Exploratory Data Analysis

Exploratory Data Analysis is an important step before applying the actual machine learning algorithms on the data. It gives a quick insight into the data, and helps us understand the data better. We shall look at 3 different ways to explore data today:

1. Visual Exploratory Analysis
2. Statistical Analysis
3. Descriptive Explorations

In [18]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import chi2_contingency
```

Reading in the Data

In [19]:
```
#Reading the input files
data=pd.read_csv('data/diabetic_data.csv')
features=pd.read_csv('data/feature_descriptions.csv')
mapping=pd.read_csv('data/IDs_mapping.csv')
```

In [46]:

```python
print data.describe()
#print data.info(verbose=True)
print features.describe()
print mapping.describe()
```

	encounter_id	patient_nbr	admission_type_id
count	1.017660e+05	1.017660e+05	101766.000000
mean	1.652016e+08	5.433040e+07	2.024006
std	1.026403e+08	3.869636e+07	1.445403
min	1.252200e+04	1.350000e+02	1.000000
25%	8.496119e+07	2.341322e+07	1.000000
50%	1.523890e+08	4.550514e+07	1.000000
75%	2.302709e+08	8.754595e+07	3.000000
max	4.438672e+08	1.895026e+08	8.000000

	discharge_disposition_id	admission_source_id	time_in_hospital
count	101766.000000	101766.000000	101766.000000
mean	3.715642	5.754437	4.395987
std	5.280166	4.064081	2.985108
min	1.000000	1.000000	1.000000
25%	1.000000	1.000000	2.000000
50%	1.000000	7.000000	4.000000
75%	4.000000	7.000000	6.000000
max	28.000000	25.000000	14.000000

	num_lab_procedures	num_procedures	num_medications	number_outpatient
count	101766.000000	101766.000000	101766.000000	101766.000000
mean	43.095641	1.339730	16.021844	0.369357
std	19.674362	1.705807	8.127566	1.267265
min	1.000000	0.000000	1.000000	0.000000
25%	31.000000	0.000000	10.000000	0.000000
50%	44.000000	1.000000	15.000000	0.000000
75%	57.000000	2.000000	20.000000	0.000000
max	132.000000	6.000000	81.000000	42.000000

	number_emergency	number_inpatient	number_diagnoses
count	101766.000000	101766.000000	101766.000000
mean	0.197836	0.635566	7.422607
std	0.930472	1.262863	1.933600
min	0.000000	0.000000	1.000000
25%	0.000000	0.000000	6.000000
50%	0.000000	0.000000	8.000000
75%	0.000000	1.000000	9.000000
max	76.000000	21.000000	16.000000

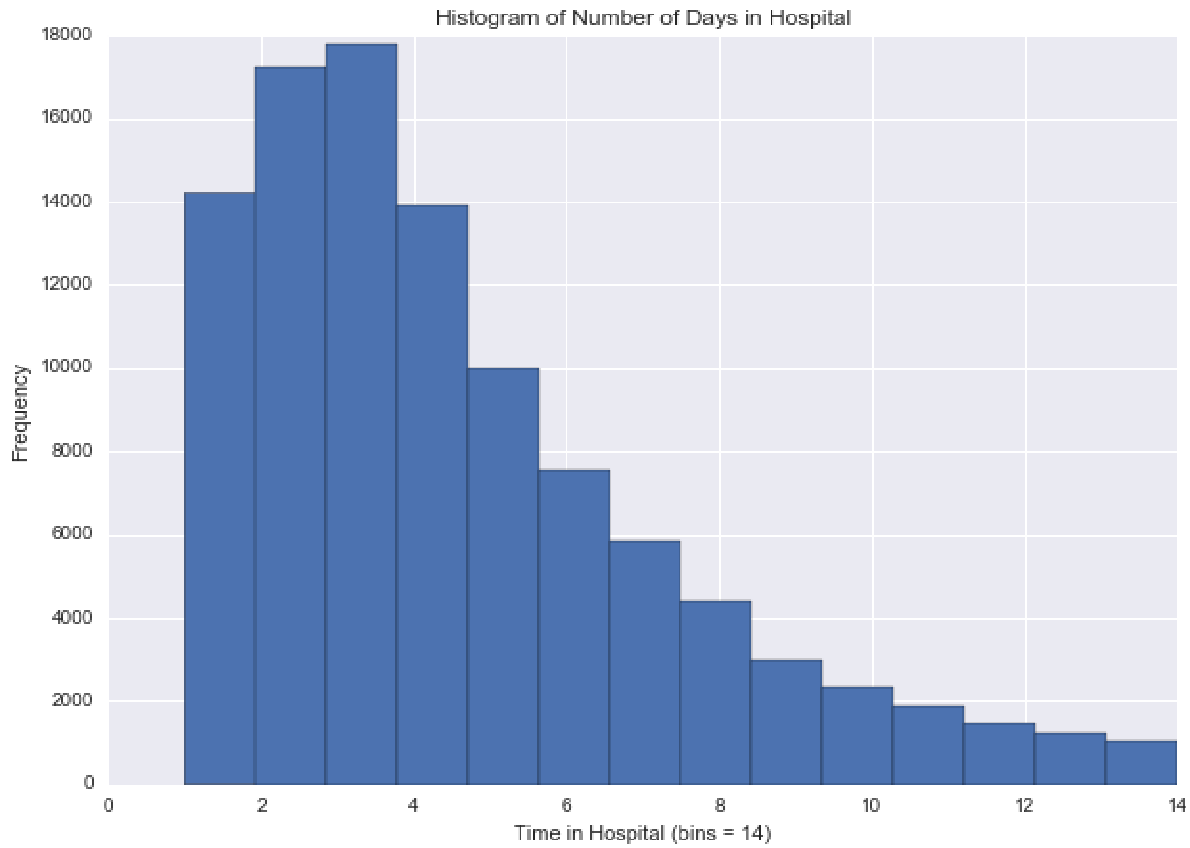
```
                       Feature name     Type          Description and v
alues  \
count                            28       28
   28
unique                           28        2
   28
top     Number of outpatient visits  Nominal  Unique identifier of a pa
tient
freq                              1       17
    1

       % missing
count         28
unique         6
top           0%
freq          23
       admission_type_id description
count                 65          62
unique                32          58
top                    8  Not Mapped
freq                   3           2
```

Visualizations

Visualizations are an important way of getting to know your data and building hypotheses. Following is an exploration of different fields in our data, to get a better understanding.

In [21]:
```
%matplotlib inline
bins = 14
h = data['time_in_hospital'].plot.hist(bins=bins, figsize=(10, 7),
                                                             title='Histogram of Number
 of Days in Hospital')
h.set_xlabel('Time in Hospital (bins = {})'.format(bins))
```

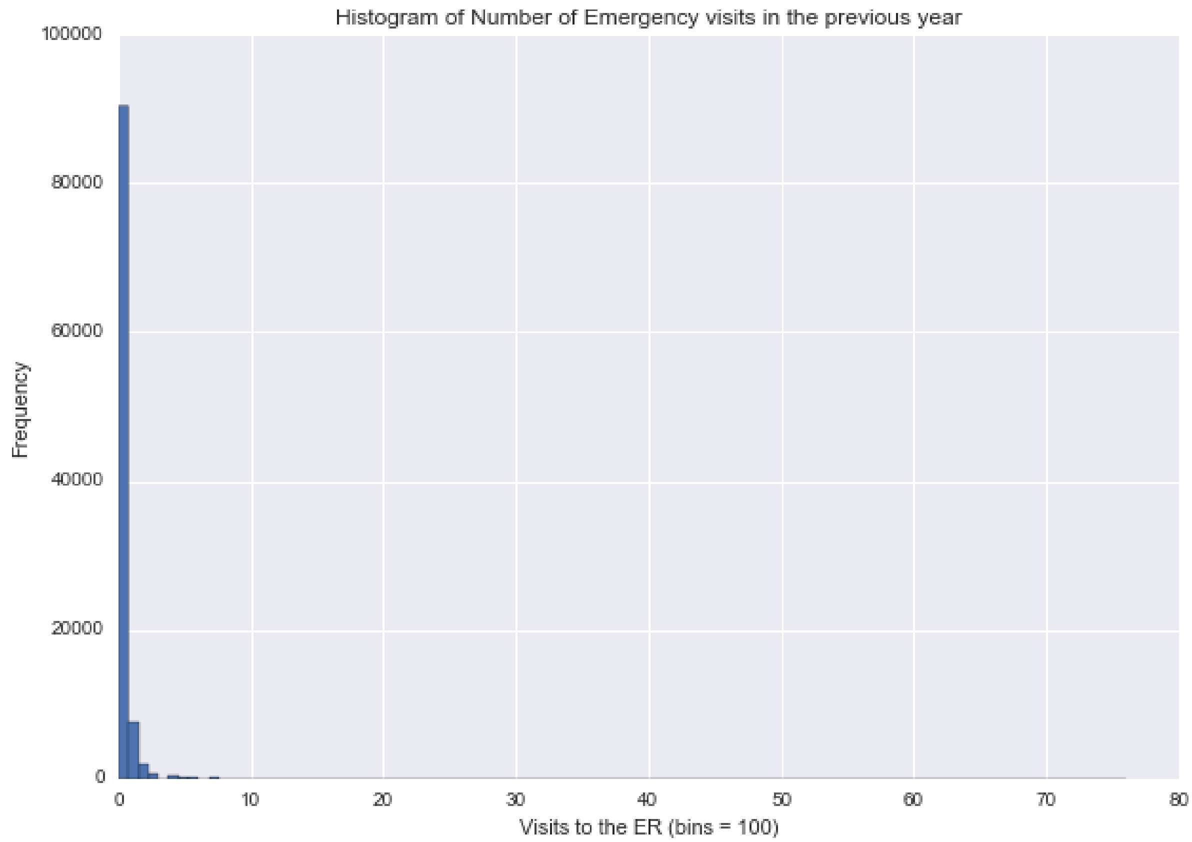
Out[21]: <matplotlib.text.Text at 0x1172d9450>



As can be observed from the visualisation, The number of days hospitalized seems to be the least at 13 days and the most at 4 days.

In [22]:
```
bins = 100
h = data['number_emergency'].plot.hist(bins=bins,figsize=(10, 7),
                      title='Histogram of Number of Emergency visits in
 the previous year')
h.set_xlabel('Visits to the ER (bins = {})'.format(bins))
```

Out[22]: <matplotlib.text.Text at 0x1152b3fd0>

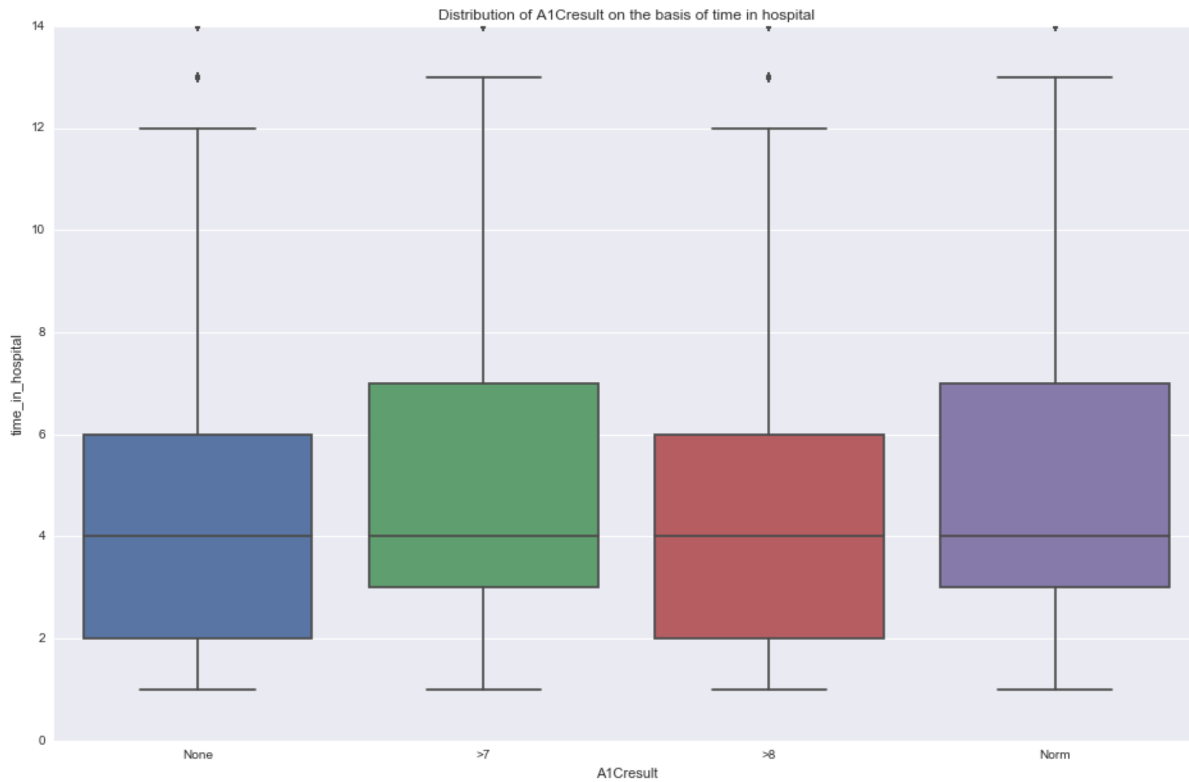


In [23]:
```
df2=data[['admission_type_id','num_lab_procedures','num_procedures','num
_medications']]
print df2.head()
```

```
   admission_type_id  num_lab_procedures  num_procedures  num_medicatio
ns
0                  6                  41               0
 1
1                  1                  59               0
 18
2                  1                  11               5
 13
3                  1                  44               1
 16
4                  1                  51               0
 8
```

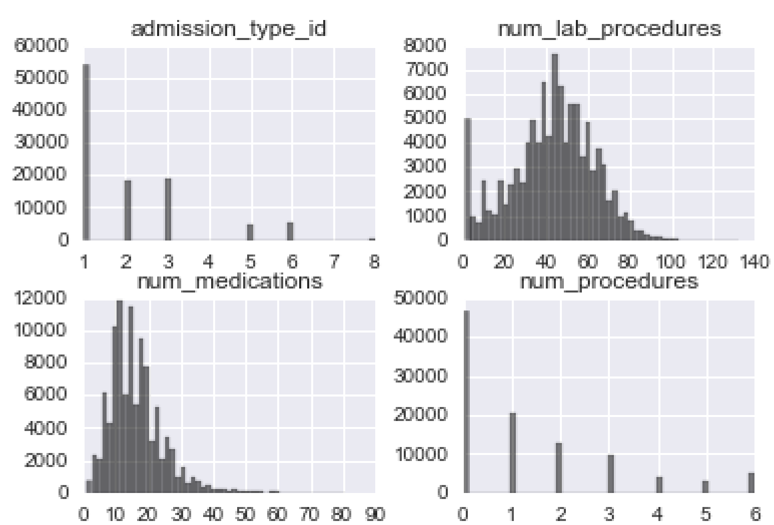
```
In [24]: plt.figure(figsize=(16, 10))
         ax = sns.boxplot(x = "A1Cresult", y = "time_in_hospital", data=data)
         ax.set(title='Distribution of A1Cresult on the basis of time in hospita
         l')
```

```
Out[24]: [<matplotlib.text.Text at 0x11b07ae10>]
```




```
In [25]: df2.hist(color='k', alpha=0.5, bins=50)
```

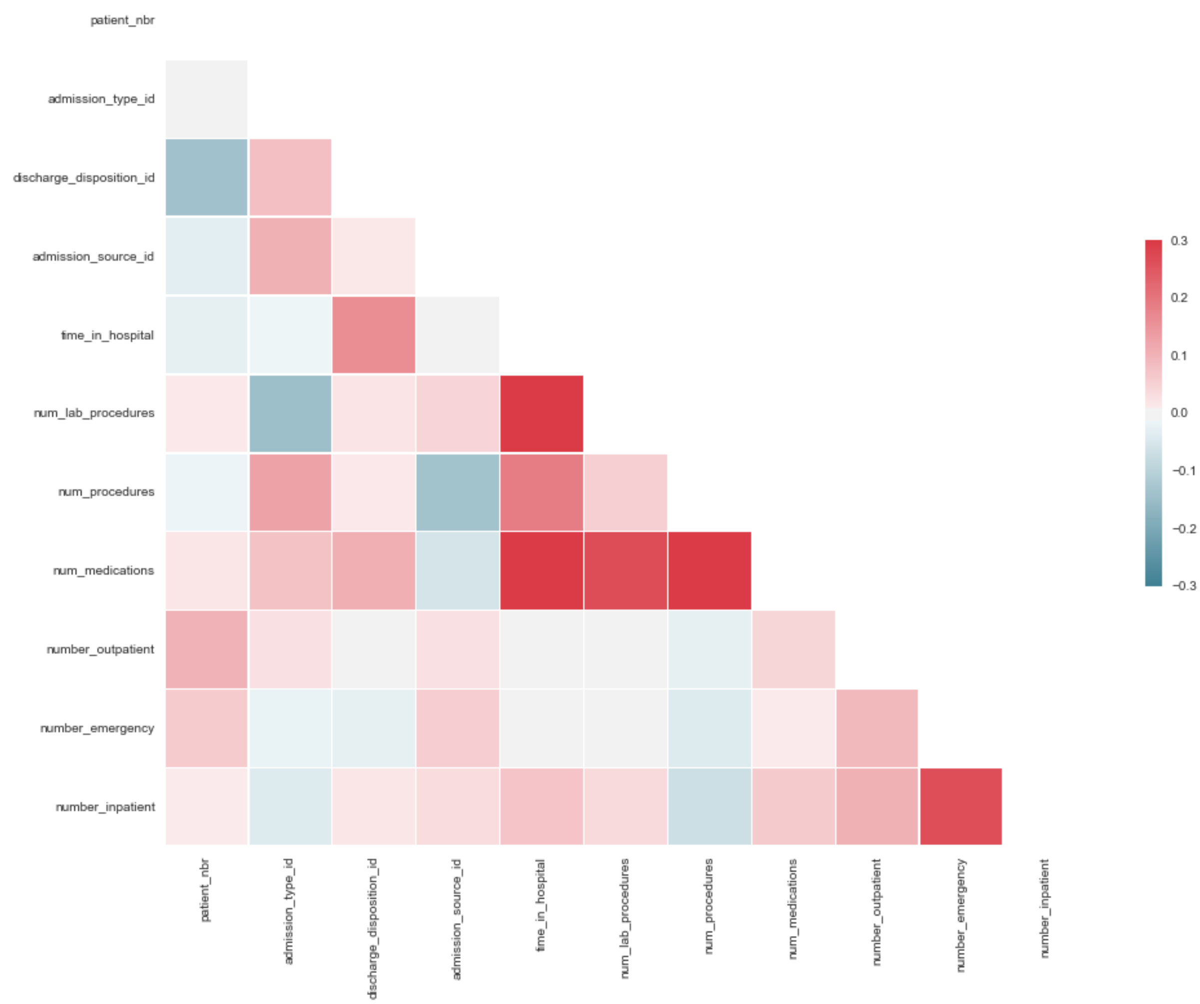
```
Out[25]: array([[<matplotlib.axes._subplots.AxesSubplot object at 0x11b0275d0>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x11afa1910>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x11af216d0>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x11ae82fd0
         >]], dtype=object)
```



The above visualisation Gives a quick glance into the frequency of the various numeric variables. The most common type of admission type seems to be 1, and the least 8 with barely any occurences of 4 and 7.

In [26]:
```
numerics = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']
diabetic_nums = data.select_dtypes(include=numerics).iloc[:, 1:-1]
corr = diabetic_nums.corr()
corr
sns.set(style='white')
mask = np.zeros_like(corr, dtype=bool)
mask[np.triu_indices_from(mask)] = True
f, ax = plt.subplots(figsize=(16, 12))
cmap = sns.diverging_palette(220, 10, as_cmap=True)
sns.heatmap(corr, mask=mask, cmap=cmap, vmax=.3, linewidths=0.5, cbar_kws={"shrink": 0.4}, ax=ax)
```

Out[26]: <matplotlib.axes._subplots.AxesSubplot at 0x117dbdd10>

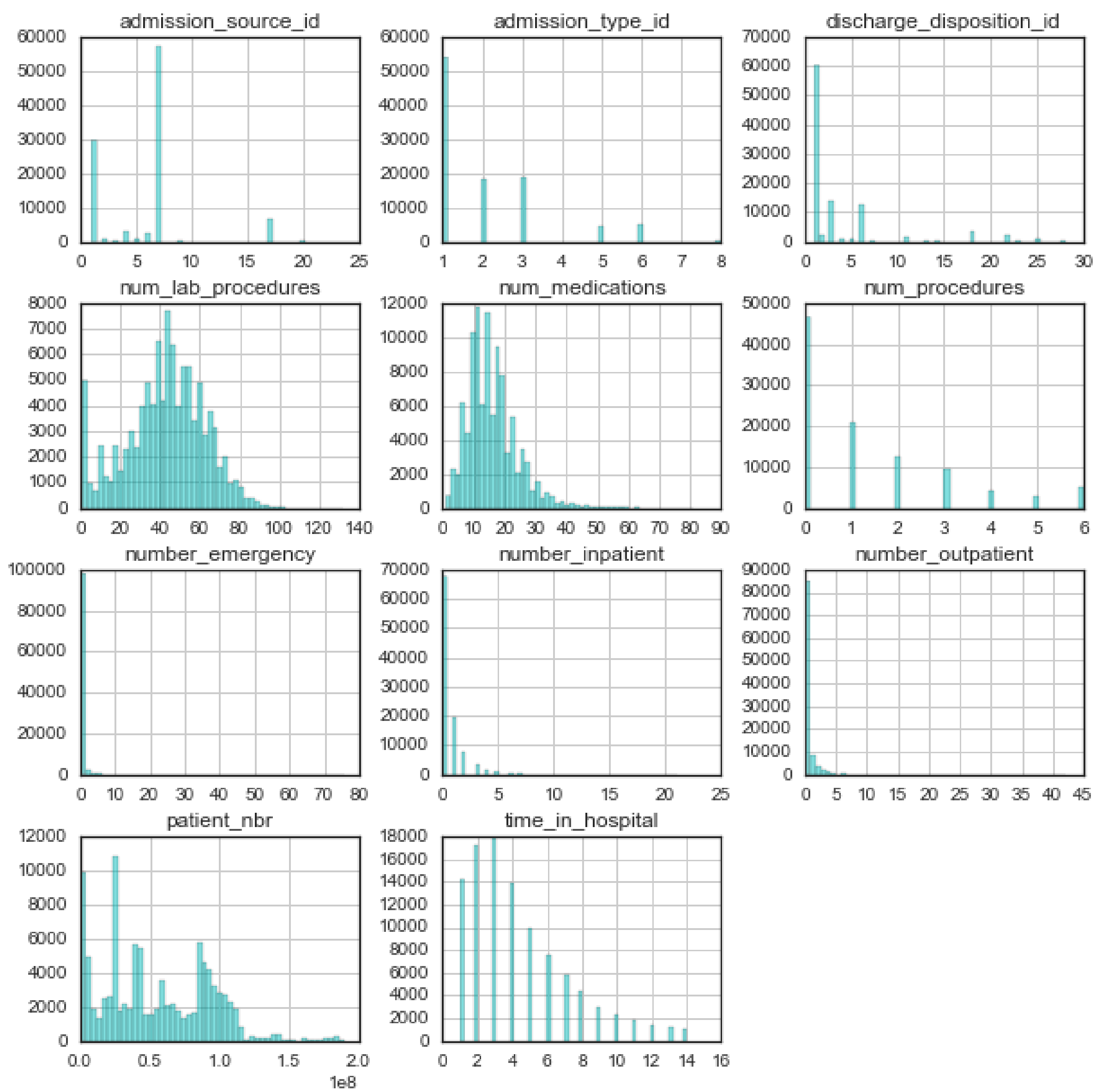


The Heatmap gives interesting insights into the data and is also a good way to formulate various hypotheses to test. For example, there seems to be a very high correlation between the number of days a patient is admitted in hospital and the amount of medications he is given. However, the number of procedures performed on a patient seems negatively correlated with the number of visits to inpatient the previous year.

The plot below shows a snapshot of all the numeric fields in our data.

```
In [27]: diabetic_nums.hist(color='c', alpha=0.5, bins=50,figsize=(10,10))
```

```
Out[27]: array([[<matplotlib.axes._subplots.AxesSubplot object at 0x117141a50>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x11dfed850>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x11e071510>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x11e0d3c10>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x11e155b90>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x11e0f7650>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x11e245c10>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x11e2c7a90>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x11e337410>],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x11e3b7390>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x11e41e150>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x11e4a0250
         >]], dtype=object)
```



Chi-2 Test for Independence: Statistical Analysis of Data

In this section we will look at A1Cresults and determine if A1c results and race/gender are related or independent. Hypotheses: A1Cresult and gender/race are independent

In [28]:
```
df3=data[['A1Cresult']]
print df3.head()
```

```
  A1Cresult
0      None
1      None
2      None
3      None
4      None
```

In [29]:
```
df3.groupby('A1Cresult').size()
```

Out[29]:
```
A1Cresult
>7       3812
>8       8216
None    84748
Norm     4990
dtype: int64
```

In [30]:
```
df3.loc[:,'Gender']=data.loc[:,'gender']
```

```
/Users/shraddhalanka/Library/Enthought/Canopy_64bit/User/lib/python2.7/
site-packages/pandas/core/indexing.py:284: 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: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
  self.obj[key] = _infer_fill_value(value)
/Users/shraddhalanka/Library/Enthought/Canopy_64bit/User/lib/python2.7/
site-packages/pandas/core/indexing.py:461: 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: http://pandas.pydata.org/pandas-d
ocs/stable/indexing.html#indexing-view-versus-copy
  self.obj[item] = s
```

In [31]:
```
print df3.head()
```

```
  A1Cresult  Gender
0      None  Female
1      None  Female
2      None  Female
3      None    Male
4      None    Male
```

In [32]:
```
observed_tab = pd.crosstab(df3.A1Cresult, df3.Gender, margins=True)
observed_tab
```

Out[32]:

Gender	Female	Male	Unknown/Invalid	All
A1Cresult				
>7	1956	1856	0	3812
>8	4117	4099	0	8216
None	45931	38814	3	84748
Norm	2704	2286	0	4990
All	54708	47055	3	101766

In [33]:
```
contingency_table = observed_tab.iloc[0:5, 0:2]
contingency_table
```

Out[33]:

Gender	Female	Male
A1Cresult		
>7	1956	1856
>8	4117	4099
None	45931	38814
Norm	2704	2286
All	54708	47055

We now have a table that lists the number of people who are male/female and the different categories each of their A1c result belongs to. As we examine the set, it looks the distribution is roughly half in each category.

In [34]:
```
from scipy.stats import chi2_contingency
chi2_contingency(contingency_table)
```

Out[34]:
```
(60.17583732536157,
 2.664247701827162e-12,
 4,
 array([[  2049.33911146,   1762.66088854],
        [  4416.93865157,   3799.06134843],
        [ 45559.08788066,  39185.91211934],
        [  2682.6343563 ,   2307.3656437 ],
        [ 54708.        ,  47055.        ]]))
```

Since p-value is much less than zero (2.664247701827162e-12), we reject the null hypotheses that A1Cresult and gender are independent, and conclude that they are dependent.

Using the similar technique we have used above, let us now test for dependence between race and A1c results.

In [35]:
```
df4=data[['A1Cresult']]
print df4.head()
df4.groupby('A1Cresult').size()
df4.loc[:,'Race']=data.loc[:,'race']
print df4.head()
observed_tab = pd.crosstab(df4.A1Cresult, df4.Race, margins=True)
observed_tab
contingency_table = observed_tab.iloc[0:5, 0:2]
contingency_table

chi2_contingency(contingency_table)
```

```
  A1Cresult
0      None
1      None
2      None
3      None
4      None
  A1Cresult             Race
0      None        Caucasian
1      None        Caucasian
2      None  AfricanAmerican
3      None        Caucasian
4      None        Caucasian
```

Out[35]:
```
(23.683006027411224,
 9.2452027911037754e-05,
 4,
 array([[    67.82074198,    573.17925802],
        [   222.18963832,   1877.81036168],
        [  1856.9763534 ,  15694.0236466 ],
        [   126.0132663 ,   1064.9867337 ],
        [  2273.        ,  19210.        ]]))
```

Since p-value is much less than zero (9.2452027911037754e-05), we reject the null hypotheses that A1Cresult and race are independent, and conclude that they are dependent.

ANOVA

We could also check to see if there is any relationship between the number of days admitted in the hospital and the type of admit (reason): emergency, urgent, elective, newborn, and not available. This can be accomplished using the statistical technique of one way ANOVA which checks if the values across different categories is similar or not. Hypotheses: length of stay values across admission type is similar

We will first begin by subsetting the data into different samples. Type1 is the number of days each person admitted for type1 was admitted and so on.

In [36]:
```
Type1 = data[data['admission_type_id'] == 1]['time_in_hospital']
Type2 = data[data['admission_type_id'] == 2]['time_in_hospital']
Type3 = data[data['admission_type_id'] == 3]['time_in_hospital']
Type4 = data[data['admission_type_id'] == 4]['time_in_hospital']
Type5 = data[data['admission_type_id'] == 5]['time_in_hospital']
Type6 = data[data['admission_type_id'] == 6]['time_in_hospital']
Type7 = data[data['admission_type_id'] == 7]['time_in_hospital']
Type8 = data[data['admission_type_id'] == 8]['time_in_hospital']
```

We shall now perform oneway ANOVA on this sample.

In [37]:
```
from scipy.stats import f_oneway
f_oneway(Type1,Type2,Type3,Type4,Type5,Type6,Type7,Type8)
```

Out[37]:
```
F_onewayResult(statistic=43.63205640485306, pvalue=5.2393952454503473e-
62)
```

The p-value of 5.2393952454503473e-62 is < 0.05, so we reject the null hypothesis and conclude that the time spent in hospital between the 8 different types of admit groups is different across groups.

In [42]:
```
#The different Groups available for Medical Specialty
data.groupby('medical_specialty').size().sort_values().tail(n=10)
```

Out[42]:
```
medical_specialty
Radiologist                      1140
Orthopedics-Reconstructive       1233
Orthopedics                      1400
Nephrology                       1613
Surgery-General                  3099
Cardiology                       5352
Family/GeneralPractice           7440
Emergency/Trauma                 7565
InternalMedicine                14635
?                               49949
dtype: int64
```

In [39]:
```
#Taking the top 10 medical specialties
subset = data[data['medical_specialty'].isin(['InternalMedicine', 'Psych
iatry',
        'Emergency/Trauma','Pulmonology','Urology','Cardiology','Family/
GeneralPractice',
        'Surgery-General','Nephrology', 'Orthopedics','Orthopedics-Recon
structive',
         'Radiologist'])]
```

In [40]:
```
subset.head()
```

Out[40]:

	encounter_id	patient_nbr	race	gender	age	weight	admission_type_id	discha
9	15738	63555939	Caucasian	Female	[90-100)	?	3	3
12	40926	85504905	Caucasian	Female	[40-50)	?	1	3
13	42570	77586282	Caucasian	Male	[80-90)	?	1	6
17	84222	108662661	Caucasian	Female	[50-60)	?	1	1
26	236316	40523301	Caucasian	Male	[80-90)	?	1	3

5 rows × 50 columns

In [41]:
```
Type1 = data[data['medical_specialty'] == 'InternalMedicine']['time_in_hospital']
Type2 = data[data['medical_specialty'] == 'Psychiatry']['time_in_hospital']
Type3 = data[data['medical_specialty'] == 'Emergency/Trauma']['time_in_hospital']
Type4 = data[data['medical_specialty'] == 'Pulmonology']['time_in_hospital']
Type5 = data[data['medical_specialty'] == 'Urology']['time_in_hospital']
Type6 = data[data['medical_specialty'] == 'Cardiology']['time_in_hospital']
Type7 = data[data['medical_specialty'] == 'Family/GeneralPractice']['time_in_hospital']
Type8 = data[data['medical_specialty'] == 'Surgery-General']['time_in_hospital']
Type9 = data[data['medical_specialty'] == 'Nephrology']['time_in_hospital']
Type10 = data[data['medical_specialty'] == 'Orthopedics']['time_in_hospital']

from scipy.stats import f_oneway
f_oneway(Type1,Type2,Type3,Type4,Type5,Type6,Type7,Type8,Type9,Type10)
```

Out[41]:
```
F_onewayResult(statistic=130.04674253293857, pvalue=6.1300426046739457e-243)
```

The P-value is much less than 0.05 and approximately equal to zero. so we reject the null hypothesis and conclude that the time spent in hospital between the 8 different medical specialties is different across groups.

Conclusion

This exercise of Exploratory Data Analysis gave us an insight into the dataset at hand. Race, gender and age appear to have an influence on the A1c results, however, the length of stay has little across medical specialties and admission types seem to differ.