```
In [1]:
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the curren
```

```
In [3]:
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
%matplotlib inline
import sys
from sklearn.linear_model import LinearRegression
from sklearn import metrics
from sklearn import preprocessing
import statsmodels.api as sm
```

```
In [6]:
train=pd.read_csv('/users/sarafarhat/Desktop/train_data.csv.zip')
test=pd.read_csv('/users/sarafarhat/Desktop/test_data.csv.zip')
sub=pd.read_csv('/users/sarafarhat/Desktop/sample_sub.csv.zip')
```

# Data Exploration

```
In [7]: train.head()
```

Out[7]:

| | case_id | Hospital_code | Hospital_type_code | City_Code_Hospital | Hospital_region_code | Available Extra Rooms in Hospital | Department | Ward_Ty |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 8 | c | 3 | Z | 3 | radiotherapy | |
| 1 | 2 | 2 | c | 5 | Z | 2 | radiotherapy | |
| 2 | 3 | 10 | e | 1 | X | 2 | anesthesia | |
| 3 | 4 | 26 | b | 2 | Y | 2 | radiotherapy | |
| 4 | 5 | 26 | b | 2 | Y | 2 | radiotherapy | |

```
In [8]: train.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 318438 entries, 0 to 318437
Data columns (total 18 columns):
 #   Column                             Non-Null Count   Dtype
---  ------                             --------------   -----
 0   case_id                            318438 non-null  int64
 1   Hospital_code                      318438 non-null  int64
 2   Hospital_type_code                 318438 non-null  object
 3   City_Code_Hospital                 318438 non-null  int64
 4   Hospital_region_code               318438 non-null  object
 5   Available Extra Rooms in Hospital  318438 non-null  int64
 6   Department                         318438 non-null  object
 7   Ward_Type                          318438 non-null  object
 8   Ward_Facility_Code                 318438 non-null  object
 9   Bed Grade                          318325 non-null  float64
 10  patientid                          318438 non-null  int64
 11  City_Code_Patient                  313906 non-null  float64
 12  Type of Admission                  318438 non-null  object
 13  Severity of Illness                318438 non-null  object
 14  Visitors with Patient              318438 non-null  int64
 15  Age                                318438 non-null  object
 16  Admission_Deposit                  318438 non-null  float64
 17  Stay                               318438 non-null  object
dtypes: float64(3), int64(6), object(9)
memory usage: 43.7+ MB
```

```
In [9]: train.isna().sum()

Out[9]:
case_id                                 0
Hospital_code                           0
Hospital_type_code                      0
City_Code_Hospital                      0
Hospital_region_code                    0
Available Extra Rooms in Hospital       0
Department                              0
Ward_Type                               0
Ward_Facility_Code                      0
Bed Grade                             113
patientid                               0
City_Code_Patient                    4532
Type of Admission                       0
Severity of Illness                     0
Visitors with Patient                   0
Age                                     0
Admission_Deposit                       0
Stay                                    0
dtype: int64
```

we checked for missing values, as we can see most columns of the dataset don't have missing values.

this shows that the data is pre-processed and cleaned.

```
In [10]: train.head()
```

Out[10]:

| | case_id | Hospital_code | Hospital_type_code | City_Code_Hospital | Hospital_region_code | Available Extra Rooms in Hospital | Department | Ward_Ty |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 8 | c | 3 | Z | 3 | radiotherapy | |
| 1 | 2 | 2 | c | 5 | Z | 2 | radiotherapy | |
| 2 | 3 | 10 | e | 1 | X | 2 | anesthesia | |
| 3 | 4 | 26 | b | 2 | Y | 2 | radiotherapy | |
| 4 | 5 | 26 | b | 2 | Y | 2 | radiotherapy | |

```
In [11]: train['count']=1
```

```
In [12]: train.head()
```

Out[12]:

| | case_id | Hospital_code | Hospital_type_code | City_Code_Hospital | Hospital_region_code | Available Extra Rooms in Hospital | Department | Ward_Ty |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 8 | c | 3 | Z | 3 | radiotherapy | |
| 1 | 2 | 2 | c | 5 | Z | 2 | radiotherapy | |
| 2 | 3 | 10 | e | 1 | X | 2 | anesthesia | |
| 3 | 4 | 26 | b | 2 | Y | 2 | radiotherapy | |
| 4 | 5 | 26 | b | 2 | Y | 2 | radiotherapy | |

# Data Visualization

```
In [13]: train.Stay.value_counts()

Out[13]:
21-30                 87491
11-20                 78139
31-40                 55159
51-60                 35018
0-10                  23604
41-50                 11743
71-80                 10254
More than 100 Days     6683
81-90                  4838
91-100                 2765
61-70                  2744
Name: Stay, dtype: int64
```

this shows that LOS that ranges between 21-30 days is the most frequent LOS

LOS with 11-20 days is the 2nd most freq LOS

```
In [14]: train.Age.value_counts()

Out[14]:
41-50     63749
31-40     63639
51-60     48514
21-30     40843
71-80     35792
61-70     33687
11-20     16768
81-90      7890
0-10       6254
91-100     1302
Name: Age, dtype: int64
```

Age groups 41-50 and 31-40 are the most freq group hospitalized

```
In [15]: train['Severity of Illness'].describe()

Out[15]:
count       318438
unique           3
top       Moderate
freq        175843
Name: Severity of Illness, dtype: object
```

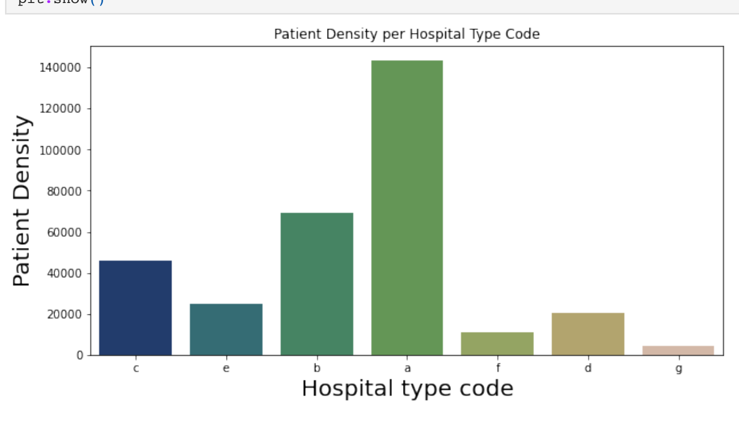
the most frequent severity of illness is "Moderate" severity

```
In [16]: train['Hospital_type_code'].describe()

Out[16]:
count     318438
unique         7
top            a
freq      143425
Name: Hospital_type_code, dtype: object
```

there are 7 unique hospital type codes with 'a' being the most frequent code which shows that the density of patient cases will be highest in hospital type code 'a'

# Bivariate Analysis

```
In [17]:
plt.figure(figsize = (10,5))
sns.countplot(x = 'Hospital_type_code', data = train, palette = 'gist_earth')
plt.xlabel('Hospital type code', size = 20)
plt.ylabel('Patient Density', size = 20)
plt.title('Patient Density per Hospital Type Code')
plt.show()
```




this confirms what I stated before : hospital with type code'a' has highest number of patients

thus, less beds/rooms left available. hospital with type code'g' has lowest number pf patients

thus, more beds/rooms left available.

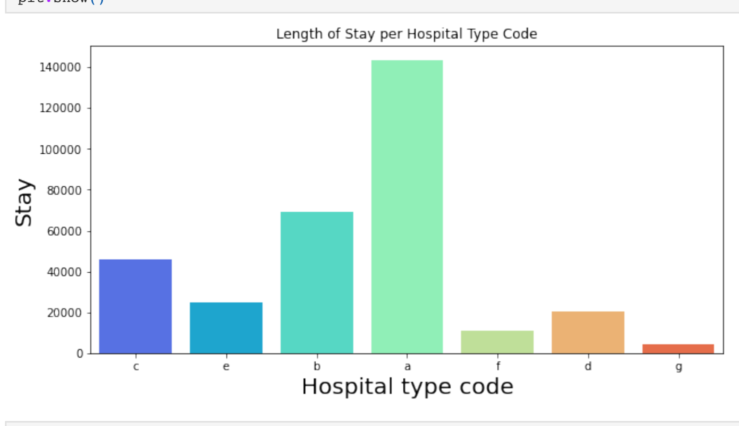
But what is the LOS of most patients in hospital with type code'a'?

what is the LOS of most patients in hospital with type code'g'?

LOS will determine how often the beds become readily available regardless of current

patient density.

```
In [18]:
plt.figure(figsize = (10,5))
sns.countplot(x = 'Hospital_type_code', data = train, palette = 'rainbow')
plt.xlabel('Hospital type code', size = 20)
plt.ylabel('Stay', size = 20)
plt.title('Length of Stay per Hospital Type Code')
plt.show()
```




```
In [19]: train.Stay.value_counts()

Out[19]:
21-30                 87491
11-20                 78139
31-40                 55159
51-60                 35018
0-10                  23604
41-50                 11743
71-80                 10254
More than 100 Days     6683
81-90                  4838
91-100                 2765
61-70                  2744
Name: Stay, dtype: int64
```

```
In [20]:
Stay_index = train.Stay.value_counts().index[:11]
Stay_index

Out[20]: Index(['21-30', '11-20', '31-40', '51-60', '0-10', '41-50', '71-80',
       'More than 100 Days', '81-90', '91-100', '61-70'],
      dtype='object')
```

```
In [21]:
Hospital_type_codeindex= train.Hospital_type_code.value_counts().index[:7]
Hospital_type_codeindex

Out[21]: Index(['a', 'b', 'c', 'e', 'd', 'f', 'g'], dtype='object')
```

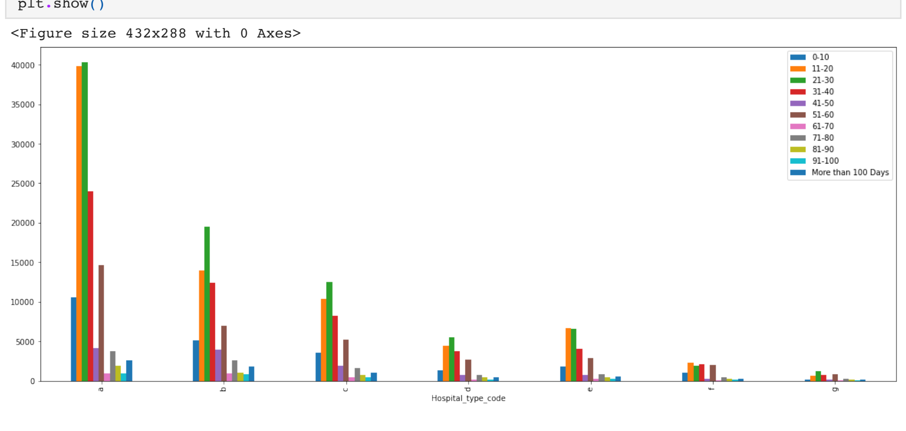
```
In [22]: subdata = train[(train.Hospital_type_code.isin(Hospital_type_codeindex)) & (train.Stay.isin(Stay_ind
```

```
In [23]:
cf = pd.crosstab(columns=subdata.Stay, index = subdata.Hospital_type_code)
cf
```

Out[23]:

| Stay | 0-10 | 11-20 | 21-30 | 31-40 | 41-50 | 51-60 | 61-70 | 71-80 | 81-90 | 91-100 | More than 100 Days |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Hospital_type_code | | | | | | | | | | | |
| a | 10559 | 39807 | 40286 | 23974 | 4111 | 14617 | 924 | 3722 | 1890 | 962 | 2573 |
| b | 5145 | 13933 | 19541 | 12385 | 3893 | 6909 | 917 | 2608 | 1026 | 783 | 1806 |
| c | 3588 | 10318 | 12526 | 8222 | 1872 | 5176 | 430 | 1611 | 725 | 439 | 1021 |
| d | 1314 | 4423 | 5527 | 3706 | 760 | 2713 | 186 | 761 | 421 | 180 | 398 |
| e | 1805 | 6644 | 6524 | 4060 | 764 | 2832 | 197 | 824 | 395 | 218 | 507 |
| f | 1023 | 2332 | 1897 | 2093 | 204 | 1960 | 44 | 482 | 254 | 138 | 276 |
| g | 170 | 682 | 1190 | 719 | 139 | 811 | 46 | 246 | 127 | 45 | 102 |

```
In [24]:
plt.figure()
cf = pd.crosstab(columns=subdata.Stay, index = subdata.Hospital_type_code)
cf.plot.bar(figsize = (20,8), )
plt.legend(loc = 'best')
plt.show()

<Figure size 432x288 with 0 Axes>
```




Most patients in all hopital type codes had a LOS= 21-30 followed by 11-20

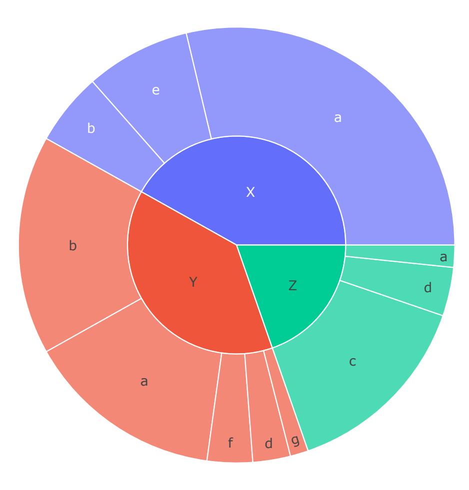
# Data Wrangling

```
In [25]: train.Hospital_region_code.describe()

Out[25]:
count     318438
unique         3
top            X
freq      133336
Name: Hospital_region_code, dtype: object
```

```
In [26]: train.Hospital_region_code.value_counts()

Out[26]:
X    133336
Y    122428
Z     62674
Name: Hospital_region_code, dtype: int64
```

```
In [27]:
import plotly.express as px
fig = px.sunburst(train, path=['Hospital_region_code', 'Hospital_type_code'])
fig.update_layout(title='Categorizing Hospital Type according to Region')
fig.show()
```

Categorizing Hospital Type according to Region




This shows that patient density is very close within X and Y region. in the X region the patients are better distributed between different hospital codes than regions Y and Z.

These indep variables are not valuable for our task. The aim of this project is to predict the LOS. Predicting LOS helps hospitals to identify patients of high LOS risk (patients who will stay longer) at the time of admission and thus better allocate resources for the management of these patients for the purposse of decreasing that long LOS.Thus, how patients are distributed between different hospital type codes and region is not important here at all. What we care about is how to better manage patient at risk of high LOS in each hospital.

For this matter I have decided to drop:case_id,Hospital_code,City_Code_Hospital,Hospital_region_code,City_Code_Patie with patient,Admission_Deposit

I will keep the indep variable Hospital_type_code

```
In [28]: train
```

Out[28]:

| | case_id | Hospital_code | Hospital_type_code | City_Code_Hospital | Hospital_region_code | Available Extra Rooms in Hospital | Department | W |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 8 | c | 3 | Z | 3 | radiotherapy | |
| 1 | 2 | 2 | c | 5 | Z | 2 | radiotherapy | |
| 2 | 3 | 10 | e | 1 | X | 2 | anesthesia | |
| 3 | 4 | 26 | b | 2 | Y | 2 | radiotherapy | |
| 4 | 5 | 26 | b | 2 | Y | 2 | radiotherapy | |
| ... | ... | ... | ... | ... | ... | ... | ... | |
| 318433 | 318434 | 6 | a | 6 | X | 3 | radiotherapy | |
| 318434 | 318435 | 24 | a | 1 | X | 2 | anesthesia | |
| 318435 | 318436 | 7 | a | 4 | X | 3 | gynecology | |
| 318436 | 318437 | 11 | b | 2 | Y | 3 | anesthesia | |
| 318437 | 318438 | 19 | a | 7 | Y | 5 | gynecology | |

318438 rows × 19 columns

```
In [29]: train.Ward_Type.value_counts()

Out[29]:
R    127947
Q    106165
S     77794
P      5046
T      1477
U         9
Name: Ward_Type, dtype: int64
```

```
In [30]: train.Ward_Facility_Code.value_counts()

Out[30]:
F    112753
E     55351
D     51809
C     35463
B     35156
A     27906
Name: Ward_Facility_Code, dtype: int64
```

I will drop Ward_Type and Ward_Facility_Code as well. As they don't play any role in predicting our dep variable LOS.

```
In [31]: train_new=train.drop(['case_id','Hospital_code','City_Code_Hospital','Hospital_region_code','City_Co
```

```
In [32]: train_new
```

Out[32]:

| | Hospital_type_code | Available Extra Rooms in Hospital | Department | Bed Grade | patientid | Type of Admission | Severity of Illness | Age | Stay | count |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | c | 3 | radiotherapy | 2.0 | 31397 | Emergency | Extreme | 51-60 | 0-10 | 1 |
| 1 | c | 2 | radiotherapy | 2.0 | 31397 | Trauma | Extreme | 51-60 | 41-50 | 1 |
| 2 | e | 2 | anesthesia | 2.0 | 31397 | Trauma | Extreme | 51-60 | 31-40 | 1 |
| 3 | b | 2 | radiotherapy | 2.0 | 31397 | Trauma | Extreme | 51-60 | 41-50 | 1 |
| 4 | b | 2 | radiotherapy | 2.0 | 31397 | Trauma | Extreme | 51-60 | 41-50 | 1 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 318433 | a | 3 | radiotherapy | 4.0 | 86499 | Emergency | Moderate | 41-50 | 11-20 | 1 |
| 318434 | a | 2 | anesthesia | 4.0 | 325 | Urgent | Moderate | 81-90 | 31-40 | 1 |
| 318435 | a | 3 | gynecology | 4.0 | 125235 | Emergency | Minor | 71-80 | 11-20 | 1 |
| 318436 | b | 3 | anesthesia | 3.0 | 91081 | Trauma | Minor | 11-20 | 11-20 | 1 |
| 318437 | a | 5 | gynecology | 2.0 | 21641 | Emergency | Minor | 11-20 | 0-10 | 1 |

318438 rows × 10 columns

the patientid in the first 5 observations is the same: 31397. Is this re-admission ? 31397 mentioned twice in hospital type code c and twice in b.

as a healthcare professional I know that the patient is given same MRN regardless of number of encounters. It is alarming because the LOS ASSOCIATED with that patientid is very high 41-50 days. Maybe there are duplicated rows? rows 3 and 4 seem duplicated. Let me drop one of them.

```
In [33]: train_new.drop(train_new.index[4])
```

Out[33]:

| | Hospital_type_code | Available Extra Rooms in Hospital | Department | Bed Grade | patientid | Type of Admission | Severity of Illness | Age | Stay | count |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | c | 3 | radiotherapy | 2.0 | 31397 | Emergency | Extreme | 51-60 | 0-10 | 1 |
| 1 | c | 2 | radiotherapy | 2.0 | 31397 | Trauma | Extreme | 51-60 | 41-50 | 1 |
| 2 | e | 2 | anesthesia | 2.0 | 31397 | Trauma | Extreme | 51-60 | 31-40 | 1 |
| 3 | b | 2 | radiotherapy | 2.0 | 31397 | Trauma | Extreme | 51-60 | 41-50 | 1 |
| 5 | a | 2 | anesthesia | 2.0 | 31397 | Trauma | Extreme | 51-60 | 11-20 | 1 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 318433 | a | 3 | radiotherapy | 4.0 | 86499 | Emergency | Moderate | 41-50 | 11-20 | 1 |
| 318434 | a | 2 | anesthesia | 4.0 | 325 | Urgent | Moderate | 81-90 | 31-40 | 1 |
| 318435 | a | 3 | gynecology | 4.0 | 125235 | Emergency | Minor | 71-80 | 11-20 | 1 |
| 318436 | b | 3 | anesthesia | 3.0 | 91081 | Trauma | Minor | 11-20 | 11-20 | 1 |
| 318437 | a | 5 | gynecology | 2.0 | 21641 | Emergency | Minor | 11-20 | 0-10 | 1 |

318437 rows × 10 columns

The questions we need to answer to predict patients at risk of high LOS:

- What department has the greater density of patients with highest LOS?

- Can bed grade affect LOS? Can it play a role in decreasing or increasing LOS?

- What type of Admission is associated with highest LOS?

- What level of severity of illness is associated with highest LOS?

- What Age group is associated with highest LOS?

We will answer these questions in the following visualizations.

```
In [34]: train_new=train_new.drop(['Hospital_type_code','patientid'],axis=1)
```

```
In [35]: train_new
```

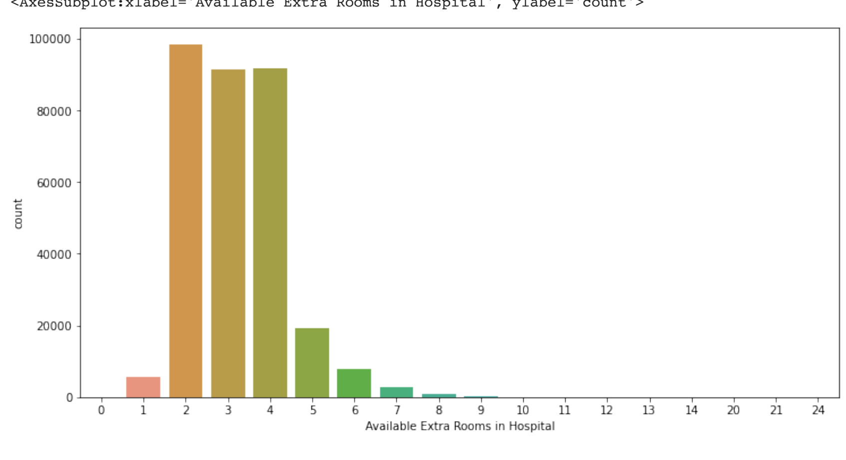
Out[35]:

| | Available Extra Rooms in Hospital | Department | Bed Grade | Type of Admission | Severity of Illness | Age | Stay | count |
|---|---|---|---|---|---|---|---|---|
| 0 | 3 | radiotherapy | 2.0 | Emergency | Extreme | 51-60 | 0-10 | 1 |
| 1 | 2 | radiotherapy | 2.0 | Trauma | Extreme | 51-60 | 41-50 | 1 |
| 2 | 2 | anesthesia | 2.0 | Trauma | Extreme | 51-60 | 31-40 | 1 |
| 3 | 2 | radiotherapy | 2.0 | Trauma | Extreme | 51-60 | 41-50 | 1 |
| 4 | 2 | radiotherapy | 2.0 | Trauma | Extreme | 51-60 | 41-50 | 1 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 318433 | 3 | radiotherapy | 4.0 | Emergency | Moderate | 41-50 | 11-20 | 1 |
| 318434 | 2 | anesthesia | 4.0 | Urgent | Moderate | 81-90 | 31-40 | 1 |
| 318435 | 3 | gynecology | 4.0 | Emergency | Minor | 71-80 | 11-20 | 1 |
| 318436 | 3 | anesthesia | 3.0 | Trauma | Minor | 11-20 | 11-20 | 1 |
| 318437 | 5 | gynecology | 2.0 | Emergency | Minor | 11-20 | 0-10 | 1 |

318438 rows × 8 columns

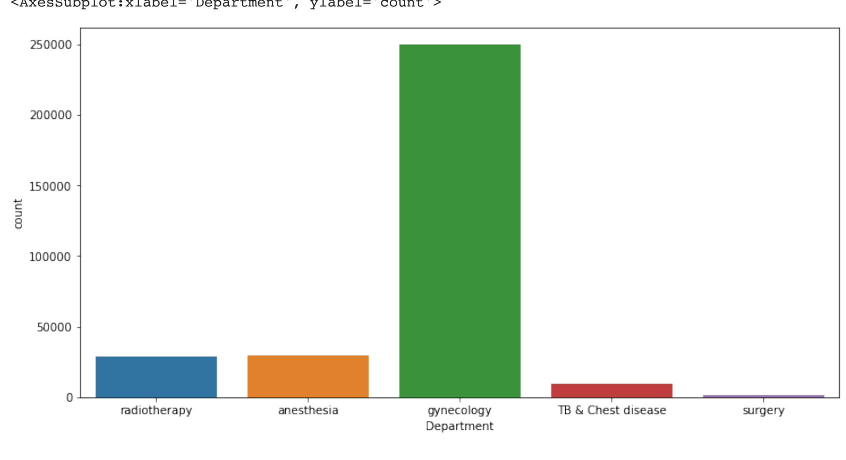
### Available extra rooms in hospital

```
In [36]:
plt.figure(figsize=(12, 6))
sns.countplot(train_new['Available Extra Rooms in Hospital'])

/opt/anaconda3/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning:

Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argumen
t will be `data`, and passing other arguments without an explicit keyword will result in an error or
misinterpretation.

Out[36]: <AxesSubplot:xlabel='Available Extra Rooms in Hospital', ylabel='count'>
```




### Patient denisty per department

```
In [37]:
plt.figure(figsize=(12, 6))
sns.countplot(train.Department)

/opt/anaconda3/lib/python3.8/site-packages/seaborn/_decorators.py:36: FutureWarning:

Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argumen
t will be `data`, and passing other arguments without an explicit keyword will result in an error or
misinterpretation.

Out[37]: <AxesSubplot:xlabel='Department', ylabel='count'>
```




This doesn't tell me much. Highest patient density (patient admissions) is in Gyn department. Typically for the LOS won't go over 2-3 days. Unless there's complications with mom or baby. Many departments are not included in this dataset; especially departments that tend to have patients with high LOS.

Where's the ICU? DOU? MedSurg? Telemetry? NICU?

Since the significant and busiest units of the hospital, where the MAJORITY of patients stay for care, are missing. I conclude that this dataset is incomplete. One can't possibly build a model to predict LOS using a dataset in which the vast majority of its records is from the labor & delivery unit. Any predictive ML model built using this dataset would do poorly and would give a misrepresentation of the predicted LOS.