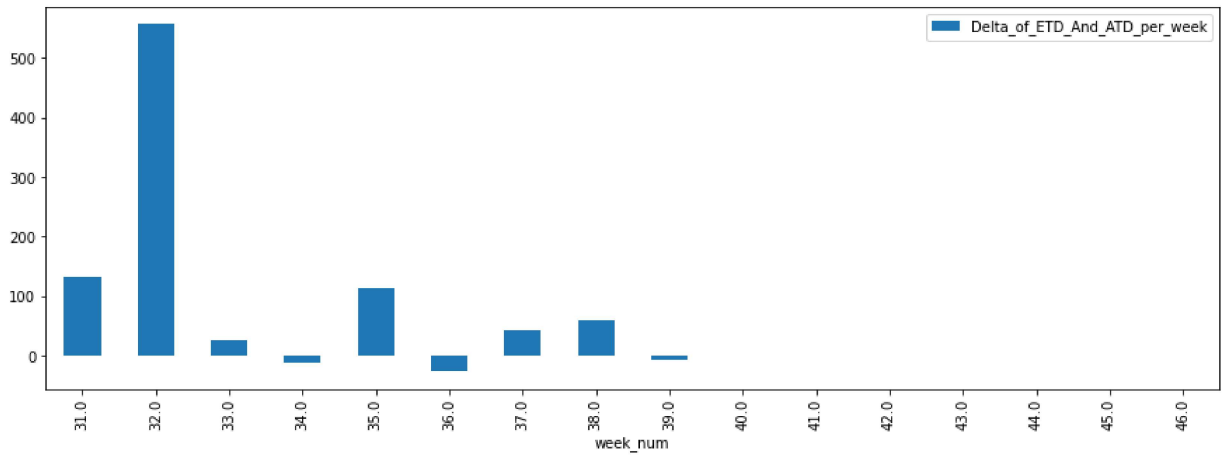
In [49]:
```
## Third part
data=[Q7['week_number'],Q7['Delta_of_ETD_And_ATD']]
Q8 = pd.DataFrame(data=data)
Q8=Q8.T
weeknum=[]
Delta_of_ETD_And_ATD_per=[]
for i in range(int(Q8['week_number'].iloc[0]),int(Q8['week_number'].iloc[-1])):
    weeknum.append(i)
    Delta_of_ETD_And_ATD_per.append((Q8['Delta_of_ETD_And_ATD'][Q8['week_number']
data=[weeknum,Delta_of_ETD_And_ATD_per]
Q9 = pd.DataFrame(data=data)
Q9=Q9.T
headers = ["week_num","Delta_of_ETD_And_ATD_per_week"]
Q9.columns = headers
#Q1.head()
#ax = Q1.plot(x="week_num", y="No_Of_Orders", kind="bar",figsize=[15,5])
#plt.show()
```

In [50]:
```
Q9.head()
```

Out[50]:

| | week_num | Delta_of_ETD_And_ATD_per_week |
|---|---|---|
| **0** | 31.0 | 132.785714 |
| **1** | 32.0 | 557.200000 |
| **2** | 33.0 | 26.191489 |
| **3** | 34.0 | -10.357143 |
| **4** | 35.0 | 114.181818 |

In [51]:
```
ax = Q9.plot(x="week_num", y="Delta_of_ETD_And_ATD_per_week", kind="bar",figsize=
plt.show()
```




In [ ]:

In [ ]:

In [ ]: