# 1, Data Exploration

## 1.1 Overview

In [2]:

```
import pandas as pd

# read data
df = pd.read_csv("IPPS_DRG_FY2017.csv")
```

### (a)

In [3]:

```
def Q1_1a():
    id = df['Provider Id']
    id_unique = pd.unique(id)
    states = pd.unique(df['Provider State'])
    maxNum = 0
    output = ""
    for state in states:
        temp_df = df[df['Provider State'] == state]
        numOfUniqueId = len(pd.unique(temp_df['Provider Id']))
        if numOfUniqueId > maxNum:
            maxNum = numOfUniqueId
            output = state
    return len(id_unique), output

Q1_1a()
```

Out[3]:

```
(3182, 'CA')
```

***3182*** Provider Id in the dataset are different.
***CA*** has the most number of unique providers out all the other states.

### (b)

In [4]:

```
def Q1_1b():
    discharge = df['Total Discharges']
    mean = discharge.mean()
    std = discharge.std()
    median = discharge.median()
    print("mean: " + str(mean))
    print("std: " + str(std))
    print("median: " + str(median))


Q1_1b()
```

```
mean: 37.604421240290336
std: 57.25932516243724
median: 21.0
```

The mean of hospital discharges in FY2017 is **37.604421240290336**.
The median of hospital discharges in FY2017 is **21.0**.
The standard deviation of hospital discharges in FY2017 is **57.25932516243724**.

## (c)

In [5]:

```
def Q1_1c():
    definition = df['DRG Definition']
    definition_unique = pd.unique(definition)
    maxNum = 0
    output = ""
    for d in definition_unique:
        temp_df = df[df['DRG Definition'] == d]
        if len(temp_df) > maxNum:
            output = d
            maxNum = len(temp_df)

    return len(definition_unique), output

Q1_1c()
```

Out[5]:

```
(563, '871 - SEPTICEMIA OR SEVERE SEPSIS W/O MV >96 HOURS W MCC')
```

**563** DRG Definitions in this dataset is unique.
**871 - SEPTICEMIA OR SEVERE SEPSIS W/O MV >96 HOURS W MCC** coded the most in FY 2017.

## (d)

In [7]:

```
def Q1_1d():
    temp_df = df[['DRG Definition','Total Discharges']]
    temp_df = temp_df.groupby('DRG Definition').sum()

    return temp_df['Total Discharges'].idxmin(), temp_df['Total Discharges'].min()

Q1_1d()
```

Out[7]:

```
('058 - MULTIPLE SCLEROSIS & CEREBELLAR ATAXIA W MCC', 11)
```

**058 - MULTIPLE SCLEROSIS & CEREBELLAR ATAXIA W MCC** has the least number of hospital discharges.
Its Total Discharge is **11**.

## 1.2 Distributions and Outliers

### (a)

In [20]:

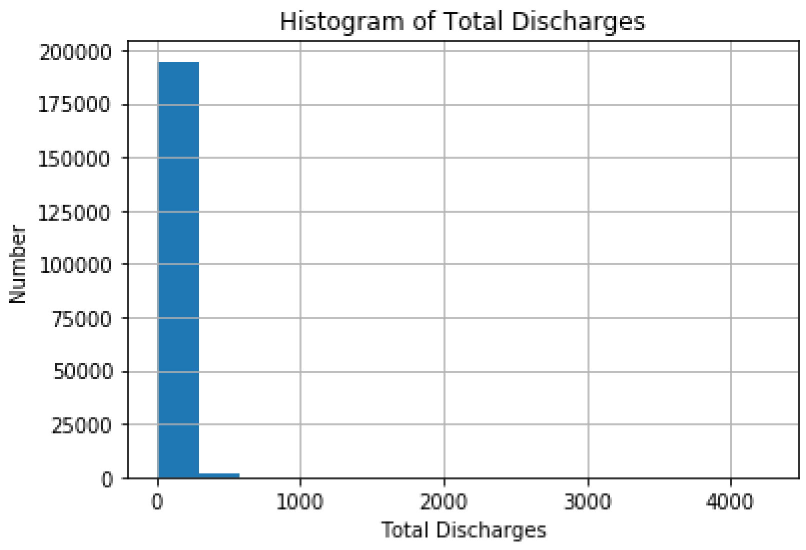
```
import pylab as pl

td = df['Total Discharges'].dropna()

hist = td.hist(bins = 15)
pl.title("Histogram of Total Discharges")
pl.xlabel("Total Discharges")
pl.ylabel("Number")
hist
```

Out[20]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e6a2f88988>
```



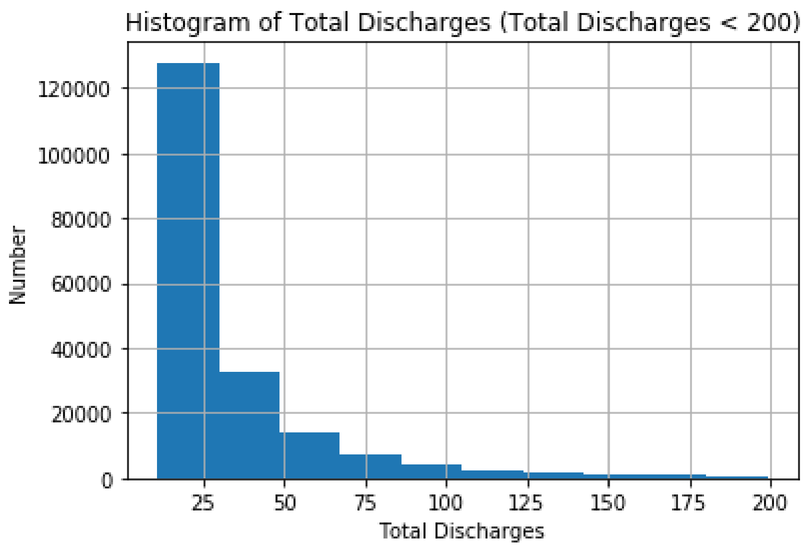
In [24]:

```
td = df[df['Total Discharges'] < 200]
td = td['Total Discharges'].dropna()

hist = td.hist(bins = 10)
pl.title("Histogram of Total Discharges (Total Discharges < 200)")
pl.xlabel("Total Discharges")
pl.ylabel("Number")
hist
```

Out[24]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e6a32b52c8>
```



In [14]:

```
td_outlier = df[df['Total Discharges'] > 3000]
td_outlier
```

Out[14]:

| | DRG Definition | Provider Id | Provider Name | Provider Street Address | Provider City | Provider State | Provider Zip Code | Ho Re R ( Descri |
|---|---|---|---|---|---|---|---|---|
| 127138 | 470 - MAJOR JOINT REPLACEMENT OR REATTACHMENT ... | 330270 | HOSPITAL FOR SPECIAL SURGERY | 535 EAST 70TH STREET | NEW YORK | NY | 10021 | Manh |

As is shown above, the provider with highest total discharges is located in New York Manhattan. It's not surprising that this provider is in the most prosperous area of the world.

## (b)

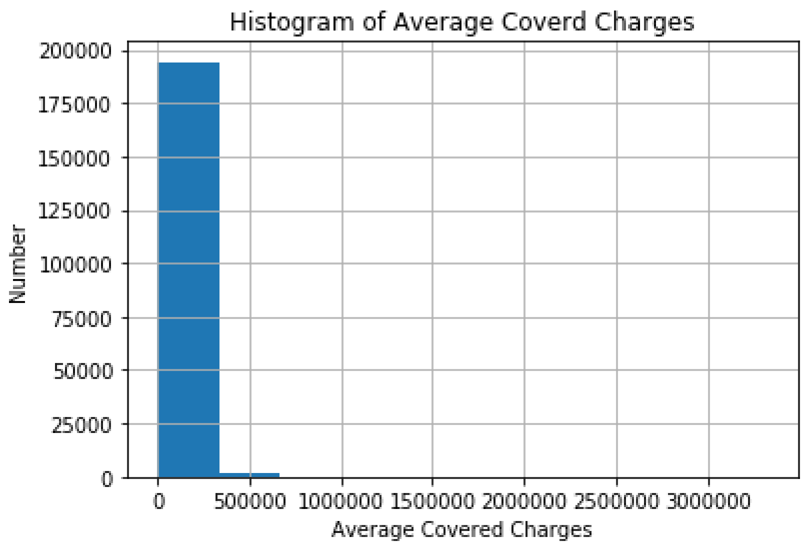
In [29]:

```
acc = df['Average Covered Charges'].dropna()

hist = acc.hist(bins = 10)
pl.title("Histogram of Average Coverd Charges")
pl.xlabel("Average Covered Charges")
pl.ylabel("Number")
hist
```

Out[29]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e6a2e15108>
```



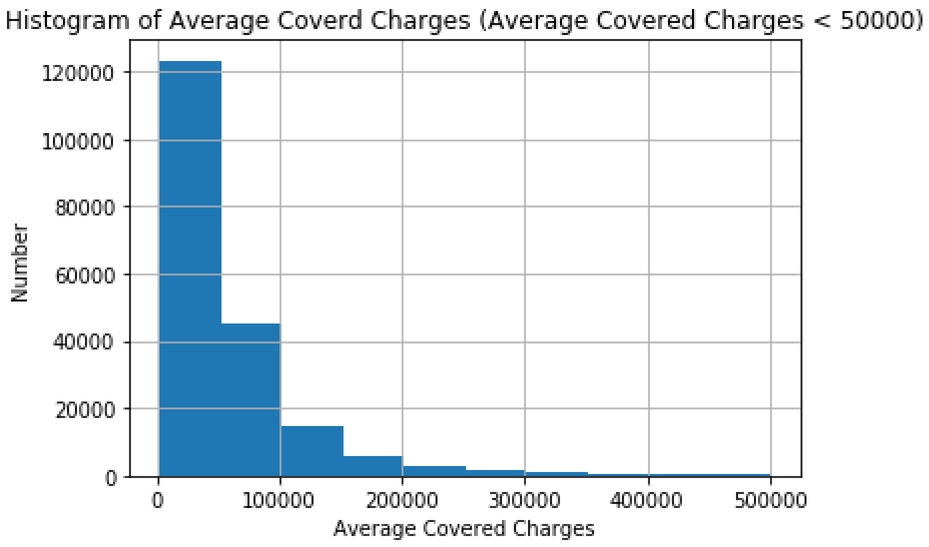
In [30]:

```
acc = df[df['Average Covered Charges'] < 500000]
acc = acc['Average Covered Charges'].dropna()

hist = acc.hist(bins = 10)
pl.title("Histogram of Average Coverd Charges (Average Covered Charges < 50000)")
pl.xlabel("Average Covered Charges")
pl.ylabel("Number")
hist
```

Out[30]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1e6a2ac2888>
```



In [27]:

```
acc_outlier = df[df['Average Covered Charges'] > 2000000]
acc_outlier
```

Out[27]:

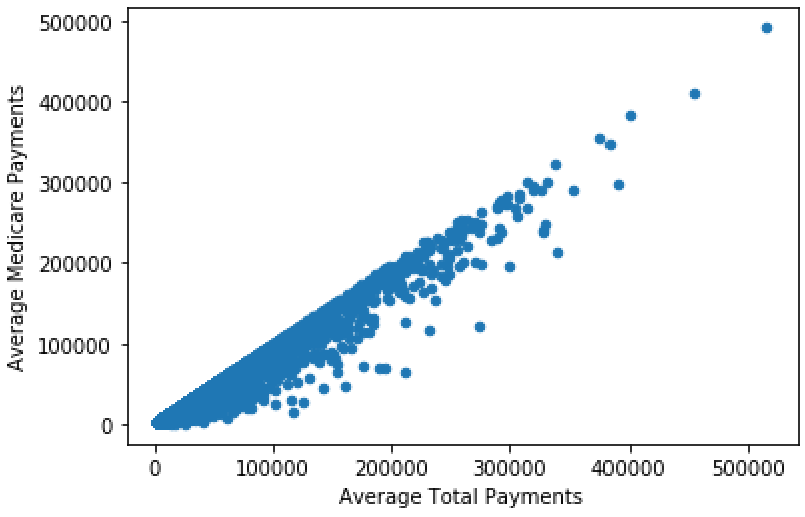
| | DRG Definition | Provider Id | Provider Name | Provider Street Address | Provider City | Provider State | Provide Zi Cod |
|---|---|---|---|---|---|---|---|
| 19446 | 003 - ECMO OR TRACH W MV >96 HRS OR PDX EXC FA... | 50380 | GOOD SAMARITAN HOSPITAL | 2425 SAMARITAN DRIVE | SAN JOSE | CA | 9512 |
| 20270 | 001 - HEART TRANSPLANT OR IMPLANT OF HEART ASS... | 50441 | STANFORD HEALTH CARE | 300 PASTEUR DRIVE | STANFORD | CA | 9430 |
| 20271 | 003 - ECMO OR TRACH W MV >96 HRS OR PDX EXC FA... | 50441 | STANFORD HEALTH CARE | 300 PASTEUR DRIVE | STANFORD | CA | 9430 |
| 23467 | 001 - HEART TRANSPLANT OR IMPLANT OF HEART ASS... | 50625 | CEDARS-SINAI MEDICAL CENTER | 8700 BEVERLY BLVD | LOS ANGELES | CA | 9004 |
| 23468 | 003 - ECMO OR TRACH W MV >96 HRS OR PDX EXC FA... | 50625 | CEDARS-SINAI MEDICAL CENTER | 8700 BEVERLY BLVD | LOS ANGELES | CA | 9004 |
| 51118 | 003 - ECMO OR TRACH W MV >96 HRS OR PDX EXC FA... | 110177 | DOCTORS HOSPITAL | 3651 WHEELER ROAD | AUGUSTA | GA | 3090 |
| 126008 | 001 - HEART TRANSPLANT OR IMPLANT OF HEART ASS... | 330234 | WESTCHESTER MEDICAL CENTER | 100 WOODS RD | VALHALLA | NY | 1059 |

Similar to Total Discharges, provider who has high Average Covered Charges basically distributed in places like California or New York, where the economy is more developed. It's also expected.

## (c)

In [35]:

```
# atp_amp = df[df['Average Total Payments'] < 400000]
atp_amp = df
scatter = df.plot.scatter(x='Average Total Payments', y ='Average Medicare Payments')
```



In [47]:

```
atmamp_outlier = df[(df['Average Total Payments'] > 500000) & (df['Average Medicare Payments'] >
400000)]
atmamp_outlier
```

Out[47]:

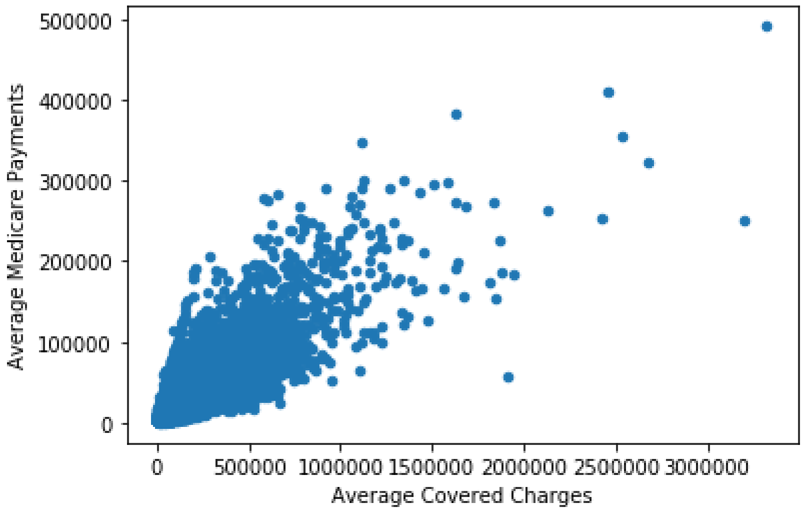
| | DRG Definition | Provider Id | Provider Name | Provider Street Address | Provider City | Provider State | Provider Zip Code | H R F Desc |
|---|---|---|---|---|---|---|---|---|
| 20270 | 001 - HEART TRANSPLANT OR IMPLANT OF HEART ASS... | 50441 | STANFORD HEALTH CARE | 300 PASTEUR DRIVE | STANFORD | CA | 94305 | C |

From the graph, we can see that there is an outlier who has the highest Average Total Payments and Average Medicare Payments at the top right of the plot. It's not surprising that this provider is located in California, who has the highest GDP in US.

## (d)

In [45]:

```
acc_amp = df
scatter = acc_amp.plot.scatter(x='Average Covered Charges', y ='Average Medicare Payments')
```



In [43]:

```
accamp_outlier = df[(df['Average Covered Charges'] > 1500000) & (df['Average Medicare Payments']
< 100000)]
accamp_outlier
```

Out[43]:

| | DRG Definition | Provider Id | Provider Name | Provider Street Address | Provider City | Provider State | Provider Zip Code | Hosp Refer Regi (HF Descripti |
|---|---|---|---|---|---|---|---|---|
| 116144 | 003 - ECMO OR TRACH W MV >96 HRS OR PDX EXC FA... | 310092 | CAPITAL HEALTH SYSTEM-FULD CAMPUS | 750 BRUNSWICK AVE | TRENTON | NJ | 8638 | F Philadelp |

As is shown above, there is an outlier in the lower middle of the plot.
After filtering, we can see that this provider's Average Total Charges is about 2,000,000 and its Average Medicare Payments is about 58,000, while other provider with this Average Total Charges basically have Average Medicare Payment more than 150,000.This provider is located in Trenton, New Jersey. It's quite surprising and it's hard to explain why this provider's Average Medicare Payments is this low by its location

## 2 Data Processing

## 2.1 Feature Creation

Create a function called get_100DRG(), which can create a array of unique DRG definition and then sort it by its frequency to find the top 100 DRG charges.

In [14]:

```python
def get_100DRG():
    definition = df['DRG Definition']
    definition_unique = pd.unique(definition)
    maxNum = 0
    output = ""
    dict_DRG = {}
    for d in definition_unique:
        temp_df = df[df['DRG Definition'] == d]
        dict_DRG[d] = len(temp_df)
    sorted_DRG = sorted(dict_DRG.items(), key=lambda kv: kv[1], reverse=True)
    DRG100 = sorted_DRG[:100]
    return DRG100

get_100DRG()
```

Out[14]:

```
[('871 - SEPTICEMIA OR SEVERE SEPSIS W/O MV >96 HOURS W MCC', 2838),
 ('291 - HEART FAILURE & SHOCK W MCC', 2742),
 ('190 - CHRONIC OBSTRUCTIVE PULMONARY DISEASE W MCC', 2687),
 ('470 - MAJOR JOINT REPLACEMENT OR REATTACHMENT OF LOWER EXTREMITY W/O MCC',
  2666),
 ('872 - SEPTICEMIA OR SEVERE SEPSIS W/O MV >96 HOURS W/O MCC', 2632),
 ('392 - ESOPHAGITIS, GASTROENT & MISC DIGEST DISORDERS W/O MCC', 2586),
 ('690 - KIDNEY & URINARY TRACT INFECTIONS W/O MCC', 2584),
 ('194 - SIMPLE PNEUMONIA & PLEURISY W CC', 2517),
 ('189 - PULMONARY EDEMA & RESPIRATORY FAILURE', 2465),
 ('603 - CELLULITIS W/O MCC', 2464),
 ('683 - RENAL FAILURE W CC', 2426),
 ('292 - HEART FAILURE & SHOCK W CC', 2414),
 ('193 - SIMPLE PNEUMONIA & PLEURISY W MCC', 2407),
 ('641 - MISC DISORDERS OF NUTRITION,METABOLISM,FLUIDS/ELECTROLYTES W/O MCC',
  2364),
 ('378 - G.I. HEMORRHAGE W CC', 2321),
 ('191 - CHRONIC OBSTRUCTIVE PULMONARY DISEASE W CC', 2202),
 ('682 - RENAL FAILURE W MCC', 2173),
 ('309 - CARDIAC ARRHYTHMIA & CONDUCTION DISORDERS W CC', 2095),
 ('065 - INTRACRANIAL HEMORRHAGE OR CEREBRAL INFARCTION W CC OR TPA IN 24 HRS',
  2090),
 ('689 - KIDNEY & URINARY TRACT INFECTIONS W MCC', 2041),
 ('481 - HIP & FEMUR PROCEDURES EXCEPT MAJOR JOINT W CC', 2001),
 ('280 - ACUTE MYOCARDIAL INFARCTION, DISCHARGED ALIVE W MCC', 1949),
 ('308 - CARDIAC ARRHYTHMIA & CONDUCTION DISORDERS W MCC', 1945),
 ('640 - MISC DISORDERS OF NUTRITION,METABOLISM,FLUIDS/ELECTROLYTES W MCC',
  1857),
 ('853 - INFECTIOUS & PARASITIC DISEASES W O.R. PROCEDURE W MCC', 1849),
 ('177 - RESPIRATORY INFECTIONS & INFLAMMATIONS W MCC', 1784),
 ('638 - DIABETES W CC', 1772),
 ('310 - CARDIAC ARRHYTHMIA & CONDUCTION DISORDERS W/O CC/MCC', 1770),
 ('312 - SYNCOPE & COLLAPSE', 1755),
 ('377 - G.I. HEMORRHAGE W MCC', 1736),
 ('812 - RED BLOOD CELL DISORDERS W/O MCC', 1716),
 ('208 - RESPIRATORY SYSTEM DIAGNOSIS W VENTILATOR SUPPORT <=96 HOURS', 1701),
 ('389 - G.I. OBSTRUCTION W CC', 1691),
 ('064 - INTRACRANIAL HEMORRHAGE OR CEREBRAL INFARCTION W MCC', 1657),
 ('247 - PERC CARDIOVASC PROC W DRUG-ELUTING STENT W/O MCC', 1565),
 ('698 - OTHER KIDNEY & URINARY TRACT DIAGNOSES W MCC', 1556),
 ('330 - MAJOR SMALL & LARGE BOWEL PROCEDURES W CC', 1527),
 ('287 - CIRCULATORY DISORDERS EXCEPT AMI, W CARD CATH W/O MCC', 1520),
 ('281 - ACUTE MYOCARDIAL INFARCTION, DISCHARGED ALIVE W CC', 1511),
 ('293 - HEART FAILURE & SHOCK W/O CC/MCC', 1415),
 ('069 - TRANSIENT ISCHEMIA', 1412),
 ('391 - ESOPHAGITIS, GASTROENT & MISC DIGEST DISORDERS W MCC', 1394),
 ('313 - CHEST PAIN', 1378),
 ('460 - SPINAL FUSION EXCEPT CERVICAL W/O MCC', 1373),
 ('192 - CHRONIC OBSTRUCTIVE PULMONARY DISEASE W/O CC/MCC', 1369),
 ('394 - OTHER DIGESTIVE SYSTEM DIAGNOSES W CC', 1359),
 ('552 - MEDICAL BACK PROBLEMS W/O MCC', 1359),
 ('483 - MAJOR JOINT/LIMB REATTACHMENT PROCEDURE OF UPPER EXTREMITIES', 1354),
 ('314 - OTHER CIRCULATORY SYSTEM DIAGNOSES W MCC', 1311),
 ('195 - SIMPLE PNEUMONIA & PLEURISY W/O CC/MCC', 1301),
 ('246 - PERC CARDIOVASC PROC W DRUG-ELUTING STENT W MCC OR 4+ VESSELS/STENTS',
  1253),
 ('101 - SEIZURES W/O MCC', 1231),
 ('066 - INTRACRANIAL HEMORRHAGE OR CEREBRAL INFARCTION W/O CC/MCC', 1225),
 ('202 - BRONCHITIS & ASTHMA W CC/MCC', 1190),
 ('286 - CIRCULATORY DISORDERS EXCEPT AMI, W CARD CATH W MCC', 1172),
 ('329 - MAJOR SMALL & LARGE BOWEL PROCEDURES W MCC', 1172),
 ('390 - G.I. OBSTRUCTION W/O CC/MCC', 1168),
 ('637 - DIABETES W MCC', 1162),
 ('917 - POISONING & TOXIC EFFECTS OF DRUGS W MCC', 1162),
 ('176 - PULMONARY EMBOLISM W/O MCC', 1141),
 ('948 - SIGNS & SYMPTOMS W/O MCC', 1126),
 ('870 - SEPTICEMIA OR SEVERE SEPSIS W MV >96 HOURS', 1077),
 ('252 - OTHER VASCULAR PROCEDURES W MCC', 1051),
 ('480 - HIP & FEMUR PROCEDURES EXCEPT MAJOR JOINT W MCC', 1051),
 ('300 - PERIPHERAL VASCULAR DISORDERS W CC', 1036),
 ('178 - RESPIRATORY INFECTIONS & INFLAMMATIONS W CC', 1035),
 ('699 - OTHER KIDNEY & URINARY TRACT DIAGNOSES W CC', 1017),
 ('372 - MAJOR GASTROINTESTINAL DISORDERS & PERITONEAL INFECTIONS W CC', 1006),
 ('305 - HYPERTENSION W/O MCC', 991),
 ('602 - CELLULITIS W MCC', 982),
 ('981 - EXTENSIVE O.R. PROCEDURE UNRELATED TO PRINCIPAL DIAGNOSIS W MCC',
  959),
 ('469 - MAJOR HIP AND KNEE JOINT REPLACEMENT OR REATTACHMENT OF LOWER EXTREM',
  943),
 ('439 - DISORDERS OF PANCREAS EXCEPT MALIGNANCY W CC', 937),
 ('175 - PULMONARY EMBOLISM W MCC', 905),
 ('253 - OTHER VASCULAR PROCEDURES W CC', 898),
 ('057 - DEGENERATIVE NERVOUS SYSTEM DISORDERS W/O MCC', 885),
 ('811 - RED BLOOD CELL DISORDERS W MCC', 881),
 ('100 - SEIZURES W MCC', 873),
 ('243 - PERMANENT CARDIAC PACEMAKER IMPLANT W CC', 870),
 ('331 - MAJOR SMALL & LARGE BOWEL PROCEDURES W/O CC/MCC', 855),
 ('897 - ALCOHOL/DRUG ABUSE OR DEPENDENCE W/O REHABILITATION THERAPY W/O MCC',
  849),
 ('039 - EXTRACRANIAL PROCEDURES W/O CC/MCC', 841),
 ('536 - FRACTURES OF HIP & PELVIS W/O MCC', 839),
 ('393 - OTHER DIGESTIVE SYSTEM DIAGNOSES W MCC', 835),
 ('563 - FX, SPRN, STRN & DISL EXCEPT FEMUR, HIP, PELVIS & THIGH W/O MCC',
  810),
 ('166 - OTHER RESP SYSTEM O.R. PROCEDURES W MCC', 799),
 ('482 - HIP & FEMUR PROCEDURES EXCEPT MAJOR JOINT W/O CC/MCC', 799),
 ('282 - ACUTE MYOCARDIAL INFARCTION, DISCHARGED ALIVE W/O CC/MCC', 792),
 ('418 - LAPAROSCOPIC CHOLECYSTECTOMY W/O C.D.E. W CC', 775),
 ('092 - OTHER DISORDERS OF NERVOUS SYSTEM W CC', 722),
 ('207 - RESPIRATORY SYSTEM DIAGNOSIS W VENTILATOR SUPPORT >96 HOURS', 675),
 ('918 - POISONING & TOXIC EFFECTS OF DRUGS W/O MCC', 671),
 ('242 - PERMANENT CARDIAC PACEMAKER IMPLANT W MCC', 661),
 ('473 - CERVICAL SPINAL FUSION W/O CC/MCC', 643),
 ('315 - OTHER CIRCULATORY SYSTEM DIAGNOSES W CC', 622),
 ('270 - OTHER MAJOR CARDIOVASCULAR PROCEDURES W MCC', 620),
 ('180 - RESPIRATORY NEOPLASMS W MCC', 618),
 ('371 - MAJOR GASTROINTESTINAL DISORDERS & PERITONEAL INFECTIONS W MCC', 616),
 ('269 - AORTIC AND HEART ASSIST PROCEDURES EXCEPT PULSATION BALLOON W/O MCC',
  608),
 ('299 - PERIPHERAL VASCULAR DISORDERS W MCC', 608)]
```

Create a function called tokenize_word(word), which is used to extract DRG number in a DRG definition.

In [38]:

```
import re
def tokenize_word(word):
    pattern =  re.compile(r"\d{3}")
    return re.match(pattern, word).group(0)
```

## 2.2 Transforming Data

In [103]:

```
import numpy as np

newdf = df[df['DRG Definition'].isin(keywords)]

column = ['Provider Id', 'Provider State']

for word in keywords:
    w = tokenize_word(word)
    column.append("DRG Charges %s"%w)
df100DRG = pd.DataFrame(columns=column)
idx = list(df100DRG.columns)

ids = pd.unique(newdf['Provider Id'])
for id in ids:
    row = []
    temp_df = newdf[newdf['Provider Id'] == id]
    row.append(id)
    row.append(temp_df.iloc[0,5])
    for x in keywords:
        if x not in list(temp_df['DRG Definition']):
            row.append(np.nan)
        else:
            acc = temp_df[temp_df['DRG Definition'] == x].iloc[0,9]
            row.append(acc)
    df100DRG = df100DRG.append(pd.Series(row, index = idx), ignore_index=True)



# save dataframe into a csv file
df100DRG.to_csv("100DRG.csv", index = False)
print("Done.")
```

```
['Provider Id', 'Provider State', 'DRG Charges 039', 'DRG Charges 057', 'DRG Charg
es 064', 'DRG Charges 065', 'DRG Charges 066', 'DRG Charges 069', 'DRG Charges 09
2', 'DRG Charges 100', 'DRG Charges 101', 'DRG Charges 166', 'DRG Charges 175', 'D
RG Charges 176', 'DRG Charges 177', 'DRG Charges 178', 'DRG Charges 180', 'DRG Cha
rges 189', 'DRG Charges 190', 'DRG Charges 191', 'DRG Charges 192', 'DRG Charges 1
93', 'DRG Charges 194', 'DRG Charges 195', 'DRG Charges 202', 'DRG Charges 207',
'DRG Charges 208', 'DRG Charges 242', 'DRG Charges 243', 'DRG Charges 246', 'DRG C
harges 247', 'DRG Charges 252', 'DRG Charges 253', 'DRG Charges 269', 'DRG Charges
270', 'DRG Charges 280', 'DRG Charges 281', 'DRG Charges 282', 'DRG Charges 286',
'DRG Charges 287', 'DRG Charges 291', 'DRG Charges 292', 'DRG Charges 293', 'DRG C
harges 299', 'DRG Charges 300', 'DRG Charges 305', 'DRG Charges 308', 'DRG Charges
309', 'DRG Charges 310', 'DRG Charges 312', 'DRG Charges 313', 'DRG Charges 314',
'DRG Charges 315', 'DRG Charges 329', 'DRG Charges 330', 'DRG Charges 331', 'DRG C
harges 371', 'DRG Charges 372', 'DRG Charges 377', 'DRG Charges 378', 'DRG Charges
389', 'DRG Charges 390', 'DRG Charges 391', 'DRG Charges 392', 'DRG Charges 393',
'DRG Charges 394', 'DRG Charges 418', 'DRG Charges 439', 'DRG Charges 460', 'DRG C
harges 469', 'DRG Charges 470', 'DRG Charges 473', 'DRG Charges 480', 'DRG Charges
481', 'DRG Charges 482', 'DRG Charges 483', 'DRG Charges 536', 'DRG Charges 552',
'DRG Charges 563', 'DRG Charges 602', 'DRG Charges 603', 'DRG Charges 637', 'DRG C
harges 638', 'DRG Charges 640', 'DRG Charges 641', 'DRG Charges 682', 'DRG Charges
683', 'DRG Charges 689', 'DRG Charges 690', 'DRG Charges 698', 'DRG Charges 699',
'DRG Charges 811', 'DRG Charges 812', 'DRG Charges 853', 'DRG Charges 870', 'DRG C
harges 871', 'DRG Charges 872', 'DRG Charges 897', 'DRG Charges 917', 'DRG Charges
918', 'DRG Charges 948', 'DRG Charges 981']
```

Transformed data is stored in dataframe and then saved in "100DRG.csv".
The dataset is shown down below.

In [9]:

```
df2 = pd.read_csv("100DRG.csv")
df2
```

Out[9]:

| | Provider Id | Provider State | DRG Charges 039 | DRG Charges 057 | DRG Charges 064 | DRG Charges 065 | DRG Charges 066 | DRG Charges 069 | DI Charg 0 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 10001 | AL | 41130.56 | 25434.17 | 46240.00 | 33440.09 | 30566.49 | 30004.42 | 25485. |
| 1 | 10005 | AL | 14450.08 | NaN | 26866.23 | 14336.27 | 13158.84 | 16008.44 | N |
| 2 | 10006 | AL | 35486.58 | 24472.15 | 35674.07 | 24571.08 | 20953.62 | 17277.38 | N |
| 3 | 10007 | AL | NaN | NaN | NaN | NaN | NaN | NaN | N |
| 4 | 10008 | AL | NaN | NaN | NaN | NaN | NaN | NaN | N |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| 3164 | 670112 | TX | NaN | NaN | NaN | NaN | NaN | NaN | N |
| 3165 | 670116 | TX | NaN | NaN | NaN | NaN | NaN | NaN | N |
| 3166 | 670119 | TX | NaN | NaN | NaN | NaN | NaN | NaN | N |
| 3167 | 670120 | TX | NaN | NaN | NaN | NaN | NaN | NaN | N |
| 3168 | 670122 | TX | NaN | NaN | NaN | NaN | NaN | NaN | N |

3169 rows × 102 columns

## 2.3 Quality Control

### 2.3.1

**Common ways to handle missing items:**
1, drop all the rows with missing values
2, replace missing values by average or median score

### 2.3.2

In [107]:

```
df2[df2.duplicated()]
```

Out[107]:

| Provider Id | Provider State | DRG Charges 039 | DRG Charges 057 | DRG Charges 064 | DRG Charges 065 | DRG Charges 066 | DRG Charges 069 | DRG Charges 092 | Ch |
|---|---|---|---|---|---|---|---|---|---|

0 rows × 102 columns

There is no duplicated rows or columns in dataset.

# 3 Data Analysis & Interpretation

## 3.1 Correlation and Scatterplots

### (a)

In [2]:

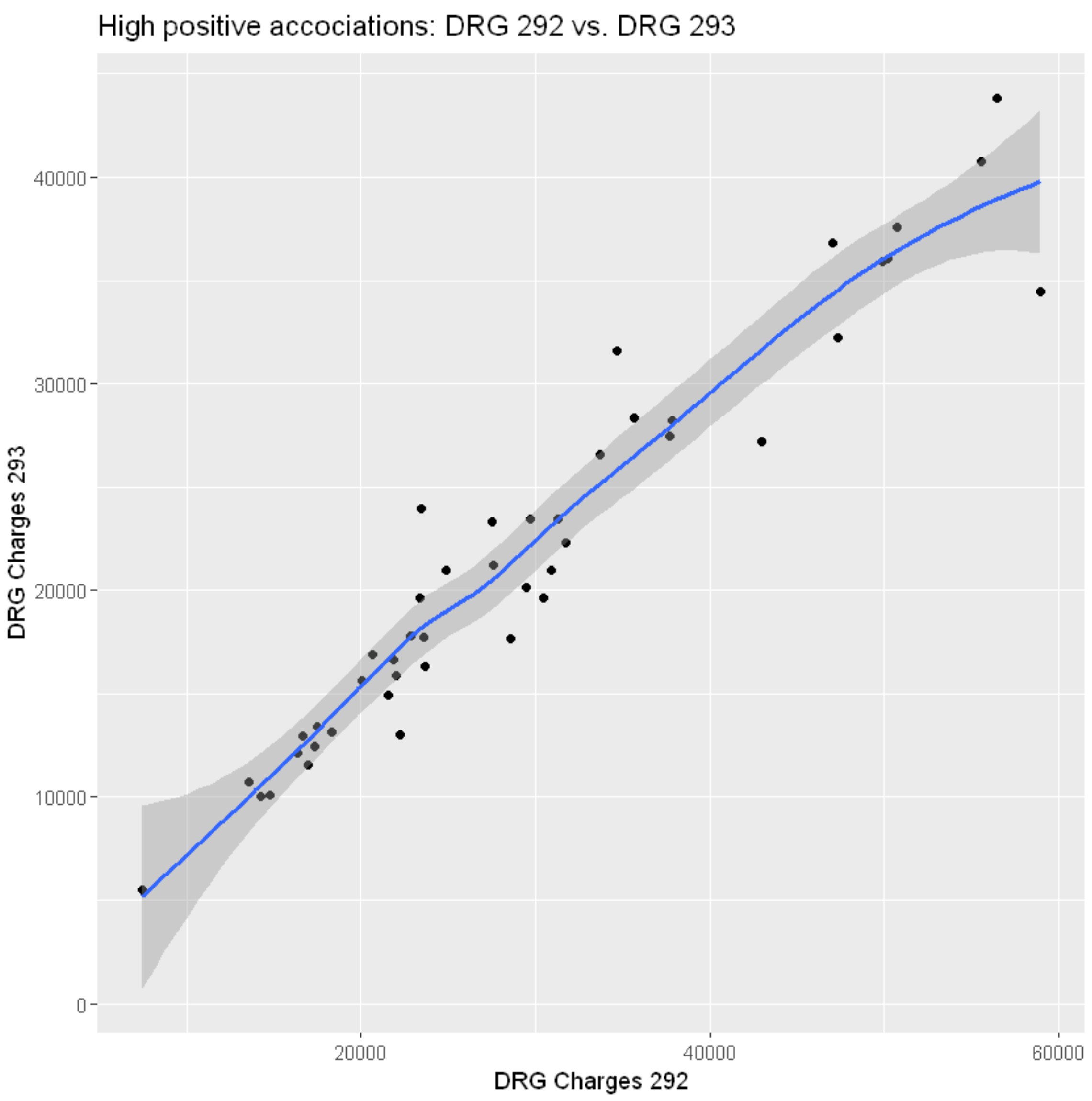
```
# install.packages("ggplot2")
library("ggplot2")
```

In [3]:

```
df = read.csv("100DRG.csv", sep = ',', header = TRUE)
df = na.omit(df, cols = c("df$DRG.Charges.292","DRG.Charges.293","DRG.Charges.481","DRG.Charges.
482",
                          "DRG.Charges.269","DRG.Charges.371","DRG.Charges.315","DRG.Charges.460"
))
```

In [4]:

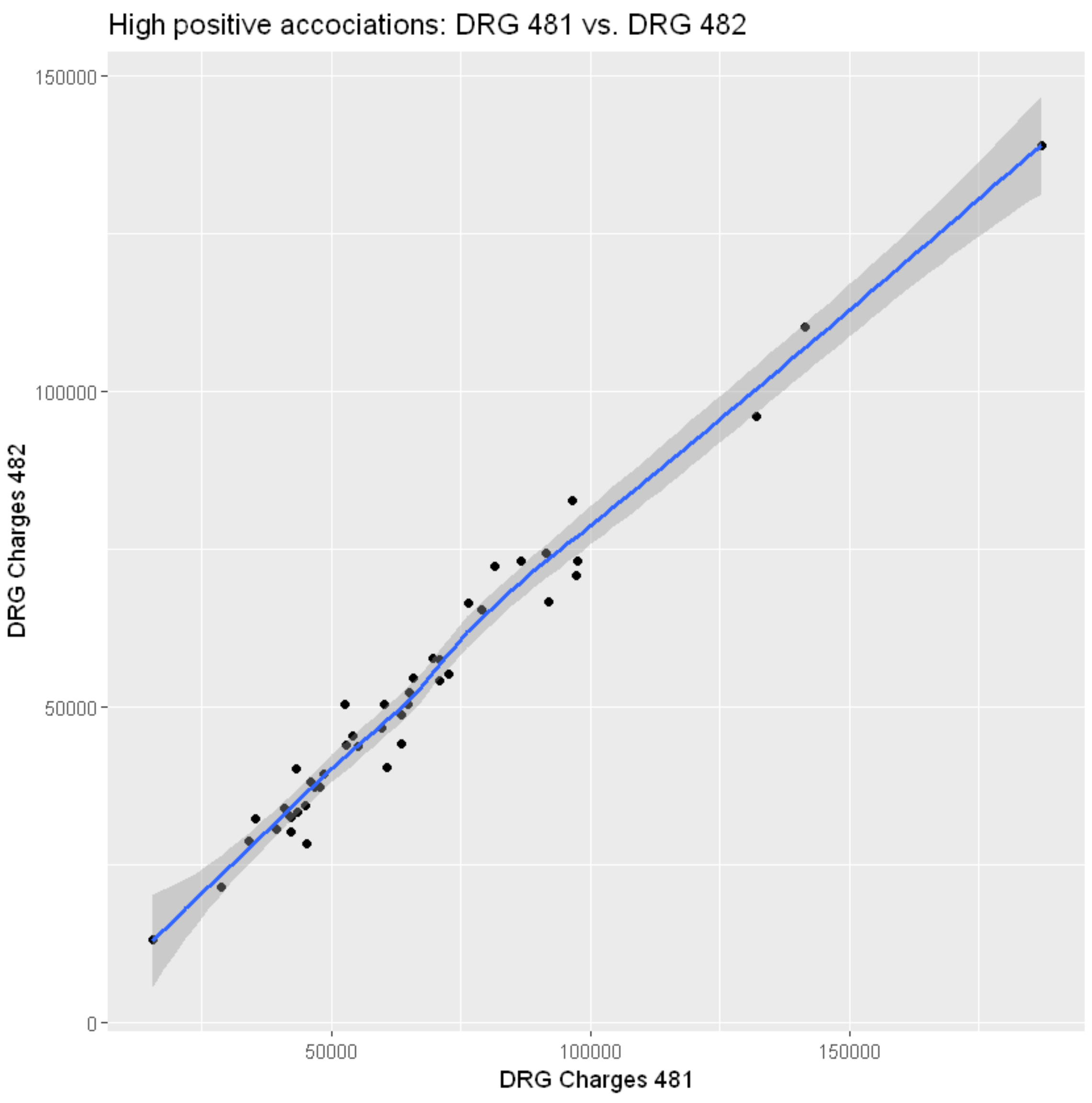
```
# max 1
ggplot(df, aes(x = df$DRG.Charges.292, y = df$DRG.Charges.293)) + geom_point() + geom_smooth(met
hod = 'loess')+
    labs(title="High positive accociations: DRG 292 vs. DRG 293",x="DRG Charges 292", y = "DRG C
harges 293")
```



High positive accociations: DRG 292 vs. DRG 293
DRG Charges 293
DRG Charges 292
0
10000
20000
30000
40000
20000
40000
60000

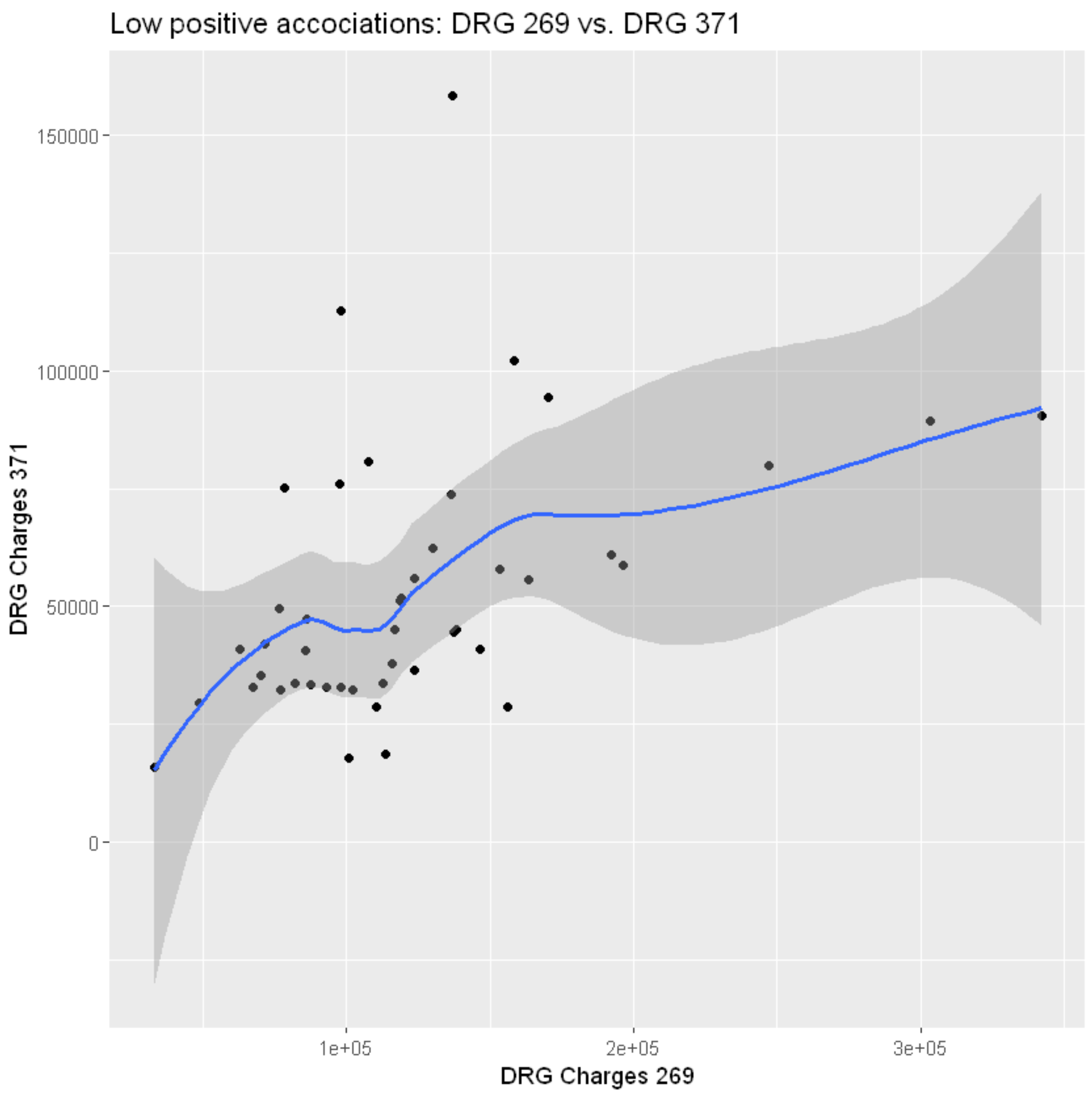
In [5]:

```
# max 2
ggplot(df, aes(x = df$DRG.Charges.481, y = df$DRG.Charges.482)) + geom_point() + geom_smooth(met
hod = 'loess')+
    labs(title="High positive accociations: DRG 481 vs. DRG 482",x="DRG Charges 481", y = "DRG C
harges 482")
```



In [7]:

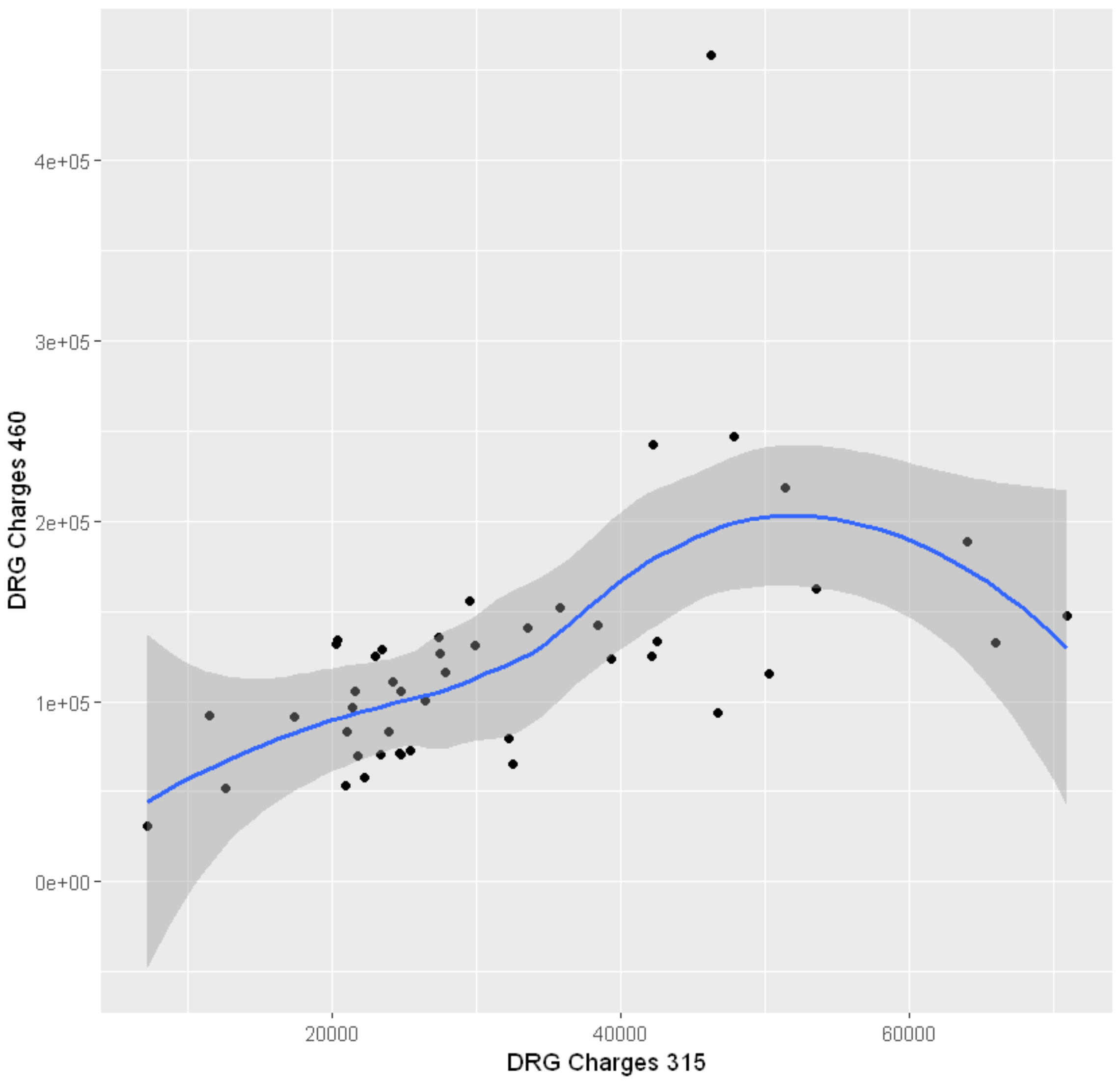
```
# min 1
ggplot(df, aes(x = df$DRG.Charges.269, y = df$DRG.Charges.371)) + geom_point() + geom_smooth(met
hod = 'loess')+
    labs(title="Low positive accociations: DRG 269 vs. DRG 371",x="DRG Charges 269", y = "DRG Ch
arges 371")
```



In [8]:

```
# min2
ggplot(df, aes(x = df$DRG.Charges.315, y = df$DRG.Charges.460)) + geom_point() + geom_smooth(met
hod = 'loess')+
    labs(title="Low positive accociations: DRG 315 vs. DRG 460",x="DRG Charges 315", y = "DRG Ch
arges 460")
```

Low positive accociations: DRG 315 vs. DRG 460



The observed relations are expeceted, given the DRG category names.
We can easily find out that pairs with high positive associations are basiclly with coherent DRG category numbers, such like 292 and 293. The coherent DRG category numbers means these pairs are more likely to come from a same region or have the same provider. While hospital pairs with low associations are usually come from different district and have different provider. Hence, So their DRG numbers are usually very far apart.

## (b)

In [142]:

```
corr = df2.iloc[:,2:].corr()
# corr.to_csv("corr.csv", index = False)
idx = list(corr.columns)
# find the minimun correlation
for row in range(corr.shape[0]): # df is the DataFrame
    for col in range(corr.shape[1]):
        if corr.iloc[row,col] == sorted(corr.min())[0]:
            print(sorted(corr.min())[0])
            print(idx[row], idx[col])
        if corr.iloc[row,col] == sorted(corr.min())[2]:
            print(sorted(corr.min())[2])
            print(idx[row], idx[col])

corr = corr.replace(1, 0)
# find the maximun correlation
for row in range(corr.shape[0]): # df is the DataFrame
    for col in range(corr.shape[1]):
        if corr.iloc[row,col] == sorted(corr.max(), reverse = True)[0]:
            print(sorted(corr.max(), reverse = True)[0])
            print(idx[row], idx[col])
        if corr.iloc[row,col] == sorted(corr.max(), reverse = True)[2]:
            print(sorted(corr.max(), reverse = True)[2])
            print(idx[row], idx[col])
```

```
0.5684269349479398
DRG Charges 269 DRG Charges 371
0.5835260020031648
DRG Charges 315 DRG Charges 460
0.5684269349479398
DRG Charges 371 DRG Charges 269
0.5835260020031648
DRG Charges 460 DRG Charges 315
0.9612236108383365
DRG Charges 292 DRG Charges 293
0.9612236108383365
DRG Charges 293 DRG Charges 292
0.977653142783413
DRG Charges 481 DRG Charges 482
0.977653142783413
DRG Charges 482 DRG Charges 481
```

For high positive association:
Correlation between DRG Charges 292 and DRG Charges 293 is 0.9612236108383365.
Correlation between DRG Charges 481 and DRG Charges 482 is 0.977653142783413.
From the first two plot, we can see that pairs of DRG Charges with high associations are almost linear, which is indicated by correlations.

For low positive association:
Correlation between DRG Charges 269 and DRG Charges 371 is 0.5684269349479398.
Correlation between DRG Charges 315 and DRG Charges 460 is 0.5835260020031648.
From the last two plot, we can see that these two pairs of DRG Charges's relationship is not very obvious. By using ggplot to add a smooth on it, we obtain a curve. While for the first two plot, what we get is almost a straight line. The correlations for these two pairs are much lower than the fisrt two pair and it support the observation.

## 3.2 Boxplots and T-tests

### (a)

Acccording to the GDP of different states, select CA, TX, GA, PA, IN, ME to exhibit differences in their hospital charges.

In [10]:

```
import pandas as pd

pd.unique(df2['Provider State'])
states = ["CA", "TX","GA","PA","IN","ME"]
df_6states = df2[df2['Provider State'].isin(states)]
df_6states.to_csv("DRG_6states.csv", index = False)
```

Preprocess data by pandas, which makes it easier to manipulate in ggplot2.

In [51]:

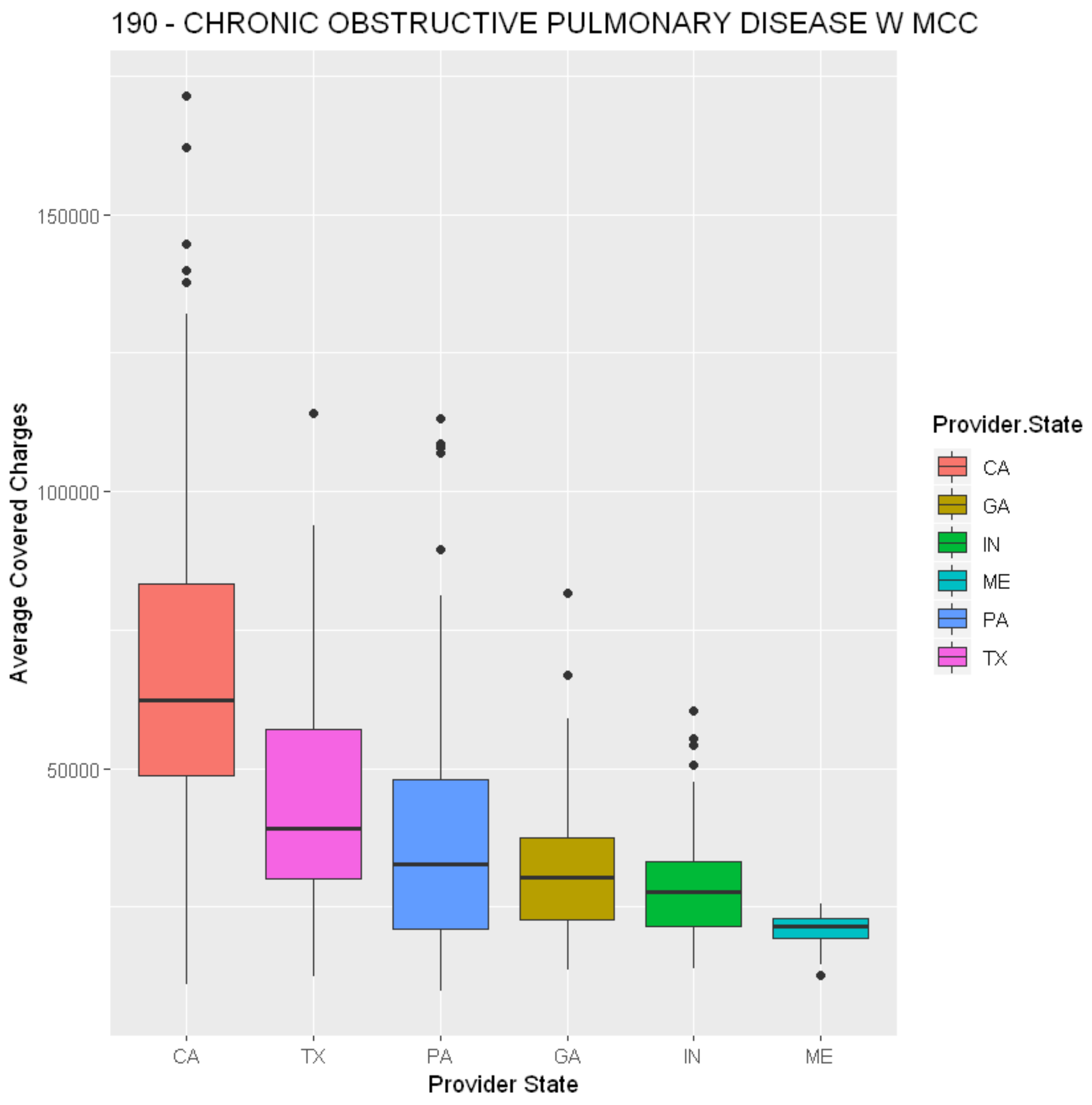
```
df_boxplot1 = df_6states[['DRG Charges 190','Provider State']]
df_boxplot1.dropna()
df_boxplot1.to_csv("df_boxplot1.csv", index = False)

df_boxplot2 = df_6states[['DRG Charges 392','Provider State']]
df_boxplot2.dropna()
df_boxplot2.to_csv("df_boxplot2.csv", index = False)

df_boxplot3 = df_6states[['DRG Charges 871','Provider State']]
df_boxplot3.dropna()
df_boxplot3.to_csv("df_boxplot3.csv", index = False)
```

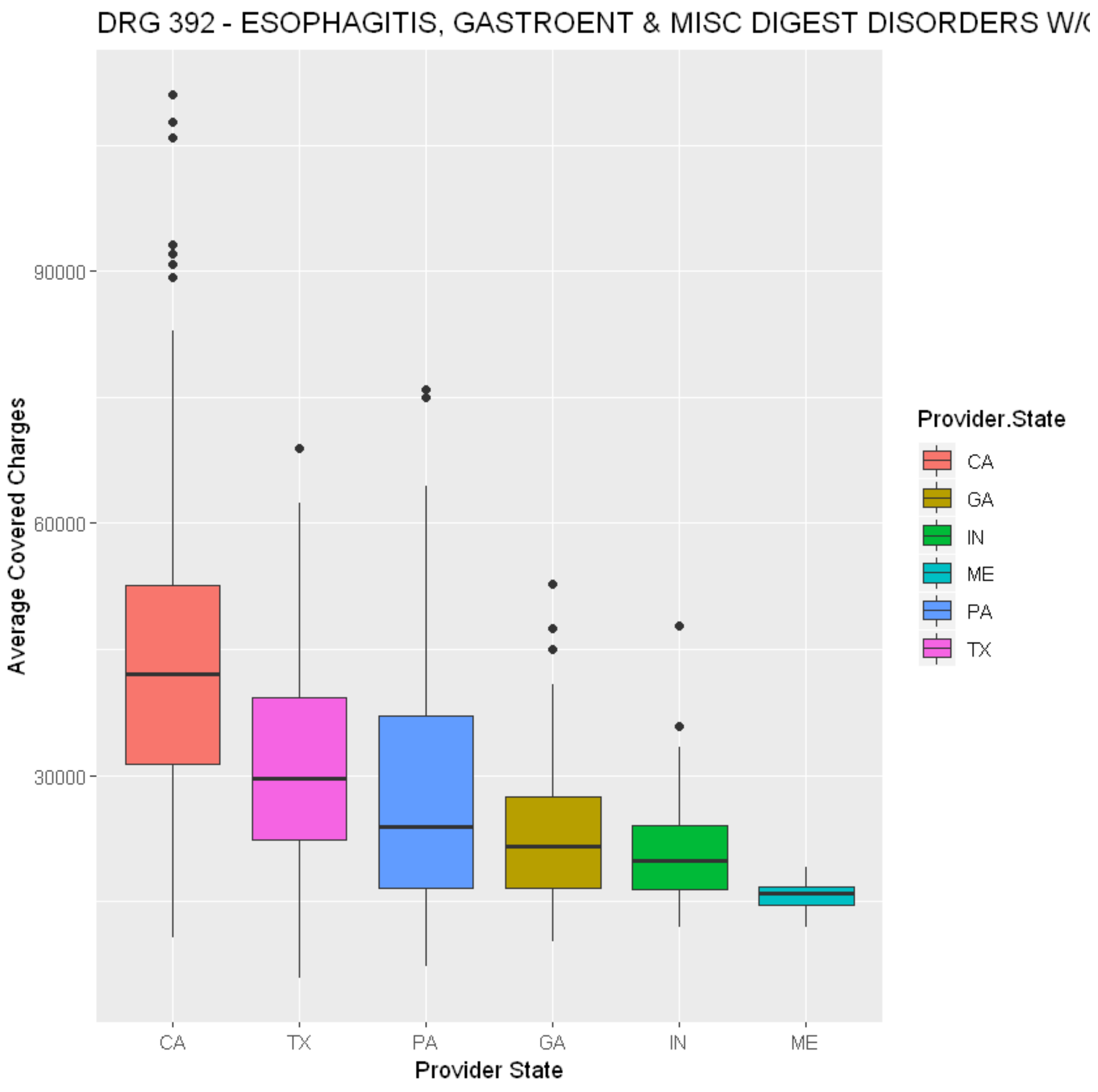
In [10]:

```
df_box1 = read.csv("df_boxplot1.csv", sep = ',', header = TRUE)
df_box1 = na.omit(df_box1)
p <- ggplot(df_box1, aes(x=Provider.State, y=DRG.Charges.190, fill = Provider.State)) +
    scale_x_discrete(limits=c("CA", "TX", "PA","GA","IN","ME"))+
    labs(title="190 - CHRONIC OBSTRUCTIVE PULMONARY DISEASE W MCC",x="Provider State", y = "Aver
age Covered Charges")+
    geom_boxplot()
p
```



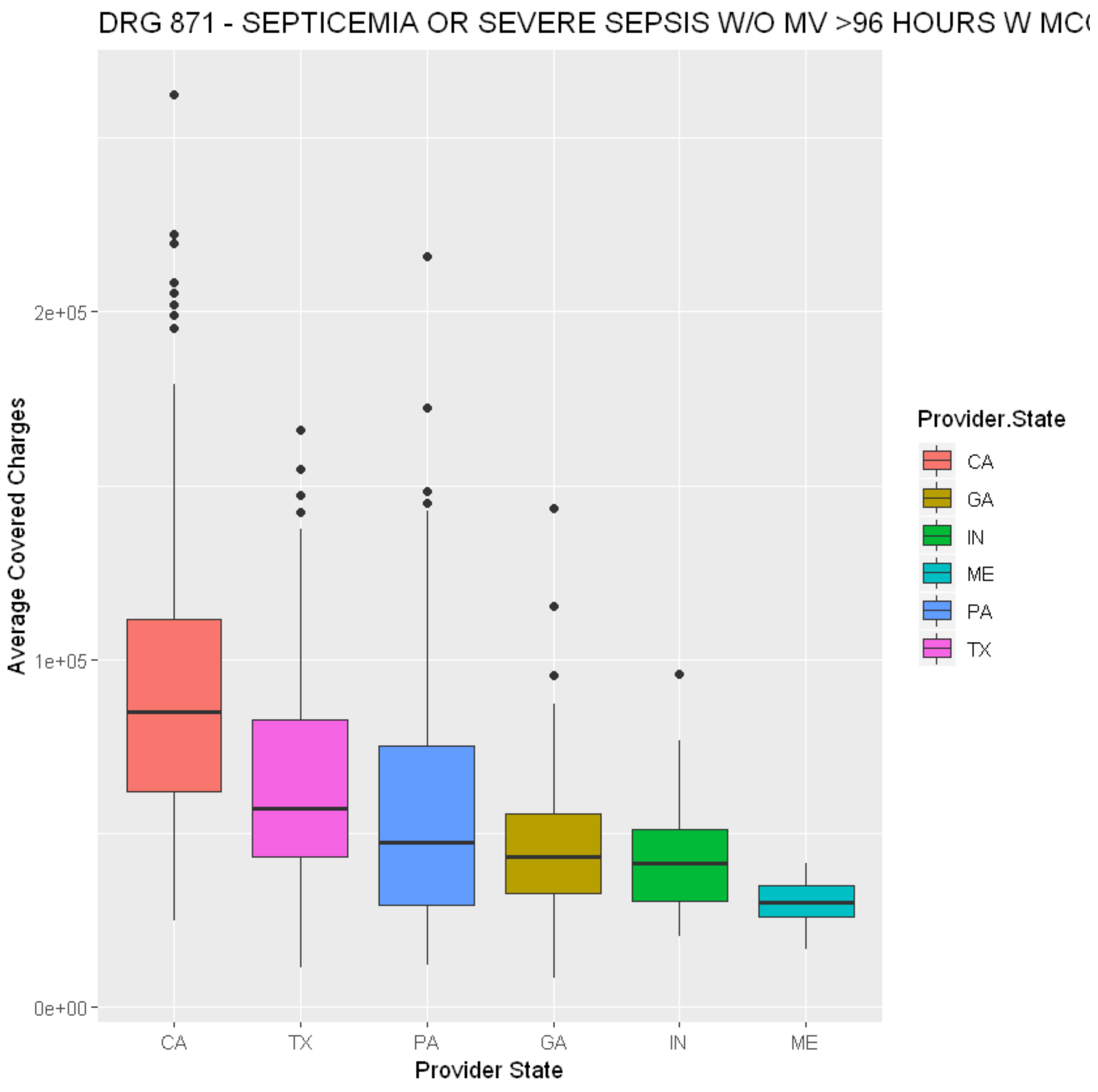
In [49]:

```
df_box2 = read.csv("df_boxplot2.csv", sep = ',', header = TRUE)
df_box2 = na.omit(df_box2)
p <- ggplot(df_box2, aes(x=Provider.State, y=DRG.Charges.392, fill = Provider.State)) +
    scale_x_discrete(limits=c("CA", "TX", "PA","GA","IN","ME"))+
    labs(title="DRG 392 - ESOPHAGITIS, GASTROENT & MISC DIGEST DISORDERS W/O MCC",x="Provider St
ate", y = "Average Covered Charges")+
    geom_boxplot()
p
```



In [47]:

```
df_box3 = read.csv("df_boxplot3.csv", sep = ',', header = TRUE)
df_box3 = na.omit(df_box3)
p <- ggplot(df_box3, aes(x=Provider.State, y=DRG.Charges.871, fill = Provider.State)) +
    scale_x_discrete(limits=c("CA", "TX", "PA","GA","IN","ME"))+
    labs(title="DRG 871 - SEPTICEMIA OR SEVERE SEPSIS W/O MV >96 HOURS W MCC",x="Provider State"
, y = "Average Covered Charges") +
    geom_boxplot()
p
```



## (b)

Hypothesis: California(CA) and Texas(TX) have the most significant difference in their charges in DRG Charges 190. And CA's average charges is greater than TX's average charges.

H0: Average charges in DRG Charges 190 in CA and TX are the same.
H1: CA's average charges in DRG Charges 190 is greater than TX's average charges.

In [17]:

```
import scipy.stats as st

df1_hypo = df_6states[['DRG Charges 190','Provider State']]
df1_CA = df1_hypo[df1_hypo['Provider State']=='CA']['DRG Charges 190']
df1_TX = df1_hypo[df1_hypo['Provider State']=='TX']['DRG Charges 190']
df1_CA = df1_CA.dropna()
df1_TX = df1_TX.dropna()

t,p = st.ttest_ind(df1_CA, df1_TX)
print("t statistic: " + str(t))
print("p-value: " + str(p))

# Proform one-sided test and use significant value 0.05
if p < 0.05/2:
    print("Reject H0.")
else:
    print("Accept H0.")
```

```
t statistic: 9.804105998394789
p-value: 1.3330879399800308e-20
Reject H0.
```

Proform **one-sided** test and use significant value **0.05**.
Since p-value < 0.05/2, null hypothesis H0 is rejected. Therefore, we tentatively conclude H1 to be the case, which support the claim.

## (c)

Hypothesis: Pennsylvania(PA) and Georgia(GA) have the significant difference in their charges in DRG Charges 190.

H0: Average charges in DRG Charges 190 in PA and GA are the same.
H1: Average charges in DRG Charges 190 in PA and GA are the different.

In [25]:

```
df2_hypo = df_6states[['Provider State','DRG Charges 190','DRG Charges 392','DRG Charges 871']]

df2_PA = df2_hypo[df2_hypo['Provider State']=='PA']
df2_PA = df2_PA.dropna()
df2_PA = pd.concat([df2_PA['DRG Charges 190'], df2_PA['DRG Charges 392'], df2_PA['DRG Charges 87
1']], ignore_index=True)

df2_GA = df2_hypo[df2_hypo['Provider State']=='GA']
df2_GA = df2_GA.dropna()
df2_GA = pd.concat([df2_GA['DRG Charges 190'], df2_GA['DRG Charges 392'], df2_GA['DRG Charges 87
1']], ignore_index=True)

length = min(len(df2_PA), len(df2_GA))
df2_GA = df2_GA.sample(n = length, random_state=3)
df2_PA = df2_PA.sample(n = length, random_state=3)

t_rel, p_rel = st.ttest_rel(df2_GA, df2_PA)
print("Two sample paired Student's t-test.")
print("t statistic(paired): " + str(t_rel))
print("p-value(paired): " + str(p_rel))
if p_rel < 0.05:
    print("Reject H0.")
else:
    print("Accept H0.")
```

```
Two sample paired Student's t-test.
t statistic(paired): -3.2213719626126287
p-value(paired): 0.0014578800424925055
Reject H0.
```

Proform **two sample paired Student's t-test** and use significant value **0.05**.
Since p-value < 0.05, null hypothesis H0 is rejected. Therefore, we tentatively conclude H1 to be the case, which support the claim.

In [24]:

```
t_ind, p_ind = st.ttest_ind(df2_GA, df2_PA)
print("\nTwo sample unpaired t-test(two sided).")
print("t statistic(unpaired): " + str(t_ind))
print("p-value(unpaired): " + str(p_ind))
if p_ind < 0.05:
    print("Reject H0.")
else:
    print("Accept H0.")
```

```
Two sample unpaired t-test(two sided).
t statistic(unpaired): -3.133704313100077
p-value(unpaired): 0.0018350152260808122
Reject H0.
```

Proform **two sample unpaired t-test(two sided)** and use significant value **0.05**.
Since p-value < 0.05, null hypothesis H0 is rejected. Therefore, we tentatively conclude H1 to be the case, which support the claim.

As is shown above, paired t-test gets p-value slightly less then unpaired t-test, which means it's more likely to reject H0.