Self Case Study-1_Pump it up-Data Mining the Water Table

Section-I

1. Business Problem

The problem that we are going to solve here is addressing the issue of scarcity of clean drinking water that is being faced by population of Tanzania owing to non-functioning of some of the water-points. Using the given dataset, we have to predict which pumps are functional, which need some repairs, and which don't work at all. We have to predict one of these three classes based on a number of variables about what kind of pump is operating, when it was installed, what is geographic location and how it is managed etc. A smart understanding of which water-points will fail can improve maintenance operations and ensure that clean, potable water is available to communities across Tanzania

2. Mapping the real-world problem to an ML problem

2.1 Description

This particular business problem can be formulated as ML classification problem where the goal is to predict the operating condition of a water-point for each record in the dataset. The target variable/ classes is named as status_group which has three values such as 'functional', 'non-functional' or 'needs repair'

2.2 Data Source

The data can be downloaded from the link: https://www.drivendata.org/competitions/7/pump-it-up-data-mining-the-water-table/ (https://www.drivendata.org/competitions/7/pump-it-up-data-mining-the-water-table/).

We are provided with 4 no. CSV files as follows,

1. train.csv
2. test.csv
3. train_labels.csv
4. SubmissionFormat.csv

The train dataset has 59400 datapoints and 40 features

3 Exploratory Data Analysis

3.1 Importing dependencies

In [103]:

```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import re
import time
import warnings
warnings.filterwarnings("ignore")
import numpy as np
from nltk.corpus import stopwords
from sklearn.preprocessing import normalize
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.manifold import TSNE
import seaborn as sns
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
import sys
```

3.2 Reading Data

In [104]:

```
pd.options.display.max_columns=60 #for reading all columns
```

In [105]:

```
df_train = pd.read_csv("train.csv")
```

In [106]:

```
df_train.shape
```

Out[106]:

```
(59400, 40)
```

In [107]:
```
df_train.head() # loading train dataset
```

Out[107]:

	id	amount_tsh	date_recorded	funder	gps_height	installer	longitude	latitude	wpt_name	num_private	basin	subvillage	region	region_c
0	69572	6000.0	2011-03-14	Roman	1390	Roman	34.938093	-9.856322	none	0	Lake Nyasa	Mnyusi B	Iringa	
1	8776	0.0	2013-03-06	Grumeti	1399	GRUMETI	34.698766	-2.147466	Zahanati	0	Lake Victoria	Nyamara	Mara	
2	34310	25.0	2013-02-25	Lottery Club	686	World vision	37.460664	-3.821329	Kwa Mahundi	0	Pangani	Majengo	Manyara	
3	67743	0.0	2013-01-28	Unicef	263	UNICEF	38.486161	-11.155298	Zahanati Ya Nanyumbu	0	Ruvuma / Southern Coast	Mahakamani	Mtwara	
4	19728	0.0	2011-07-13	Action In A	0	Artisan	31.130847	-1.825359	Shuleni	0	Lake Victoria	Kyanyamisa	Kagera	

In [108]:
```
df_train_labels = pd.read_csv("train_lables.csv") #loading train labels data
```

In [109]:
```
df_train_labels.head()
```

Out[109]:

	id	status_group
0	69572	functional
1	8776	functional
2	34310	functional
3	67743	non functional
4	19728	functional

3.3 Merging df_train & df_train_labels

In [110]:
```
df = df_train.merge(df_train_labels, how='left', on='id')
```

3.4 Reading mearged data

In [111]:
```
print('Number of data points : ', df.shape[0])
print('Number of features : ', df.shape[1])
print('Features : ', df.columns.values)
df.head()
```

```
Number of data points :  59400
Number of features :  41
Features :  ['id' 'amount_tsh' 'date_recorded' 'funder' 'gps_height' 'installer'
 'longitude' 'latitude' 'wpt_name' 'num_private' 'basin' 'subvillage'
 'region' 'region_code' 'district_code' 'lga' 'ward' 'population'
 'public_meeting' 'recorded_by' 'scheme_management' 'scheme_name' 'permit'
 'construction_year' 'extraction_type' 'extraction_type_group'
 'extraction_type_class' 'management' 'management_group' 'payment'
 'payment_type' 'water_quality' 'quality_group' 'quantity'
 'quantity_group' 'source' 'source_type' 'source_class' 'waterpoint_type'
 'waterpoint_type_group' 'status_group']
```

Out[111]:

	id	amount_tsh	date_recorded	funder	gps_height	installer	longitude	latitude	wpt_name	num_private	basin	subvillage	region	region_c
0	69572	6000.0	2011-03-14	Roman	1390	Roman	34.938093	-9.856322	none	0	Lake Nyasa	Mnyusi B	Iringa	
1	8776	0.0	2013-03-06	Grumeti	1399	GRUMETI	34.698766	-2.147466	Zahanati	0	Lake Victoria	Nyamara	Mara	
2	34310	25.0	2013-02-25	Lottery Club	686	World vision	37.460664	-3.821329	Kwa Mahundi	0	Pangani	Majengo	Manyara	
3	67743	0.0	2013-01-28	Unicef	263	UNICEF	38.486161	-11.155298	Zahanati Ya Nanyumbu	0	Ruvuma / Southern Coast	Mahakamani	Mtwara	
4	19728	0.0	2011-07-13	Action In A	0	Artisan	31.130847	-1.825359	Shuleni	0	Lake Victoria	Kyanyamisa	Kagera	

3.5 Understanding the columns of dataset

In [112]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 59400 entries, 0 to 59399
Data columns (total 41 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   id                     59400 non-null  int64  
 1   amount_tsh             59400 non-null  float64
 2   date_recorded          59400 non-null  object 
 3   funder                 55765 non-null  object 
 4   gps_height             59400 non-null  int64  
 5   installer              55745 non-null  object 
 6   longitude              59400 non-null  float64
 7   latitude               59400 non-null  float64
 8   wpt_name               59400 non-null  object 
 9   num_private            59400 non-null  int64  
 10  basin                  59400 non-null  object 
 11  subvillage             59029 non-null  object 
 12  region                 59400 non-null  object 
 13  region_code            59400 non-null  int64  
 14  district_code          59400 non-null  int64  
 15  lga                    59400 non-null  object 
 16  ward                   59400 non-null  object 
 17  population             59400 non-null  int64  
 18  public_meeting         56066 non-null  object 
 19  recorded_by            59400 non-null  object 
 20  scheme_management      55523 non-null  object 
 21  scheme_name            31234 non-null  object 
 22  permit                 56344 non-null  object 
 23  construction_year      59400 non-null  int64  
 24  extraction_type        59400 non-null  object 
 25  extraction_type_group  59400 non-null  object 
 26  extraction_type_class  59400 non-null  object 
 27  management             59400 non-null  object 
 28  management_group       59400 non-null  object 
 29  payment                59400 non-null  object 
 30  payment_type           59400 non-null  object 
 31  water_quality          59400 non-null  object 
 32  quality_group          59400 non-null  object 
 33  quantity               59400 non-null  object 
 34  quantity_group         59400 non-null  object 
 35  source                 59400 non-null  object 
 36  source_type            59400 non-null  object 
 37  source_class           59400 non-null  object 
 38  waterpoint_type        59400 non-null  object 
 39  waterpoint_type_group  59400 non-null  object 
```

```
 40  status_group           59400 non-null  object
dtypes: float64(3), int64(7), object(31)
memory usage: 19.0+ MB
```

In [113]:
```
df.isnull().sum()
```

Out[113]:
```
id                           0
amount_tsh                   0
date_recorded                0
funder                    3635
gps_height                   0
installer                 3655
longitude                    0
latitude                     0
wpt_name                     0
num_private                  0
basin                        0
subvillage                 371
region                       0
region_code                  0
district_code                0
lga                          0
ward                         0
population                   0
public_meeting            3334
recorded_by                  0
scheme_management         3877
scheme_name              28166
permit                    3056
construction_year            0
extraction_type              0
extraction_type_group        0
extraction_type_class        0
management                   0
management_group             0
payment                      0
payment_type                 0
water_quality                0
quality_group                0
quantity                     0
quantity_group               0
source                       0
source_type                  0
source_class                 0
waterpoint_type              0
waterpoint_type_group        0
status_group                 0
dtype: int64
```

3. 5.1 Data visualization using library 'pandas_profiling'

In [114]:

```
import pandas_profiling as pp
```

In [115]:
```
pp.ProfileReport(df)
```

Summarize dataset: 100% 156/156 [01:41<00:00, 1.35it/s, Completed]

Generate report structure: 100% 1/1 [00:25<00:00, 25.85s/it]

Render HTML: 100% 1/1 [00:12<00:00, 12.08s/it]

Dataset statistics

Number of variables	41
Number of observations	59400
Missing cells	46094
Missing cells (%)	1.9%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	19.0 MiB
Average record size in memory	336.0 B

Variable types

Numeric	10
Categorical	29
Boolean	2

Alerts

`recorded_by` has constant value "GeoData Consultants Ltd"	**Constant**
`date_recorded` has a high cardinality: 356 distinct values	**High cardinality**
`funder` has a high cardinality: 1897 distinct values	**High cardinality**
`installer` has a high cardinality: 2145 distinct values	**High cardinality**
`wpt_name` has a high cardinality: 37400 distinct values	**High cardinality**
`subvillage` has a high cardinality: 19287 distinct values	**High cardinality**
`lga` has a high cardinality: 125 distinct values	**High cardinality**
`ward` has a high cardinality: 2092 distinct values	**High cardinality**
`scheme_name` has a high cardinality: 2696 distinct values	**High cardinality**
`gps_height` is highly correlated with `population` and 1 other fields (population, construction_year)	**High correlation**
`population` is highly correlated with `gps_height` and 1 other fields (gps_height, construction_year)	**High correlation**
`construction_year` is highly correlated with `gps_height` and 1 other fields (gps_height, population)	**High correlation**

Out[115]:

3.5.2 Exploring the columns individually one by one for more clarity

3.5.2.1 Column 'amount_tsh'

In [116]:
```
df['amount_tsh'].value_counts()
```

Out[116]:
```
0.00         41639
500.00        3102
50.00         2472
1000.00       1488
20.00         1463
200.00        1220
100.00         816
10.00          806
30.00          743
2000.00        704
250.00         569
300.00         557
5000.00        450
5.00           376
25.00          356
3000.00        334
1200.00        267
1500.00        197
6.00           190
```

In [117]:
```
df.loc[df['amount_tsh']==0].groupby('status_group').size()
```

Out[117]:
```
status_group
functional                 19706
functional needs repair     3048
non functional             18885
dtype: int64
```

In [119]:
```python
df.groupby(['amount_tsh','status_group']).size().head(20)
```

Out[119]:
```
amount_tsh  status_group
0.00        functional                 19706
            functional needs repair     3048
            non functional             18885
0.20        non functional                 3
0.25        functional                     1
1.00        non functional                 3
2.00        functional                    13
5.00        functional                   330
            non functional                46
6.00        functional                   174
            functional needs repair        3
            non functional                13
7.00        functional                    54
            non functional                15
9.00        non functional                 1
10.00       functional                   623
            functional needs repair       16
            non functional               167
```

From the above analysis of 'amount_tsh' it is observed that the column has total 41639 zero values which is about 70% of total datapoints. When the static head is zero, the suction and discharge points are at same level. this is in favour of the pump. we have total 19706 functinal water points when static head is zero. However there are 18885 non functional water points at zero head.

3.5.2.2 Column 'date_recorded'

In [120]:
```python
df['date_recorded'].isna().sum()
```

Out[120]:
```
0
```

In [121]:
```python
df.date_recorded.nunique()
```

Out[121]:
```
356
```

This column has zero null values and total 356 unique values. For now let us keep this column as is.

3.5.2.3 Column 'funder'

In [122]:
```python
df.funder.nunique()
```
Out[122]:
```
1897
```
In [123]:
```python
df['funder'].isna().sum()
```
Out[123]:
```
3635
```
this column has 1897 unique values and 3635 missing values

In [124]:
```python
pd.set_option('display.max_rows', None)
df['funder'].value_counts().head(150).sum()
```
Out[124]:
```
47206
```
In [125]:
```python
df['funder'].fillna(value='Undefined',inplace=True)
df['funder'].replace(to_replace = '0', value ='Undefined' , inplace=True) #replacing '0' & missing values with 'Undefined'
```
In [127]:
```python
df['funder'].value_counts().head(30)
```
Out[127]:
```
Government Of Tanzania    9084
Undefined                 4412
Danida                    3114
Hesawa                    2202
Rwssp                     1374
World Bank                1349
Kkkt                      1287
World Vision              1246
Unicef                    1057
Tasaf                      877
District Council           843
Dhv                        829
Private Individual         826
Dwsp                       811
Norad                      765
Germany Republi            610
Tcrs                       602
Ministry Of Water          590
Water                      583
```
In [129]:
```python
top_30_funders = ['Government Of Tanzania','Undefined','Danida','Hesawa','Rwssp','World Bank','Kkkt','World Vision','Unicef','Tas
```

In [132]:
```
df.loc[~df["funder"].isin(top_30_funders), "funder"] = 'other'
```

In [134]:
```
df['funder'].unique()
```

Out[134]:
```
array(['other', 'Unicef', 'Dwsp', 'Rwssp', 'Wateraid', 'Danida',
       'World Vision', 'Hesawa', 'Isf', 'Government Of Tanzania', 'Water',
       'Private Individual', 'Undefined', 'Lga', 'District Council',
       'Kkkt', 'Norad', 'Dwe', 'Rc Church', 'Tcrs', 'Germany Republi',
       'Netherlands', 'Tasaf', 'World Bank', 'Fini Water', 'Dhv', 'Amref',
       'Ministry Of Water', 'Adb', 'Oxfam', 'Hifab'], dtype=object)
```

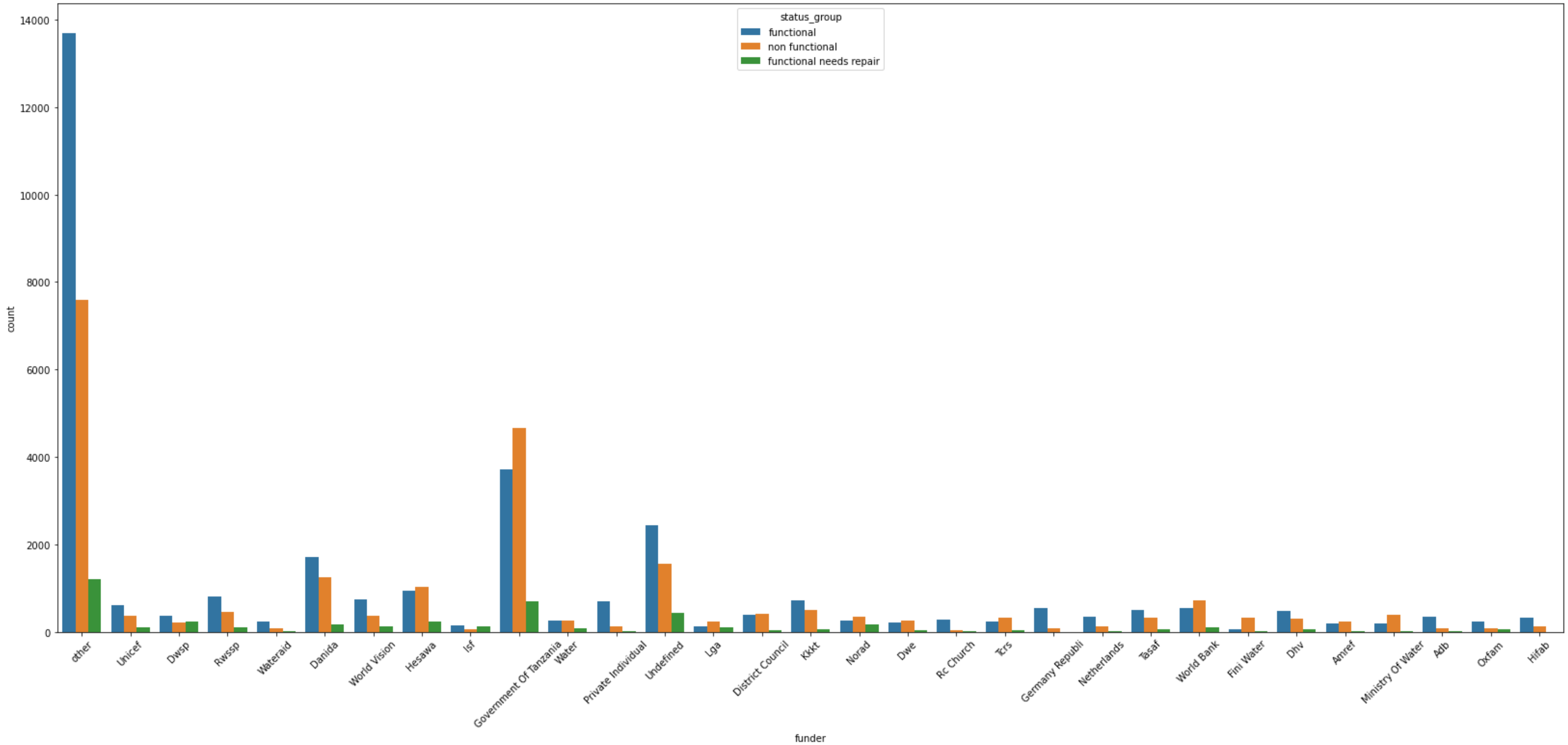
In [135]:
```
df['funder'].isna().sum()
```

Out[135]:
```
0
```

In [136]:
```
df['funder'].value_counts().head(20)
```

Out[136]:
```
other                     22498
Government Of Tanzania     9084
Undefined                  4412
Danida                     3114
Hesawa                     2202
Rwssp                      1374
World Bank                 1349
Kkkt                       1287
World Vision               1246
Unicef                     1057
Tasaf                       877
District Council            843
Dhv                         829
Private Individual          826
Dwsp                        811
Norad                       765
Germany Republi             610
Tcrs                        602
Ministry Of Water           590
Water                       583
Name: funder, dtype: int64
```

In [137]:
```
plt.figure(figsize=(28,12))
ax = sns.countplot(x='funder', hue="status_group", data=df)
ax.tick_params(axis='x', rotation=45)
```



As can be seen from the above plot the most of the waterpoints funded by government of tanzania are non-functional

3.5.2.4 Column 'gps_height'

In [138]:
```
df['gps_height'].value_counts().head(20)
```

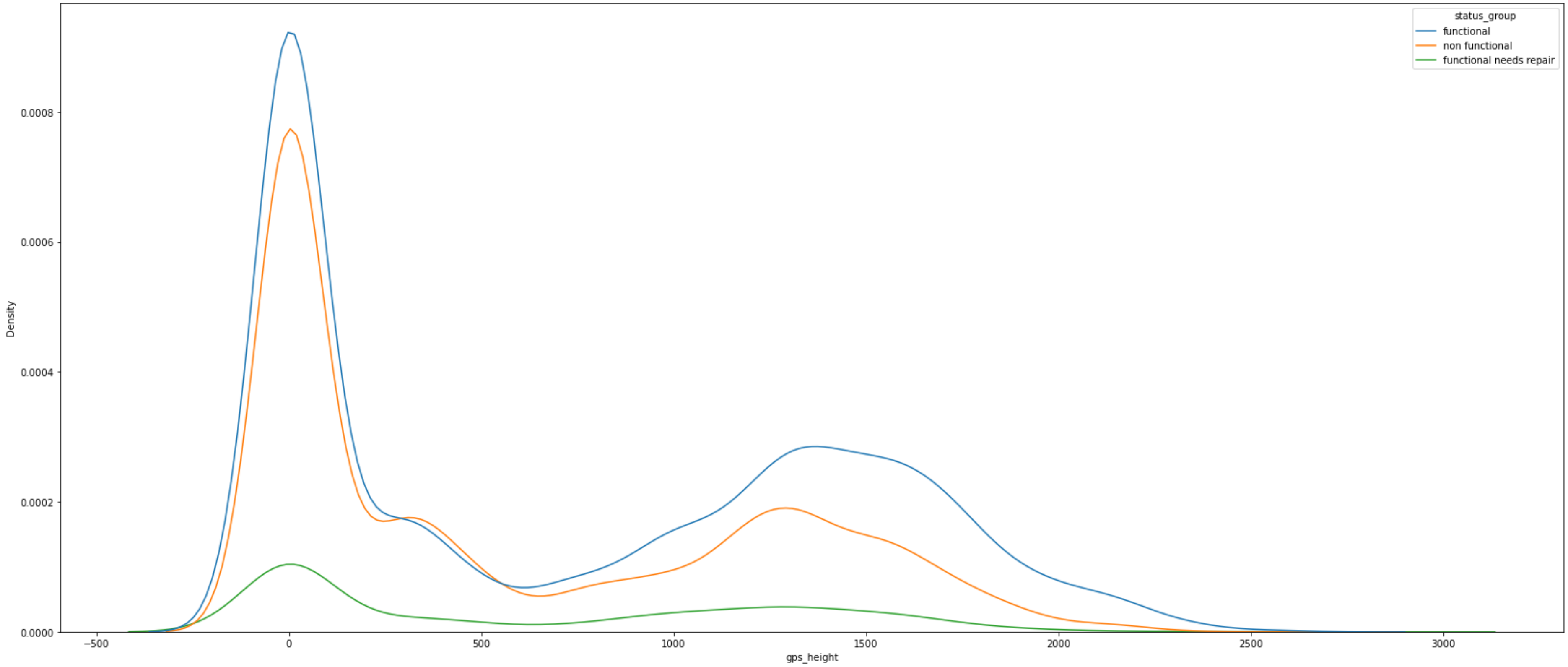
Out[138]:
```
 0       20438
-15         60
-16         55
-13         55
 1290       52
-20         52
-14         51
 303        51
-18         49
-19         47
 1295       46
 1269       46
 1304       45
-23         45
 280        44
 1538       44
-8          44
 1286       44
-17         44
 320        43
Name: gps_height, dtype: int64
```

In [139]:
```
df['gps_height'].isna().sum()
```

Out[139]:
```
0
```

In [140]:
```
plt.figure(figsize=(28,12))
sns.kdeplot(data=df, x='gps_height', hue="status_group", gridsize=200)
```

Out[140]: `<AxesSubplot:xlabel='gps_height', ylabel='Density'>`



3.5.2.5 Column 'installer'

In [141]:
```
df['installer'].value_counts().head(100).sum()
```
Out[141]:
```
43663
```

In [142]:
```
df['installer'].nunique()
```
Out[142]:
```
2145
```

In [143]:
```
df['installer'].isna().sum()
```
Out[143]:
```
3655
```

In [144]:
```
df['installer'].fillna(value='Undefined',inplace=True)
df['installer'].replace(to_replace = '0', value ='Undefined' , inplace=True) #replacing '0' category with 'Undefined'
```

In [145]:
```
pd.set_option('display.max_rows', None)
df['installer'].value_counts().head(20)
```
Out[145]:
```
DWE                    17402
Undefined               4432
Government              1825
RWE                     1206
Commu                   1060
DANIDA                  1050
KKKT                     898
Hesawa                   840
TCRS                     707
Central government       622
CES                      610
Community                553
DANID                    552
District Council         551
HESAWA                   539
LGA                      408
World vision             408
WEDECO                   397
TASAF                    396
```

In [146]:
```
df['installer'].replace(to_replace = ("Gove","Gover","GOVERM", "GOVERN", "GOVERNME", "Governmen","Government","GOVER"), value ="G
df['installer'].replace(to_replace = ("RW","RWE","RWE /Community","RWE Community","RWE/ Community","RWE/Community","RWE/DWE","RWE
df['installer'].replace(to_replace = ("Commu", "Communit", "Community", "COMMUNITY BANK","Comunity"), value ="Community", inplace
df['installer'].replace(to_replace = ("Danda","DANIAD","Danid","DANIDA","DANIDA CO","DANIDS","DANNIDA","DANID"), value ="DANIDA",
df['installer'].replace(to_replace = ("Cebtral Government","Cental Government","Centr","Centra Government","Centra govt","Central
df['installer'].replace(to_replace = ("COUN","Counc","Council","Distri","District  Council","District Community j","District Coun
df['installer'].replace(to_replace = ("Hesawa","HESAW","Hesewa","HESAWA"),value ='HESAWA' , inplace=True)
df['installer'].replace(to_replace = ("World Division","World Visiin","World vision","World Vission","World Vision"),value ='Worl
df['installer'].replace(to_replace = ("Distric Water Department","District Water Department","District water depar","District wat
df['installer'].replace(to_replace = ("FINN WATER","FinW","FinWate","FinWater","Fini water","Fini Water" ),value ='Fini Water' ,
df['installer'].replace(to_replace = ("RC","RC .Church","RC C","RC Ch","RC Churc","RC Church","RC CHURCH BROTHER","RC church/CEFA
df['installer'].replace(to_replace = ("Villa","VILLAGER","Villagerd","Villagers","Villages","Villege Council","Villi","villigers"
```

In [147]:
```
df['installer'].value_counts().head(30)
```

Out[147]:
```
DWE                   17402
Undefined              4432
Government             2592
DANIDA                 1679
Community              1670
HESAWA                 1381
RWE                    1306
District Council       1162
Central Government     1080
KKKT                    898
TCRS                    707
World Vision            693
CES                     610
Fini Water              553
RC Church               490
LGA                     408
WEDECO                  397
TASAF                   396
AMREF                   329
```

In [148]:
```
top_30_installer = ["DWE","Undefined","Government","DANIDA","Community","HESAWA","RWE","District Council","Central Government","K
```

In [149]:
```
df.loc[~df["installer"].isin(top_30_installer), "installer"] = 'other'
```

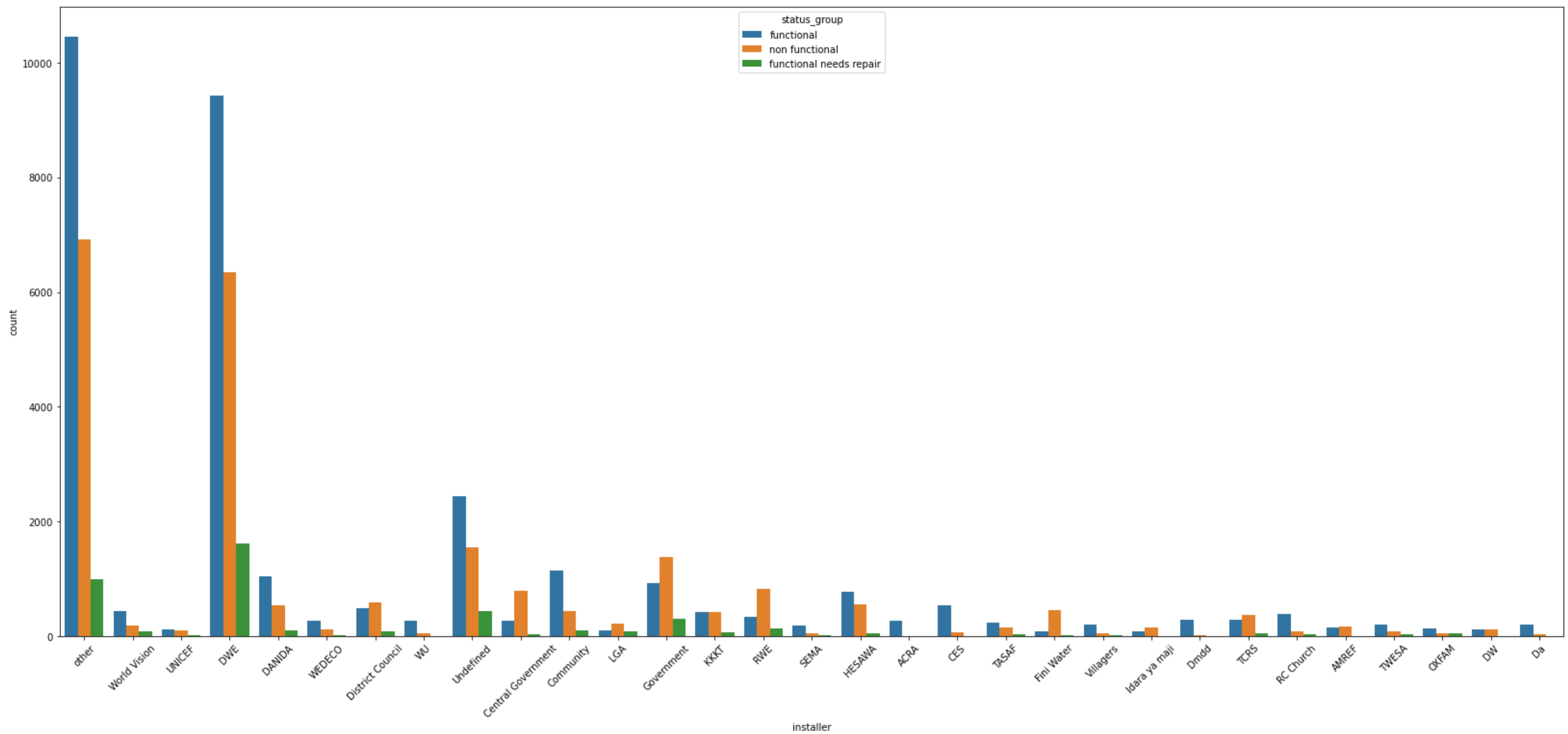
In [150]:
```
df['installer'].nunique()
```

Out[150]:
```
31
```

In [151]:
```
df['installer'].isna().sum()
```

Out[151]: 0

In [152]:
```
plt.figure(figsize=(28,12))
ax = sns.countplot(x='installer', hue="status_group", data=df)
ax.tick_params(axis='x', rotation=45)
```



From the above plot it can be seen that most of the woterpoints install by goverment are non-functional.

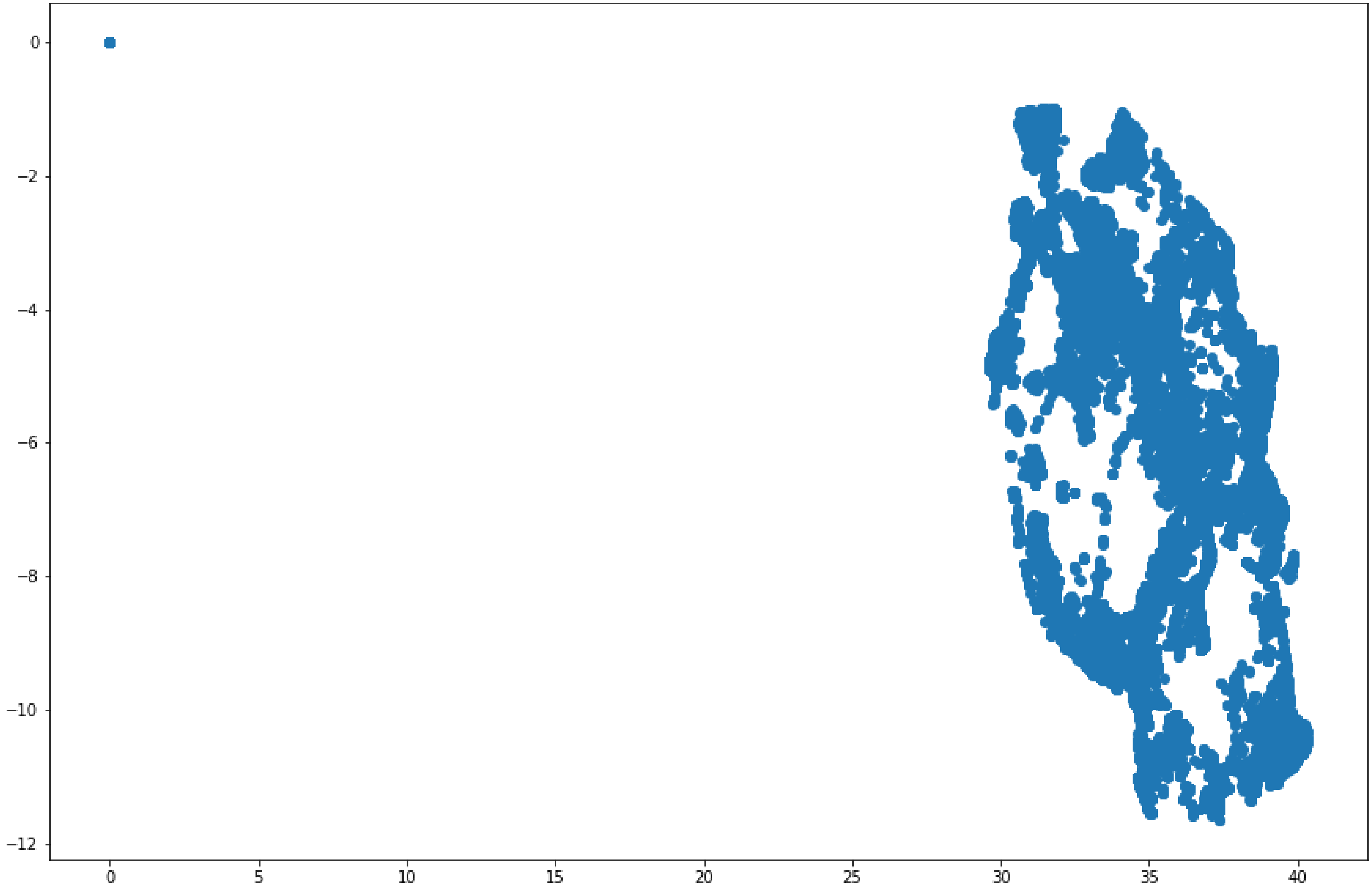
3.5.2.6 Column 'longitude & latitude'

In [153]:
```
df['longitude'].isna().sum()
```

Out[153]: 0

In [154]:
```
df['latitude'].isna().sum()
```
Out[154]:
```
0
```
In [155]:
```
df['longitude'].nunique()
```
Out[155]:
```
57516
```
In [156]:
```
df['latitude'].nunique()
```
Out[156]:
```
57517
```

In [157]:
```
plt.figure(figsize=(15,10))
plt.scatter(x="longitude", y="latitude", data=df)
plt.show()
```



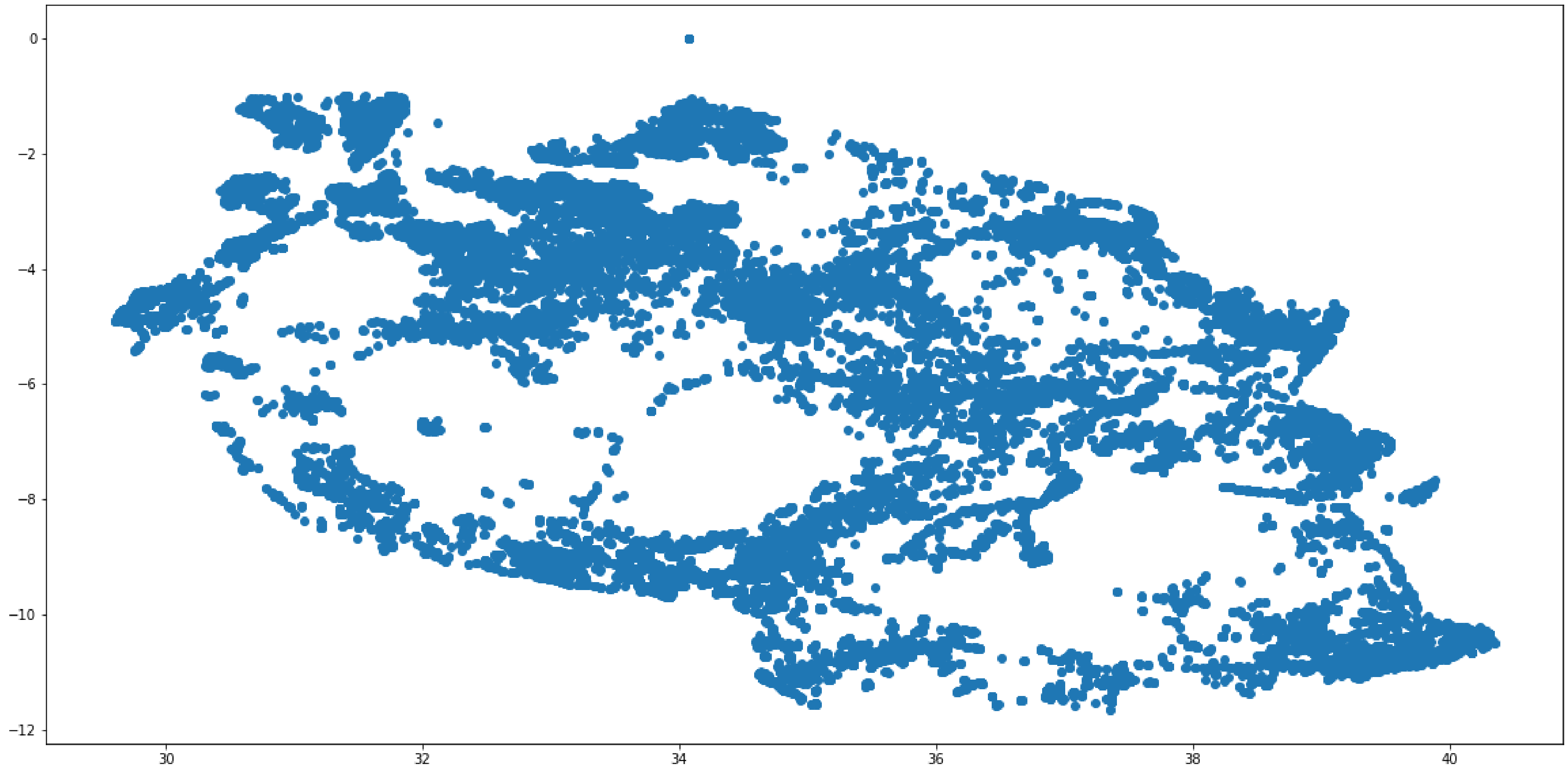
From the above scatterplot it is seen that there is outlier at 0. we will replace the same with 'mean'

In [158]:
```
df['longitude'].mean()
```

Out[158]:
```
34.07742669202832
```

In [159]:
```
df['longitude'].replace(to_replace = 0 , value =34.07742669202832 , inplace=True)
```

In [160]:
```
plt.figure(figsize=(20,10))
plt.scatter(x="longitude", y="latitude", data=df)
plt.show()
```



3.5.2.7 Column 'wpt_name'

In [161]:
```
df['wpt_name'].isna().sum()
```

Out[161]:
```
0
```

In [162]:
```
df['wpt_name'].nunique()
```

Out[162]:
```
37400
```

In [164]:
```
df['wpt_name'].value_counts().head(20)
```

Out[164]:
```
none                3563
Shuleni             1748
Zahanati             830
Msikitini            535
Kanisani             323
Bombani              271
Sokoni               260
Ofisini              254
School               208
Shule Ya Msingi      199
Shule                152
Sekondari            146
Muungano             133
Mkombozi             111
Madukani             104
Mbugani               94
Hospital              94
Upendo                93
Kituo Cha Afya        90
Mkuyuni               88
Name: wpt_name, dtype: int64
```

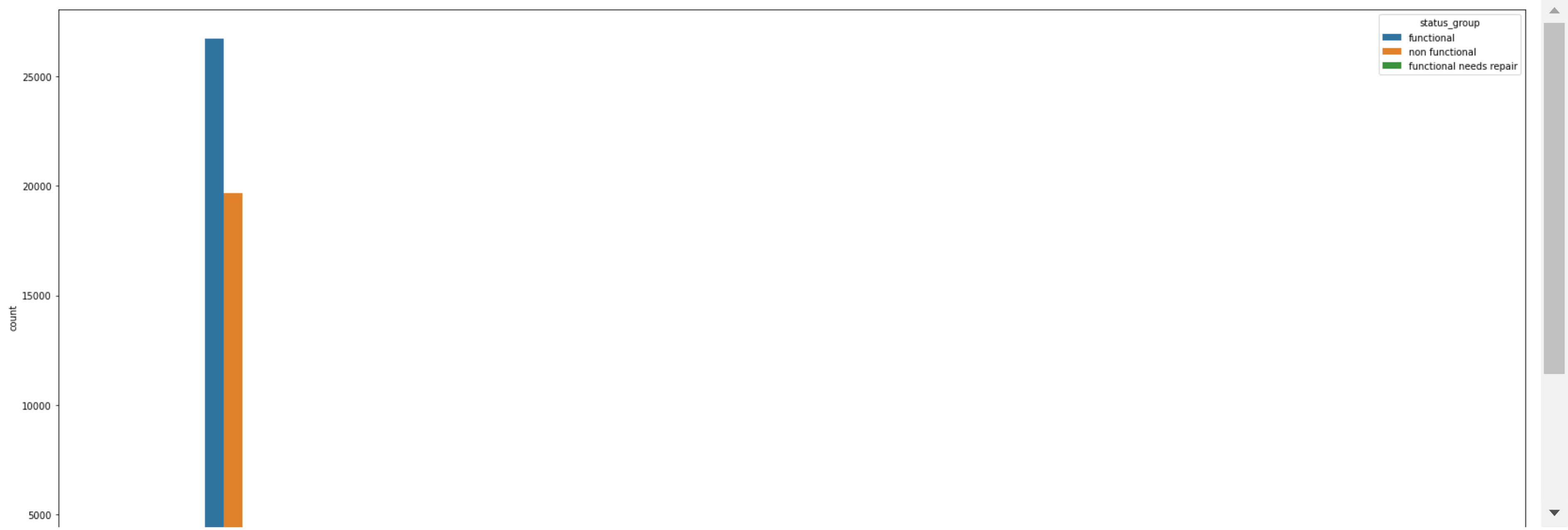
In [165]:
```
top_20_wpt = ["none","Shuleni","Zahanati","Msikitini","Kanisani","Bombani","Sokoni","Ofisini","School","Shule Ya Msingi","Shule",
```

In [170]:
```
w = list(df.loc[~df["wpt_name"].isin(top_20_wpt), "wpt_name"])
df['wpt_name'].replace(to_replace = w , value ='other' , inplace=True)
```

In [171]:
```
df["wpt_name"].nunique()
```

Out[171]:
```
21
```

In [172]:
```python
plt.figure(figsize=(28,12))
ax = sns.countplot(x='wpt_name', hue="status_group", data=df)
ax.tick_params(axis='x', rotation=45)
```



"wpt_name" has 37400 unique values so it does not make sense to retain this feature. the same need be droped.

In [173]:
```python
df.drop(columns='wpt_name', inplace=True)
```

3.5.2.8 Column 'num_private'

In [174]:
```python
df['num_private'].nunique()
```

Out[174]: 65

In [175]:
```
df['num_private'].value_counts()
```

Out[175]:
```
0      58643
6         81
1         73
5         46
8         46
32        40
45        36
15        35
39        30
93        28
3         27
7         26
2         23
65        22
47        21
4         20
102       20
17        17
80        15
```

most of the values in "num_private" are zero. hence we will remove this features

In [176]:
```
df.drop(columns='num_private',inplace=True )
```

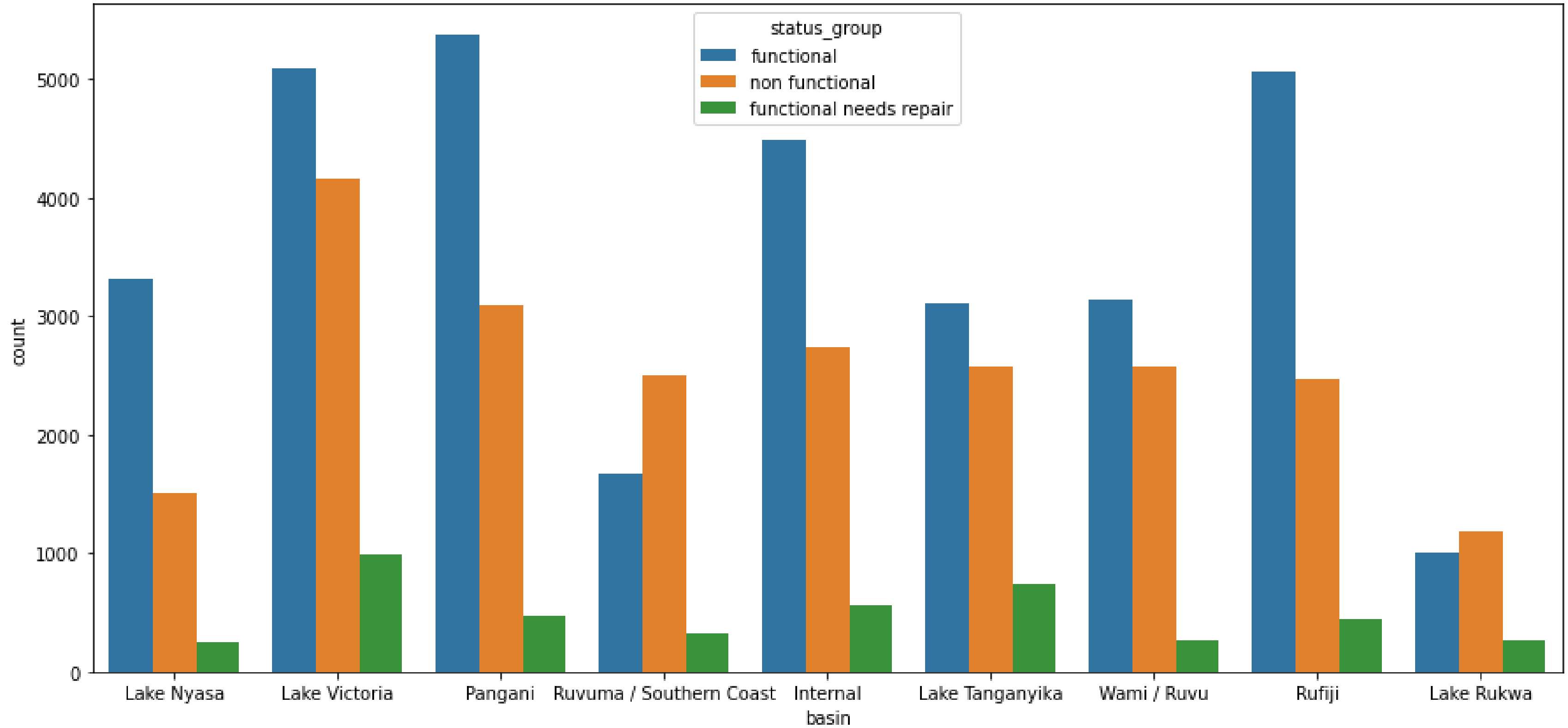
3.5.2.9 Column 'basin'

In [177]:
```
df['basin'].value_counts()
```

Out[177]:
```
Lake Victoria              10248
Pangani                     8940
Rufiji                      7976
Internal                    7785
Lake Tanganyika             6432
Wami / Ruvu                 5987
Lake Nyasa                  5085
Ruvuma / Southern Coast     4493
Lake Rukwa                  2454
Name: basin, dtype: int64
```

In [178]:
```
df['basin'].isna().sum()
```
Out[178]: 0

In [179]:
```
plt.figure(figsize=(15,7))
ax = sns.countplot(x='basin', hue="status_group", data=df)
```



This feature seems have good corelation with class variable 'status_group'

3.5.2.10 Column 'subvillage', 'region'

In [180]:
```
df['subvillage'].isna().sum()
```
Out[180]: 371

In [181]:
```
df['subvillage'].nunique()
```
Out[181]:
```
19287
```
In [182]:
```
df['subvillage'].value_counts().head(30)
```
Out[182]:
```
Madukani      508
Shuleni       506
Majengo       502
Kati          373
Mtakuja       262
Sokoni        232
M             187
Muungano      172
Mbuyuni       164
Mlimani       152
Songambele    147
Msikitini     134
Miembeni      134
1             132
Kibaoni       114
Kanisani      111
I             109
Mapinduzi     109
Mjimwema      108
Mjini         108
Mkwajuni      104
Mwenge        102
Mabatini       98
Azimio         98
Mission        95
Mbugani        95
Bwawani        91
Bondeni        90
Chang'Ombe     88
Zahanati       86
Name: subvillage, dtype: int64
```
In [183]:
```
df['region'].isna().sum()
```
Out[183]:
```
0
```
In [184]:
```
df['region'].nunique()
```
Out[184]:
```
21
```

In [185]:
```
df['region'].value_counts()
```

Out[185]:
```
Iringa           5294
Shinyanga        4982
Mbeya            4639
Kilimanjaro      4379
Morogoro         4006
Arusha           3350
Kagera           3316
Mwanza           3102
Kigoma           2816
Ruvuma           2640
Pwani            2635
Tanga            2547
Dodoma           2201
Singida          2093
Mara             1969
Tabora           1959
Rukwa            1808
Mtwara           1730
Manyara          1583
Lindi            1546
Dar es Salaam     805
Name: region, dtype: int64
```

In [189]:
```
df.groupby(['region','subvillage']).size().head(20)
```

Out[189]:
```
region  subvillage
Arusha  Afya            15
        Ahara            1
        Alairataat       3
        Alakirikir       4
        Alasai           1
        Aleilelai        4
        Alsini           2
        Ambara           1
        Ambureni         8
        Arahati          4
        Arashi           3
        Arati            5
        Arauyo          21
        Ariahati         1
        Arkaria          5
        Arudeko         11
        Ascarida         1
        Athin Kati       1
        Athni Mwisho     1
        Atsin            1
dtype: int64
```

'subvillage' and 'region' both provide information about location of wells. as 'subvillage' and more number of categorical values we will drop column 'subvillage'

In [190]:
```
df.drop(columns='subvillage',inplace=True )
```

In [191]:
```
df.shape
```

Out[191]:
```
(59400, 38)
```

3.5.2.12 Column 'region_code', 'district_code'

In [192]:
```
df['region_code'].nunique()
```

Out[192]:
```
27
```

In [193]:
```
df['district_code'].nunique()
```
Out[193]:
```
20
```
In [194]:
```
df['region_code'].value_counts()
```
Out[194]:
```
11    5300
17    5011
12    4639
3     4379
5     4040
18    3324
19    3047
2     3024
16    2816
10    2640
4     2513
1     2201
13    2093
14    1979
20    1969
15    1808
6     1609
21    1583
80    1238
60    1025
90     917
7      805
99     423
9      390
24     326
8      300
40       1
Name: region_code, dtype: int64
```

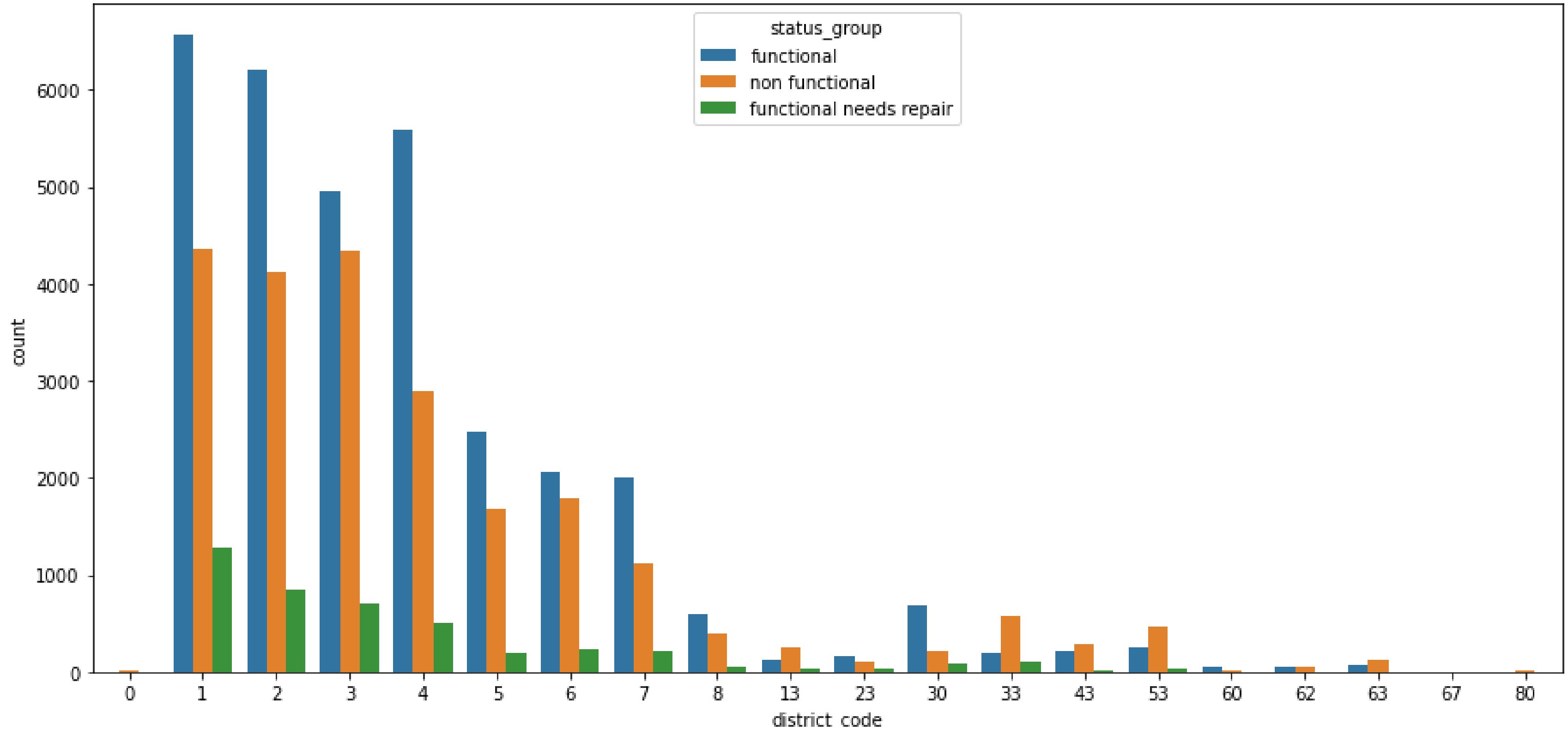
In [195]:
```
df['district_code'].value_counts()
```

Out[195]:
```
1     12203
2     11173
3      9998
4      8999
5      4356
6      4074
7      3343
8      1043
30      995
33      874
53      745
43      505
13      391
23      293
63      195
62      109
60       63
0        23
80       12
67        6
Name: district_code, dtype: int64
```

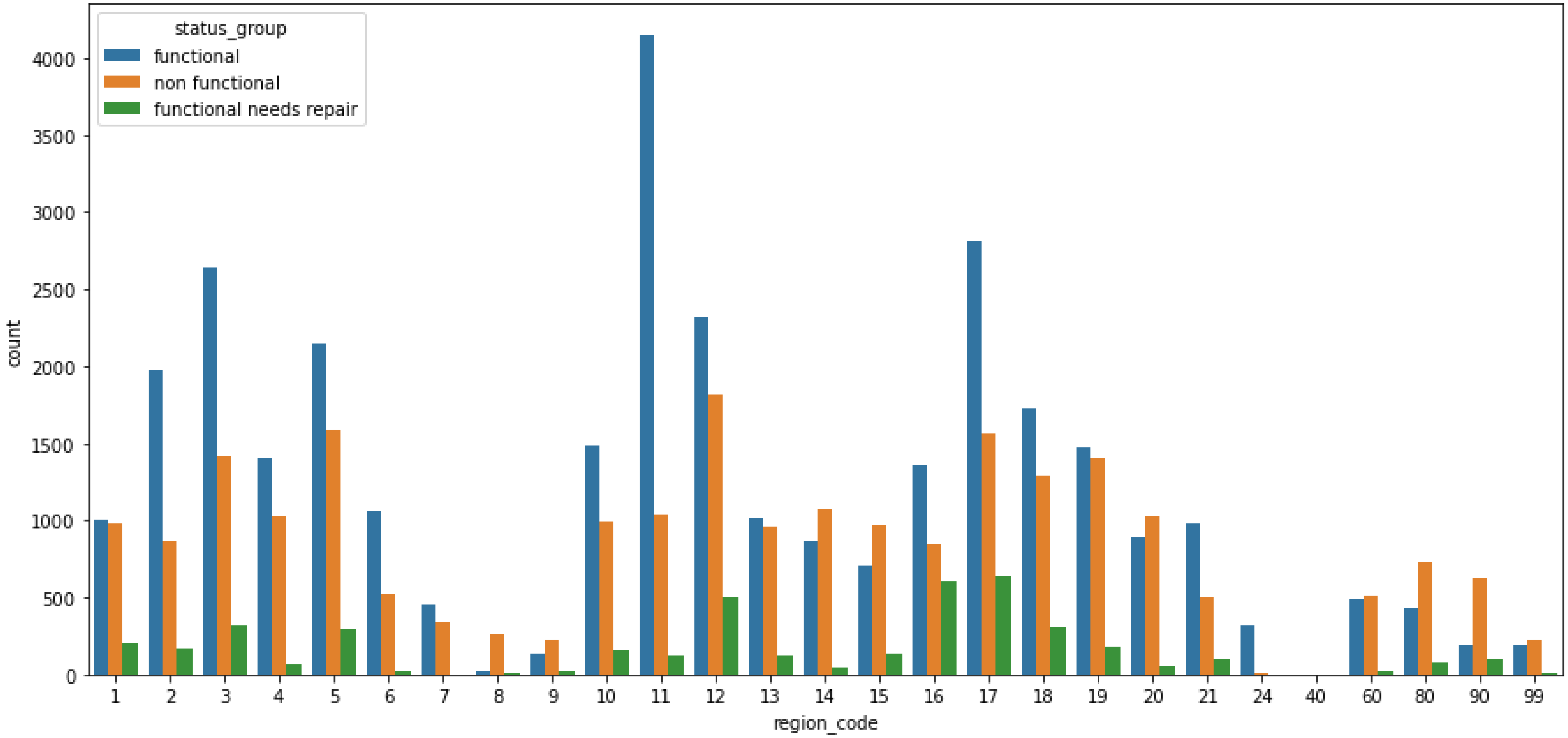
In [196]:
```
df.groupby(['region_code','district_code']).size()
```

Out[196]:
```
region_code  district_code
1            0                  23
             1                 888
             3                 361
             4                 347
             5                 358
             6                 224
2            1                 189
             2                1206
             3                 109
             5                 201
             6                 310
             7                1009
3            1                 595
             2                 519
             3                 877
             4                1225
             5                 620
             6                 109
```

In [197]:
```
plt.figure(figsize=(15,7))
ax = sns.countplot(x='district_code', hue="status_group", data=df)
```



In [198]:
```
plt.figure(figsize=(15,7))
ax = sns.countplot(x='region_code', hue="status_group", data=df)
```



For now we will keep the feature "district_code" and remove "region_code"

In [199]:
```
df.drop(columns='region_code',inplace=True )
```

In [200]:
```
df.shape
```

Out[200]:
```
(59400, 37)
```

3.5.2.13 Column 'lga', 'ward'

In [201]:
```
df['lga'].isna().sum()
```

Out[201]:
```
0
```

In [202]:
```
df['ward'].isna().sum()
```

Out[202]:
```
0
```

In [203]:
```
df['lga'].value_counts().head(20)
```

Out[203]:
```
Njombe           2503
Arusha Rural     1252
Moshi Rural      1251
Bariadi          1177
Rungwe           1106
Kilosa           1094
Kasulu           1047
Mbozi            1034
Meru             1009
Bagamoyo          997
Singida Rural     995
Kilombero         959
Same              877
Kibondo           874
Kyela             859
Kahama            836
Magu              824
Kigoma Rural      824
Maswa             809
Karagwe           771
Name: lga, dtype: int64
```

In [204]:
```
df['lga'].nunique()
```

Out[204]:
```
125
```

In [205]:
```
df['ward'].nunique()
```

Out[205]:
```
2092
```

In [206]:
```
df['ward'].isna().sum()
```

Out[206]:
```
0
```

In [207]:
```
df['ward'].value_counts().head(20)
```

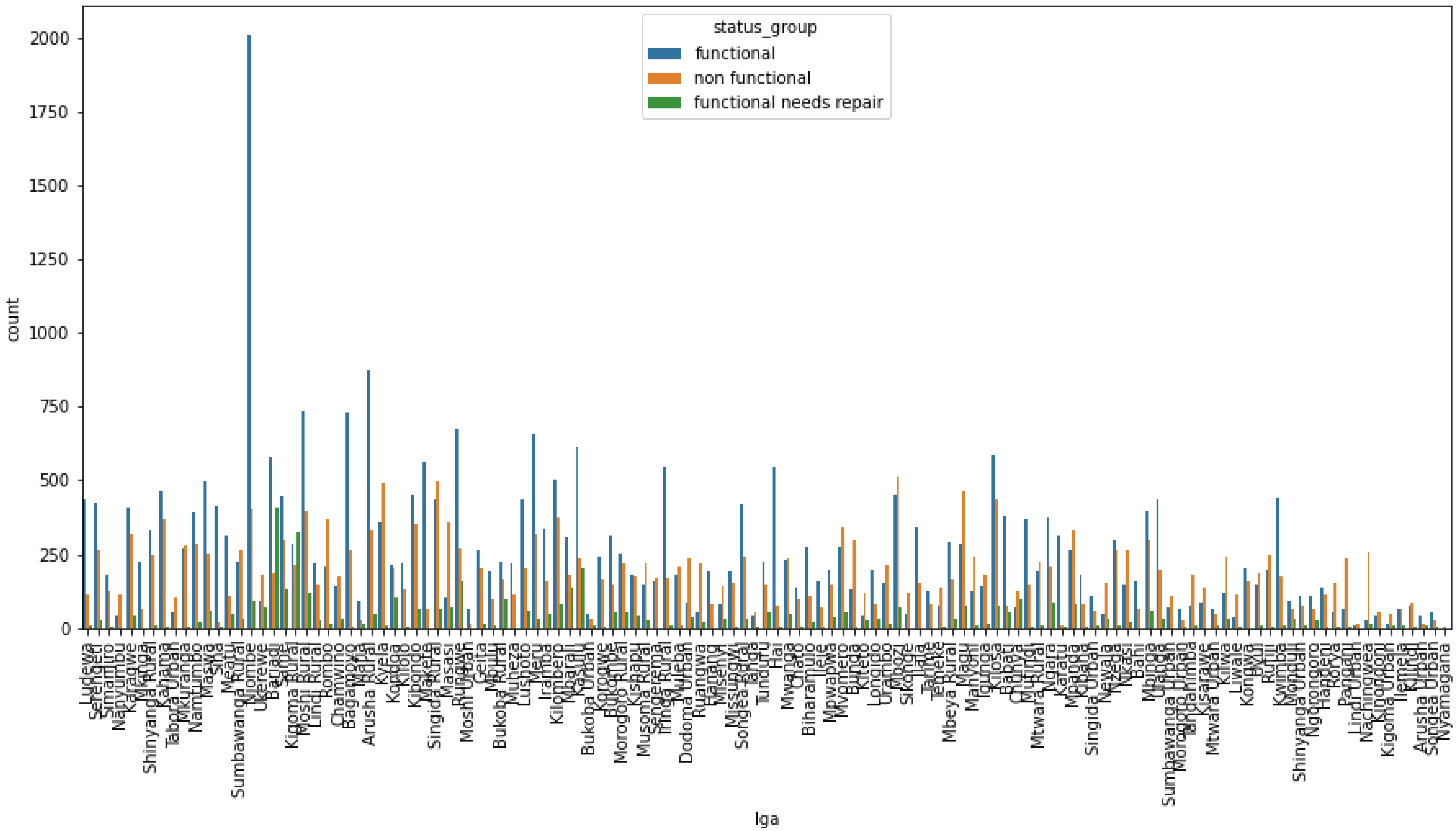
Out[207]:
```
Igosi            307
Imalinyi         252
Siha Kati        232
Mdandu           231
Nduruma          217
Mishamo          203
Kitunda          203
Msindo           201
Chalinze         196
Maji ya Chai     190
Usuka            187
Ngarenanyuki     172
Chanika          171
Vikindu          162
Mtwango          153
Matola           145
Zinga/Ikerege    141
Wanging'ombe     139
Maramba          139
Itete            137
Name: ward, dtype: int64
```

In [208]:
```
df.groupby(["lga", "ward"]).size().head(50)
```

Out[208]:
```
lga           ward
Arusha Rural  Bangata          33
              Bwawani          37
              Ilkiding'a       86
              Kimnyaki         79
              Kiranyi         115
              Kisongo          33
              Mateves          22
              Mlangarini       92
              Moivo            44
              Moshono          44
              Murieti          29
              Musa             29
              Mwandeti         17
              Nduruma         205
              Oldonyosambu     77
              Oljoro            8
              Olkokola        133
              Oltroto          75
              Oltrumet         52
              Sokoni II        42
Arusha Urban  Baraa             2
              Daraja Mbili      3
              Elerai           11
              Engutoto          1
              Kaloleni          5
              Kimandolu         2
              Lemara            4
              Levolosi          2
              Ngarenaro         3
              Olorien           4
              Sekei             3
              Sokon I           4
              Sombetini         4
              Terrat            8
              Themi             1
              Unga Ltd          6
Babati        Arri             19
              Bashinet         19
              Bonga            11
              Dabil            47
              Dareda           52
              Duru             15
              Gidas            19
              Madunga          24
```

```
                Magara          38
                Magugu          40
                Mamire         115
                Mwada           15
                Nkaiti          22
                Qash            13
dtype: int64
```

In [209]:
```python
plt.figure(figsize=(15,7))
ax = sns.countplot(x='lga', hue="status_group", data=df)
ax.tick_params(axis='x', rotation=90)
```



For now we will remove 'ward' and keep "lga"

In [210]:
```
df.drop(columns='ward',inplace=True )
```

3.5.2.14 Column 'population'

In [211]:
```
df["population"].isna().sum()
```

Out[211]:
```
0
```

In [212]:
```
df["population"].nunique()
```

Out[212]:
```
1049
```

In [213]:
```
df["population"].value_counts().head(20)
```

Out[213]:
```
0      21381
1       7025
200     1940
150     1892
250     1681
300     1476
100     1146
50      1139
500     1009
350      986
120      916
400      775
60       706
30       626
40       552
80       533
450      499
20       462
600      438
230      388
Name: population, dtype: int64
```

This fearure does not have missing values but most of the values are zero. let us explore more

In [214]:
```
df.loc[df["population"]!=0].describe()
```
Out[214]:

	id	amount_tsh	gps_height	longitude	latitude	district_code	population	construction_year
count	38019.000000	38019.000000	38019.000000	38019.000000	38019.000000	38019.000000	38019.000000	38019.000000
mean	37107.559115	447.787681	969.889634	36.074387	-6.139781	6.299456	281.087167	1961.399721
std	21406.803661	3706.770967	612.544787	2.586779	2.737733	11.303334	564.687660	263.994165
min	1.000000	0.000000	-90.000000	29.607122	-11.649440	1.000000	1.000000	0.000000
25%	18514.500000	0.000000	347.000000	34.715340	-8.388839	2.000000	40.000000	1986.000000
50%	37128.000000	0.000000	1135.000000	36.706815	-5.750877	3.000000	150.000000	2000.000000
75%	55505.500000	100.000000	1465.000000	37.940149	-3.597016	5.000000	324.000000	2008.000000
max	74247.000000	350000.000000	2770.000000	40.345193	-1.042375	67.000000	30500.000000	2013.000000

We will replace zeros with mean value

In [215]:
```
df['population'].replace(to_replace = 0, value = 281.087167 , inplace=True)
```
In [216]:
```
df["population"].describe()
```
Out[216]:
```
count    59400.000000
mean       281.087167
std        451.765813
min          1.000000
25%        100.000000
50%        281.087167
75%        281.087167
max      30500.000000
Name: population, dtype: float64
```

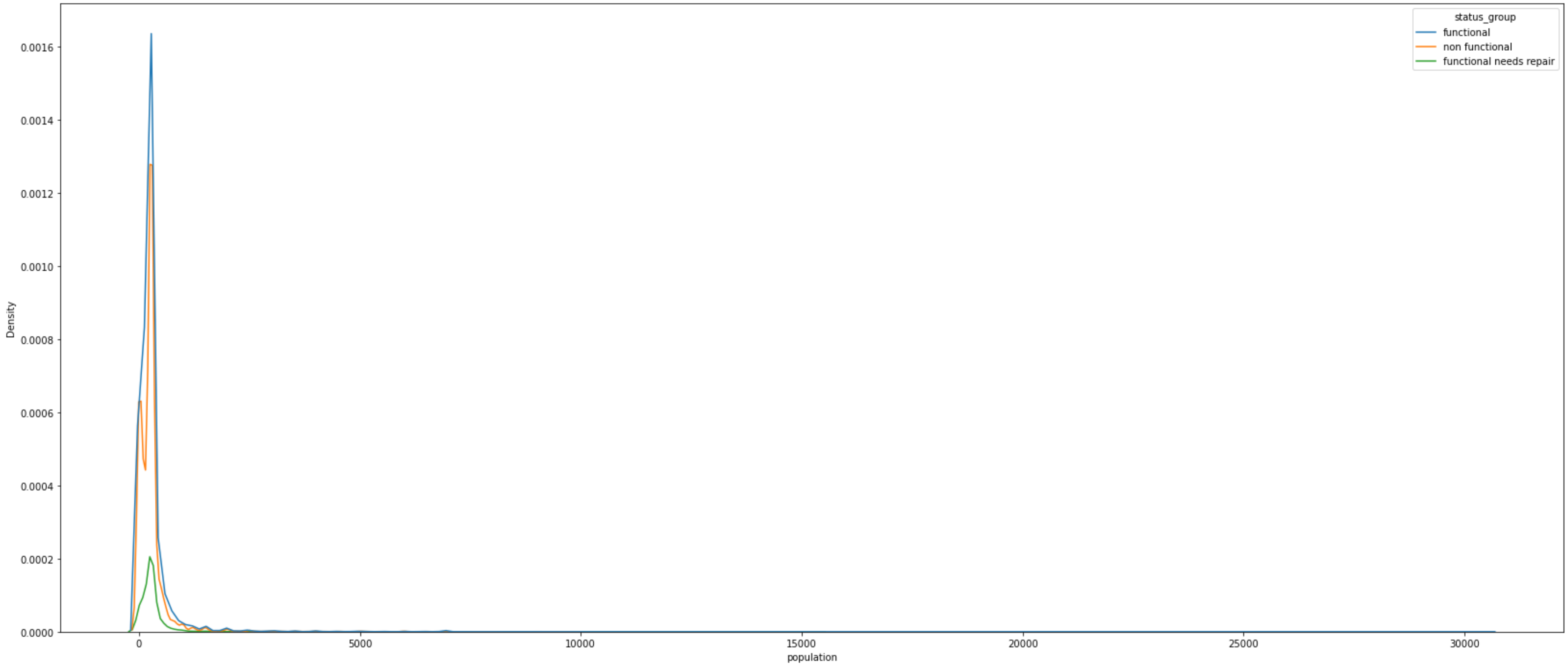
In [217]:
```
df["population"].value_counts().head(20)
```

Out[217]:
```
281.087167    21381
1.000000       7025
200.000000     1940
150.000000     1892
250.000000     1681
300.000000     1476
100.000000     1146
50.000000      1139
500.000000     1009
350.000000      986
120.000000      916
400.000000      775
60.000000       706
30.000000       626
40.000000       552
80.000000       533
450.000000      499
20.000000       462
600.000000      438
230.000000      388
Name: population, dtype: int64
```

In [389]:
```
#df.groupby(['population', 'status_group']).size()
```

In [218]:
```python
plt.figure(figsize=(28,12))
sns.kdeplot(data=df, x='population', hue="status_group", gridsize=200)
```

Out[218]: <AxesSubplot:xlabel='population', ylabel='Density'>



From the above kd plot it is understood that the more populated area is, higher the chances of the waterpoint being functional

3.5.2.14 Column 'public_meeting'

In [219]:
```
df['public_meeting'].value_counts()
```

Out[219]:
```
True     51011
False     5055
Name: public_meeting, dtype: int64
```

In [220]:
```
df['public_meeting'].isna().sum()
```

Out[220]:
```
3334
```

there are 3334 missing values, we will replace those with most occurring values i.e. 'True'

In [221]:
```
df['public_meeting'].fillna(value=True, inplace=True)
```

In [222]:
```
df['public_meeting'].value_counts()
```

Out[222]:
```
True     54345
False     5055
Name: public_meeting, dtype: int64
```

3.5.2.14 Column 'recorded_by'

In [223]:
```
df['recorded_by'].isna().sum()
```

Out[223]:
```
0
```

In [224]:
```
df['recorded_by'].value_counts()
```

Out[224]:
```
GeoData Consultants Ltd    59400
Name: recorded_by, dtype: int64
```

This featured will not be useful as it has only one value. hence we will remove the same.

In [225]:
```python
df.drop(columns='recorded_by', inplace = True)
```

In [226]:
```python
df.shape
```

Out[226]:
```
(59400, 35)
```

3.5.2.15 Column "scheme_management", "scheme_name", "management", "management_group"

In [227]:
```python
print(df['scheme_management'].isna().sum())
print(df['scheme_management'].nunique())
df['scheme_management'].value_counts()
```

```
3877
12
```

Out[227]:
```
VWC                 36793
WUG                  5206
Water authority      3153
WUA                  2883
Water Board          2748
Parastatal           1680
Private operator     1063
Company              1061
Other                 766
SWC                    97
Trust                  72
None                    1
Name: scheme_management, dtype: int64
```

In [301]:
```python
print(df['scheme_name'].isna().sum())
print(df['scheme_name'].nunique())
df['scheme_name'].value_counts()
```

In [229]:
```
print(df['management'].isna().sum())
print(df['management'].nunique())
df['management'].value_counts()
```

```
0
12
```

Out[229]:
```
vwc                 40507
wug                  6515
water board          2933
wua                  2535
private operator     1971
parastatal           1768
water authority       904
other                 844
company               685
unknown               561
other - school         99
trust                  78
Name: management, dtype: int64
```

In [230]:
```
print(df['management_group'].isna().sum())
print(df['management_group'].nunique())
df['management_group'].value_counts()
```

```
0
5
```

Out[230]:
```
user-group    52490
commercial     3638
parastatal     1768
other           943
unknown         561
Name: management_group, dtype: int64
```

the columns 'management' & 'scheme_management' has almost similar categorical values. So it is better to drop one of them. as can be seen above, 'management' has zero missing values whereas 'scheme_management' has 3877 missing values. In view of this we will drop the column 'scheme_management'

In [231]:
```
df.drop(columns=['scheme_management'], inplace=True)
```

In [232]:
```
df.shape
```

Out[232]:
```
(59400, 34)
```

In [233]:
```
df.groupby(['management_group','management']).size()
```

Out[233]:
```
management_group  management
commercial        company               685
                  private operator     1971
                  trust                  78
                  water authority       904
other             other                 844
                  other - school         99
parastatal        parastatal           1768
unknown           unknown               561
user-group        vwc                 40507
                  water board          2933
                  wua                  2535
                  wug                  6515
dtype: int64
```

From above analysis it is understood that the column 'management_group' contains group of categories present in the column 'management'. Hence we will drop the column 'management_group' which has less information as compared to 'management'

In [234]:
```
df.drop(columns='management_group', inplace=True)
```

In [235]:
```
df.shape
```

Out[235]:
```
(59400, 33)
```

Further, the column 'scheme_name' has 28166 missing values and 2696 categories. We will drop the same for now.

In [236]:
```
df.drop(columns='scheme_name', inplace=True)
```

In [237]:
```
df.shape
```

Out[237]:
```
(59400, 32)
```

3.5.2.16 Column "permit"

In [238]:
```
df['permit'].value_counts()
```

Out[238]:
```
True     38852
False    17492
Name: permit, dtype: int64
```

In [239]:
```
df['permit'].isna().sum()
```

Out[239]:
```
3056
```

There are 3334 missing values, we will replace those with most occurring values i.e. 'True'

In [240]:
```
df['permit'].fillna(value=True, inplace=True)
```

In [241]:
```
df['permit'].value_counts()
```

Out[241]:
```
True     41908
False    17492
Name: permit, dtype: int64
```

3.5.2.17 Column "construction_year"

In [242]:
```
print(df['construction_year'].isna().sum())
print(df['construction_year'].nunique())
```

```
0
55
```

In [243]:
```
df['construction_year'].value_counts()
```

Out[243]:
```
0       20709
2010     2645
2008     2613
2009     2533
2000     2091
2007     1587
2006     1471
2003     1286
2011     1256
2004     1123
2012     1084
2002     1075
1978     1037
1995     1014
2005     1011
1999      979
1998      966
1990      954
1985      945
```

We can make use of this column in conjunction with the column 'date_recorded'. We can get time difference between the year which the waterpoint was contrcted in and the year which the data was recorded in. This time difference is nothing but the time which the waterpoint has been operational for. This way we can create a new feature called 'Operational_years'

first we will replace the missing values present in 'construction_year'

In [244]:
```
df.loc[df['construction_year']!=0].describe() #to know the statistic of feature without zero values
```

Out[244]:

	id	amount_tsh	gps_height	longitude	latitude	district_code	population	construction_year
count	38691.000000	38691.000000	38691.000000	38691.000000	38691.000000	38691.000000	38691.000000	38691.000000
mean	37083.008736	466.457534	1002.367760	35.983262	-6.235372	5.969786	279.585470	1996.814686
std	21420.922010	3541.036030	618.078669	2.558709	2.761317	10.700673	549.961837	12.472045
min	1.000000	0.000000	-63.000000	29.607122	-11.649440	1.000000	1.000000	1960.000000
25%	18489.500000	0.000000	372.000000	34.676719	-8.755274	2.000000	40.000000	1987.000000
50%	37078.000000	0.000000	1154.000000	36.648187	-6.064216	3.000000	150.000000	2000.000000
75%	55514.500000	200.000000	1488.000000	37.803940	-3.650661	5.000000	305.000000	2008.000000
max	74247.000000	350000.000000	2770.000000	40.345193	-1.042375	63.000000	30500.000000	2013.000000

In [245]:
```
df['construction_year'].replace(to_replace = 0, value = 1996, inplace=True)
#replacing the missing values in construction_year column with mean value i.e. 1996
```

In [246]:
```
df.describe()
```

Out[246]:

	id	amount_tsh	gps_height	longitude	latitude	district_code	population	construction_year
count	59400.000000	59400.000000	59400.000000	59400.000000	5.940000e+04	59400.000000	59400.000000	59400.000000
mean	37115.131768	317.650385	668.297239	35.116960	-5.706033e+00	5.629747	281.087167	1996.530657
std	21453.128371	2997.574558	693.116350	2.573963	2.946019e+00	9.633649	451.765813	10.073265
min	0.000000	0.000000	-90.000000	29.607122	-1.164944e+01	0.000000	1.000000	1960.000000
25%	18519.750000	0.000000	0.000000	33.354079	-8.540621e+00	2.000000	100.000000	1996.000000
50%	37061.500000	0.000000	369.000000	34.908743	-5.021597e+00	3.000000	281.087167	1996.000000
75%	55656.500000	20.000000	1319.250000	37.178387	-3.326156e+00	5.000000	281.087167	2004.000000
max	74247.000000	350000.000000	2770.000000	40.345193	-2.000000e-08	80.000000	30500.000000	2013.000000

creating new column 'operational_years'

In [247]:
```
print(df.date_recorded.head(5))
print(df.construction_year.head(5))
```

```
0    2011-03-14
1    2013-03-06
2    2013-02-25
3    2013-01-28
4    2011-07-13
Name: date_recorded, dtype: object
0    1999
1    2010
2    2009
3    1986
4    1996
Name: construction_year, dtype: int64
```

In [248]:
```
df['date_recorded'] = pd.to_datetime(df['date_recorded']) #converting dates to 'datetime' datatype
```

In [249]:
```
df['date_recorded'].head()
```

Out[249]:
```
0   2011-03-14
1   2013-03-06
2   2013-02-25
3   2013-01-28
4   2011-07-13
Name: date_recorded, dtype: datetime64[ns]
```

In [250]:
```
df.date_recorded.dt.year.head(5)
```

Out[250]:
```
0    2011
1    2013
2    2013
3    2013
4    2011
Name: date_recorded, dtype: int64
```

In [251]:
```python
df['operational_year'] = df.date_recorded.dt.year - df.construction_year
df.operational_year.head(5)
```

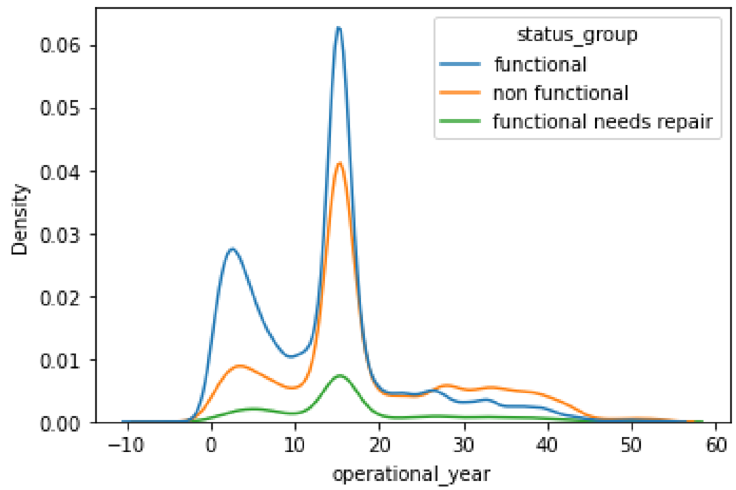
Out[251]:
```
0    12
1     3
2     4
3    27
4    15
Name: operational_year, dtype: int64
```

In [252]:
```python
df['operational_year'].value_counts()
```

Out[252]:
```
 15    14336
 16     5968
 17     2846
 3      2740
 1      2303
 2      2129
 5      1980
 4      1890
 13     1869
 7      1404
 6      1382
 11     1352
 8      1173
 14     1160
 33     1120
 23      905
 10      868
 9       814
 19      766
```

In [253]:
```
sns.kdeplot(data=df, x='operational_year', hue="status_group", gridsize=200)
```

Out[253]: <AxesSubplot:xlabel='operational_year', ylabel='Density'>



In [254]:
```
# plt.figure(figsize=(26,15))
# ax = sns.countplot(data=df, x='operational_year', hue="status_group")
```

There are some anamolies in data as some of the values in 'operational_year' are negative and operatinal years cant be negative. we will replace the negative values with minimum value.

In [255]:
```
df.loc[~df['operational_year']<0].describe()
```

Out[255]:

	id	amount_tsh	gps_height	longitude	latitude	district_code	population	construction_year	operational_year
count	59391.000000	59391.000000	59391.000000	59391.000000	5.939100e+04	59391.000000	59391.000000	59391.000000	59391.000000
mean	37116.883753	317.687240	668.297688	35.116650	-5.705857e+00	5.628597	281.086305	1996.528919	15.393949
std	21452.886206	2997.799583	693.118358	2.573822	2.946049e+00	9.632382	451.785541	10.073017	10.089123
min	0.000000	0.000000	-90.000000	29.607122	-1.164944e+01	0.000000	1.000000	1960.000000	0.000000
25%	18522.500000	0.000000	0.000000	33.353967	-8.540784e+00	2.000000	100.000000	1996.000000	8.000000
50%	37063.000000	0.000000	369.000000	34.908362	-5.021241e+00	3.000000	281.087167	1996.000000	15.000000
75%	55661.000000	20.000000	1319.000000	37.177970	-3.326129e+00	5.000000	281.087167	2004.000000	17.000000
max	74247.000000	350000.000000	2770.000000	40.345193	-2.000000e-08	80.000000	30500.000000	2013.000000	53.000000

we will replace the negative values in 'operational_year' with the minimum value which is zero

In [256]:
```
df.loc[df['operational_year']<0, 'operational_year'] = 0
```

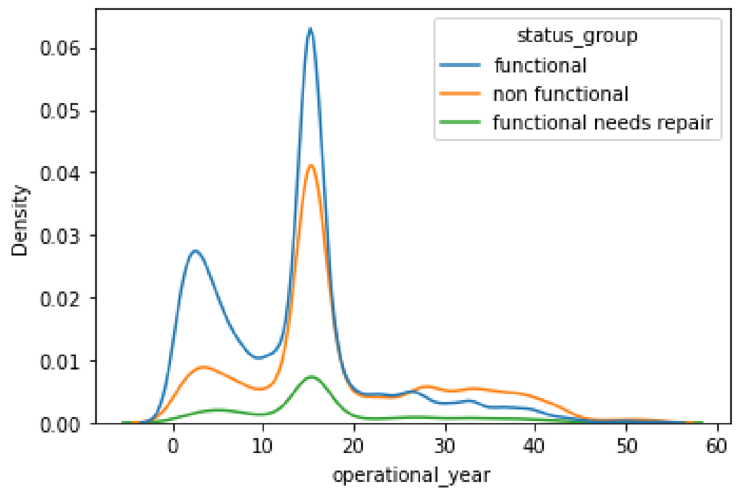
In [257]:
```
df['operational_year'].value_counts().head(20)
```

Out[257]:
```
15    14336
16     5968
17     2846
3      2740
1      2303
2      2129
5      1980
4      1890
13     1869
7      1404
6      1382
11     1352
8      1173
14     1160
33     1120
23      905
10      868
9       814
19      766
```

In [258]:
```
# f = lambda x: 0 if x<0 else 1
# df['my_column'] = df['my_column'].map(f) #above can be done using map fuction
```

In [259]:
```
sns.kdeplot(data=df, x='operational_year', hue="status_group", gridsize=200)
```

Out[259]:
```
<AxesSubplot:xlabel='operational_year', ylabel='Density'>
```



From above plot, we can see that the negative values have disappeared. We will now drop columns 'construction_year' & 'date_recorded'

In [260]:
```
df.drop(columns=["construction_year", "date_recorded"], inplace=True)
```

In [261]:
```
df.shape
```

Out[261]:
```
(59400, 31)
```

In [262]:
```
df.columns
```

Out[262]:
```
Index(['id', 'amount_tsh', 'funder', 'gps_height', 'installer', 'longitude',
       'latitude', 'basin', 'region', 'district_code', 'lga', 'population',
       'public_meeting', 'permit', 'extraction_type', 'extraction_type_group',
       'extraction_type_class', 'management', 'payment', 'payment_type',
       'water_quality', 'quality_group', 'quantity', 'quantity_group',
       'source', 'source_type', 'source_class', 'waterpoint_type',
       'waterpoint_type_group', 'status_group', 'operational_year'],
      dtype='object')
```

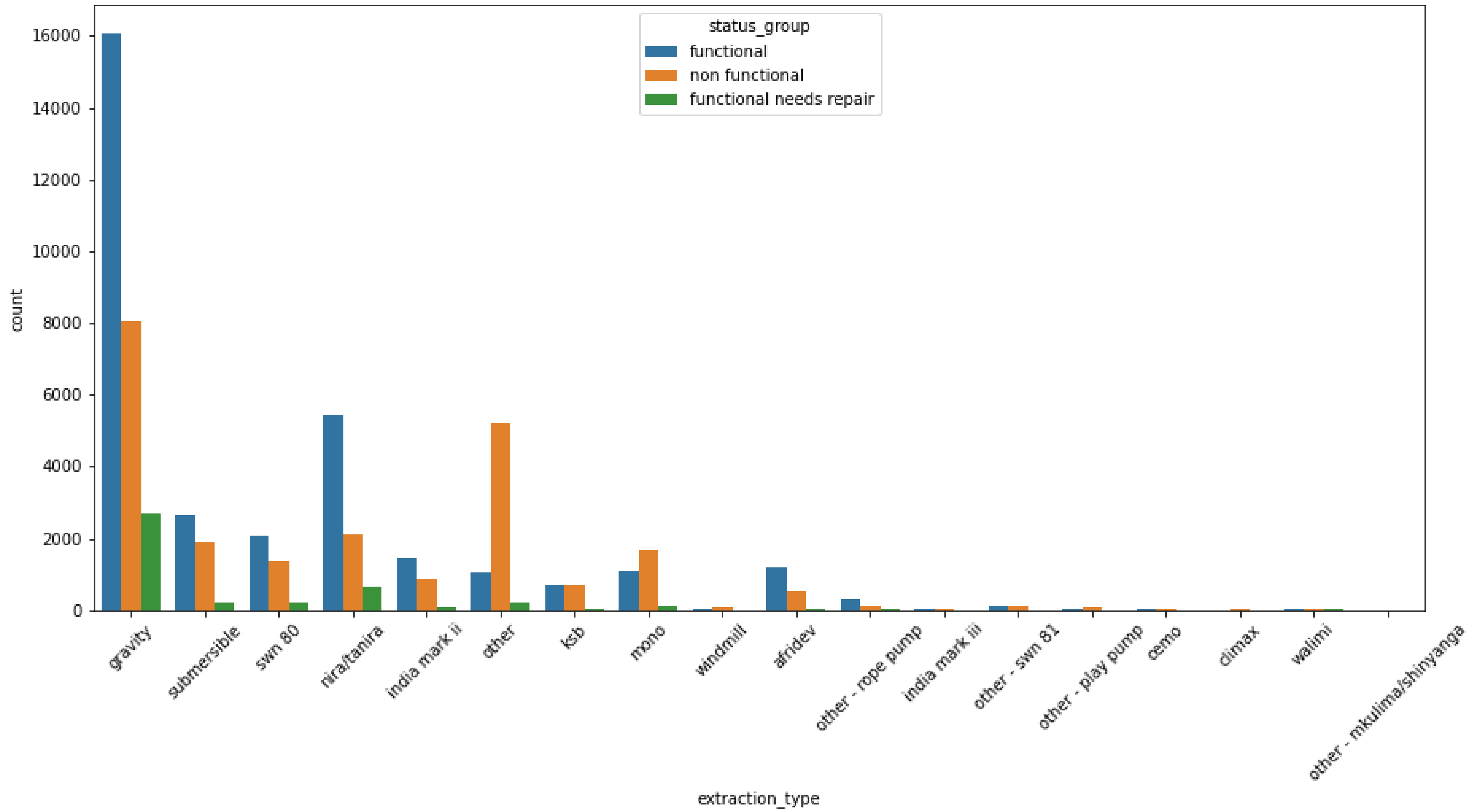
3.5.2.18 Column "extraction_type", "extraction_type_group", "extraction_type_class"

In [263]:
```
print(df['extraction_type'].isna().sum())
print(df['extraction_type'].nunique())
df['extraction_type'].value_counts()
```

```
0
18
```

Out[263]:
```
gravity                        26780
nira/tanira                     8154
other                           6430
submersible                     4764
swn 80                          3670
mono                            2865
india mark ii                   2400
afridev                         1770
ksb                             1415
other - rope pump                451
other - swn 81                   229
windmill                         117
india mark iii                    98
cemo                              90
other - play pump                 85
walimi                            48
climax                            32
other - mkulima/shinyanga          2
Name: extraction_type, dtype: int64
```

In [264]:
```
plt.figure(figsize=(15,7))
ax = sns.countplot(x='extraction_type', hue="status_group", data=df)
ax.tick_params(axis='x', rotation=45)
# plt.xticks(rotation=90)
```



In [265]:
```
print(df['extraction_type_group'].isna().sum())
print(df['extraction_type_group'].nunique())
df['extraction_type_group'].value_counts()
```

```
0
13
```

Out[265]:
```
gravity            26780
nira/tanira         8154
other               6430
submersible         6179
swn 80              3670
mono                2865
india mark ii       2400
afridev             1770
rope pump            451
other handpump       364
other motorpump      122
wind-powered         117
india mark iii        98
Name: extraction_type_group, dtype: int64
```

In [266]:
```
print(df['extraction_type_class'].isna().sum())
print(df['extraction_type_class'].nunique())
df['extraction_type_class'].value_counts()
```

```
0
7
```

Out[266]:
```
gravity         26780
handpump        16456
other            6430
submersible      6179
motorpump        2987
rope pump         451
wind-powered      117
Name: extraction_type_class, dtype: int64
```

In [267]:
```
df.groupby(['extraction_type_class', 'extraction_type']).size()
```

Out[267]:
```
extraction_type_class  extraction_type
gravity                gravity                        26780
handpump               afridev                         1770
                       india mark ii                   2400
                       india mark iii                    98
                       nira/tanira                     8154
                       other - mkulima/shinyanga          2
                       other - play pump                 85
                       other - swn 81                   229
                       swn 80                          3670
                       walimi                            48
motorpump              cemo                              90
                       climax                            32
                       mono                            2865
other                  other                           6430
rope pump              other - rope pump                451
submersible            ksb                             1415
                       submersible                     4764
wind-powered           windmill                         117
dtype: int64
```

In [268]:
```
df.groupby(['extraction_type_class', 'extraction_type_group']).size()
```

Out[268]:
```
extraction_type_class  extraction_type_group
gravity                gravity                 26780
handpump               afridev                  1770
                       india mark ii            2400
                       india mark iii             98
                       nira/tanira              8154
                       other handpump            364
                       swn 80                   3670
motorpump              mono                     2865
                       other motorpump           122
other                  other                    6430
rope pump              rope pump                 451
submersible            submersible              6179
wind-powered           wind-powered              117
dtype: int64
```

"extraction_type", "extraction_type_group" & "extraction_type_class" provide the details as regard the type of pumping system being used. Here, 'extraction_type_class' is sort of group version of "extraction_type" & "extraction_type_group". Here we will chose to keep "extraction_type_group" and remove other two. The reason we are removing "extraction_type" is because it has few categories which have very less no. of values.

In [269]:
```
df.drop(columns=["extraction_type", "extraction_type_class"], inplace=True)
```

In [270]:
```
df.shape
```

Out[270]:
```
(59400, 29)
```

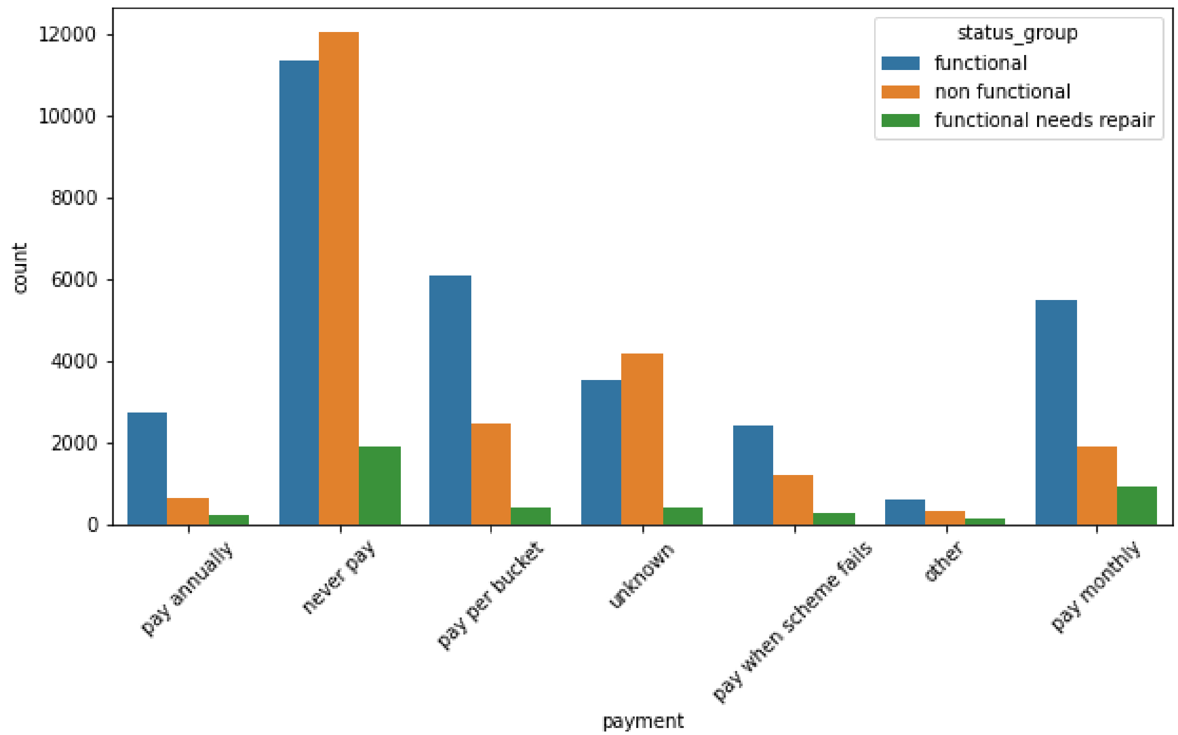
3.5.2.19 Column "payment", "payment_type"

In [271]:
```
print(df['payment'].isna().sum())
print(df['payment'].nunique())
df['payment'].value_counts()
```

```
0
7
```

Out[271]:
```
never pay               25348
pay per bucket           8985
pay monthly              8300
unknown                  8157
pay when scheme fails    3914
pay annually             3642
other                    1054
Name: payment, dtype: int64
```

In [272]:
```
plt.figure(figsize=(10,5))
ax = sns.countplot(x='payment', hue="status_group", data=df)
ax.tick_params(axis='x', rotation=45)
```



In [273]:
```
print(df['payment_type'].isna().sum())
print(df['payment_type'].nunique())
df['payment_type'].value_counts()
```

```
0
7
```

Out[273]:
```
never pay     25348
per bucket     8985
monthly        8300
unknown        8157
on failure     3914
annually       3642
other          1054
Name: payment_type, dtype: int64
```

Here both features are providing similar information. We will remove one of them.

In [274]:
```
df.drop(columns='payment_type', inplace=True)
```

In [275]:
```
df.shape
```

Out[275]:
```
(59400, 28)
```

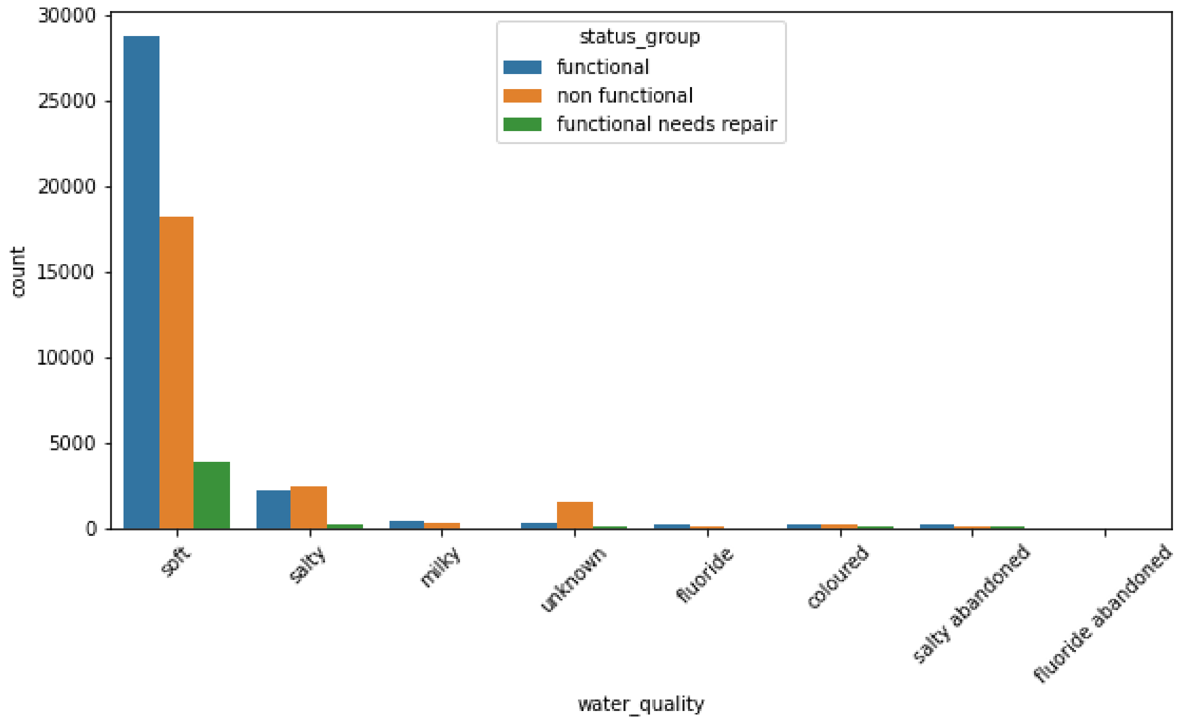
3.5.2.20 Column "water_quality", "quality_group", "quantity", "quantity_group"

In [276]:
```
print(df['water_quality'].isna().sum())
print(df['water_quality'].nunique())
df['water_quality'].value_counts()
```

```
0
8
```

Out[276]:
```
soft                  50818
salty                  4856
unknown                1876
milky                   804
coloured                490
salty abandoned         339
fluoride                200
fluoride abandoned       17
Name: water_quality, dtype: int64
```

In [277]:
```
plt.figure(figsize=(10,5))
ax = sns.countplot(x='water_quality', hue="status_group", data=df)
ax.tick_params(axis='x', rotation=45)
```



In [278]:
```
print(df['quality_group'].isna().sum())
print(df['quality_group'].nunique())
df['quality_group'].value_counts()
```

```
0
6
```

Out[278]:
```
good        50818
salty        5195
unknown      1876
milky         804
colored       490
fluoride      217
Name: quality_group, dtype: int64
```

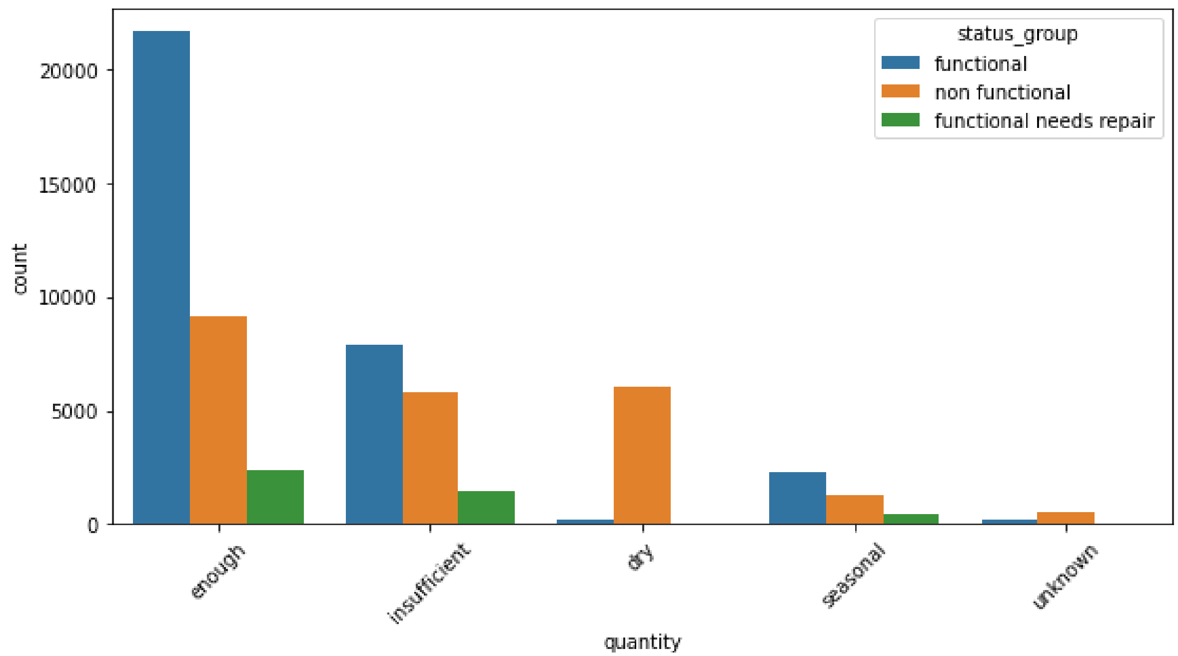
The columns are similar. The column "water_quality" tend to give more informayion than that of "quality_group" so we will drop "quality_group"

In [279]:
```
print(df['quantity'].isna().sum())
print(df['quantity'].nunique())
df['quantity'].value_counts()
```

```
0
5
```

Out[279]:
```
enough          33186
insufficient    15129
dry              6246
seasonal         4050
unknown           789
Name: quantity, dtype: int64
```

In [280]:
```
plt.figure(figsize=(10,5))
ax = sns.countplot(x='quantity', hue="status_group", data=df)
ax.tick_params(axis='x', rotation=45)
```



In [281]:
```
print(df['quantity_group'].isna().sum())
print(df['quantity_group'].nunique())
df['quantity_group'].value_counts()
```

```
0
5
```

Out[281]:
```
enough          33186
insufficient    15129
dry              6246
seasonal         4050
unknown           789
Name: quantity_group, dtype: int64
```

quantity_group & quantity are similar. We will drop one of them

In [282]:
```
df.drop(columns=["quality_group", "quantity_group"], inplace=True)
```

In [283]:
```
df.shape
```

Out[283]:
```
(59400, 26)
```

3.5.2.21 Column "source", "source_type", "source_class"

In [284]:
```
print(df['source'].isna().sum())
print(df['source'].nunique())
df['source'].value_counts()
```

```
0
10
```

Out[284]:
```
spring                  17021
shallow well            16824
machine dbh             11075
river                    9612
rainwater harvesting     2295
hand dtw                  874
lake                      765
dam                       656
other                     212
unknown                    66
Name: source, dtype: int64
```

In [285]:
```
print(df['source_type'].isna().sum())
print(df['source_type'].nunique())
df['source_type'].value_counts()
```

```
0
7
```

Out[285]:
```
spring                  17021
shallow well            16824
borehole                11949
river/lake              10377
rainwater harvesting     2295
dam                       656
other                     278
Name: source_type, dtype: int64
```

In [286]:
```
print(df['source_class'].isna().sum())
print(df['source_class'].nunique())
df['source_class'].value_counts()
```

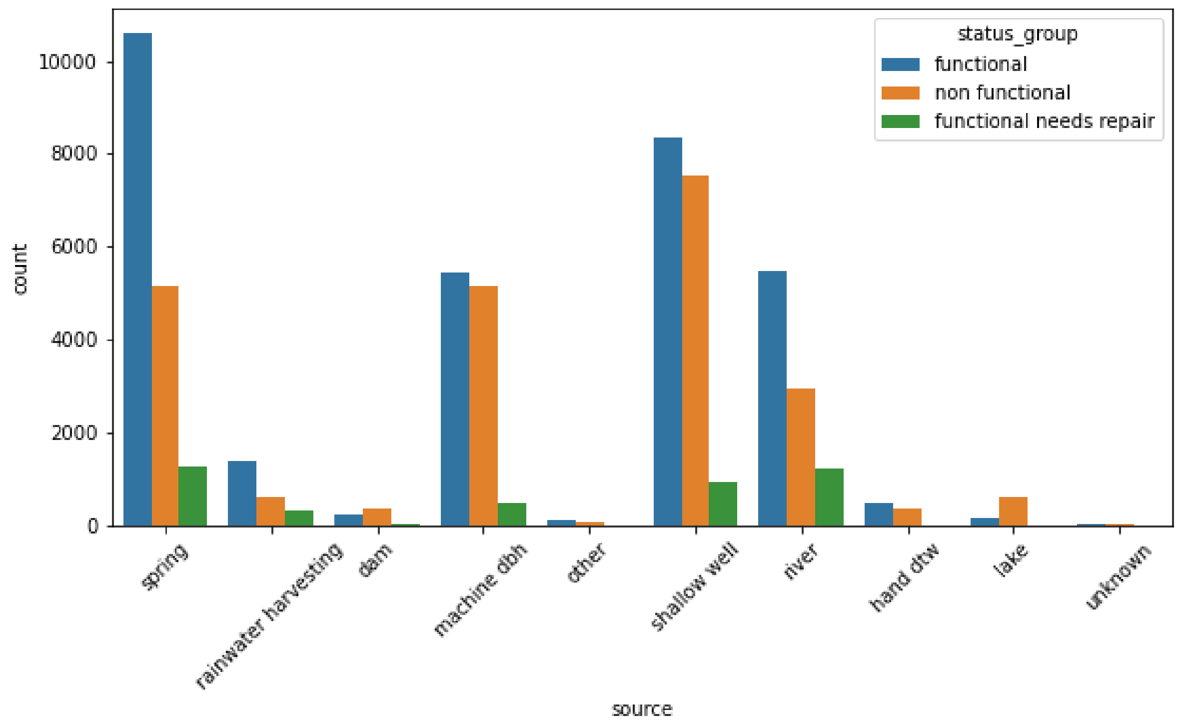
```
0
3
```

Out[286]:
```
groundwater    45794
surface        13328
unknown          278
Name: source_class, dtype: int64
```

In [287]:
```
df.groupby(['source_class', 'source']).size()
```

Out[287]:
```
source_class  source
groundwater   hand dtw                  874
              machine dbh             11075
              shallow well            16824
              spring                  17021
surface       dam                       656
              lake                      765
              rainwater harvesting     2295
              river                    9612
unknown       other                     212
              unknown                    66
dtype: int64
```

In [288]:
```
plt.figure(figsize=(10,5))
ax = sns.countplot(x='source', hue="status_group", data=df)
ax.tick_params(axis='x', rotation=45)
```



Among above 3 columns we will choose to keep 'source' as it hase more information. We will drop rest two columns.

In [289]:
```
df.drop(columns=["source_type", "source_class"], inplace=True)
```

In [290]:
```
df.shape
```

Out[290]:
```
(59400, 24)
```

3.5.2.22 Column "waterpoint_type", "waterpoint_type_group"

In [291]:
```
print(df['waterpoint_type'].isna().sum())
print(df['waterpoint_type'].nunique())
df['waterpoint_type'].value_counts()
```

```
0
7
```

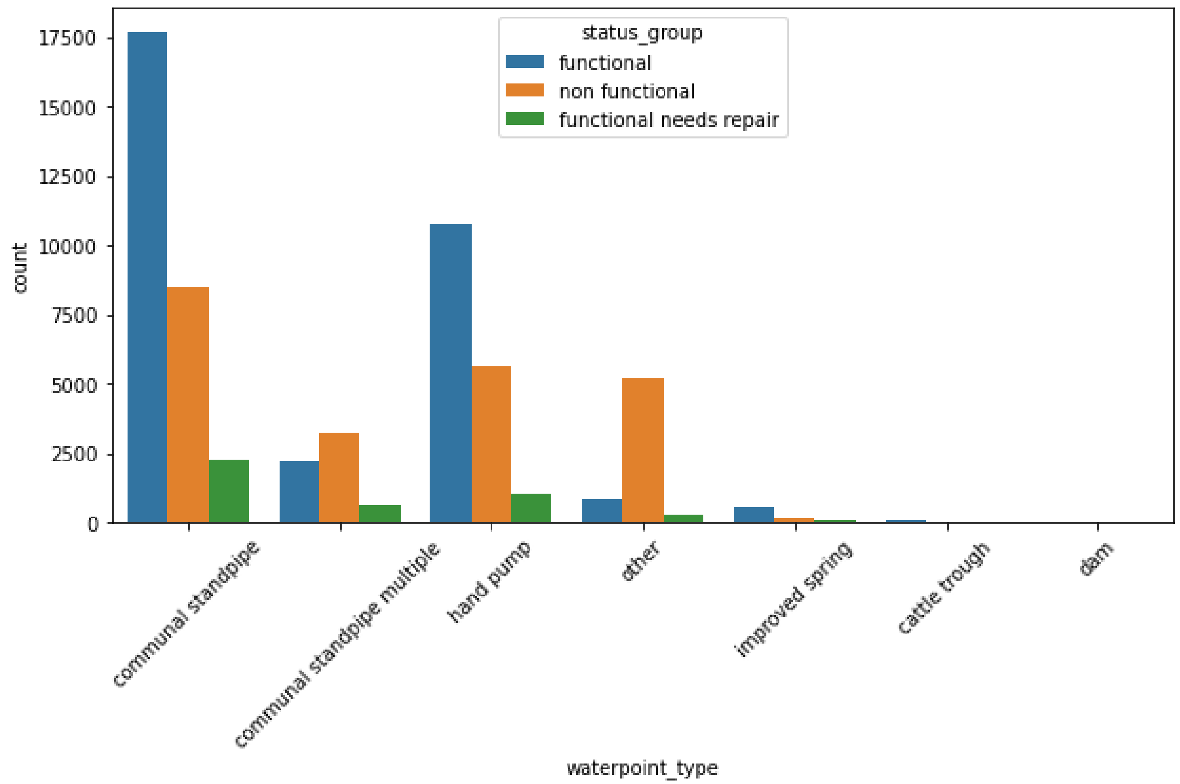
Out[291]:
```
communal standpipe             28522
hand pump                      17488
other                           6380
communal standpipe multiple     6103
improved spring                  784
cattle trough                    116
dam                                7
Name: waterpoint_type, dtype: int64
```

In [292]:
```
print(df['waterpoint_type_group'].isna().sum())
print(df['waterpoint_type_group'].nunique())
df['waterpoint_type_group'].value_counts()
```

```
0
6
```

Out[292]:
```
communal standpipe    34625
hand pump             17488
other                  6380
improved spring         784
cattle trough           116
dam                       7
Name: waterpoint_type_group, dtype: int64
```

In [293]:
```
plt.figure(figsize=(10,5))
ax = sns.countplot(x='waterpoint_type', hue="status_group", data=df)
ax.tick_params(axis='x', rotation=45)
```



here we will drop the column "waterpoint_type_group" as it has less information

In [294]:
```
df.drop(columns=["waterpoint_type_group"], inplace=True)
```

In [295]:
```
df.columns
```

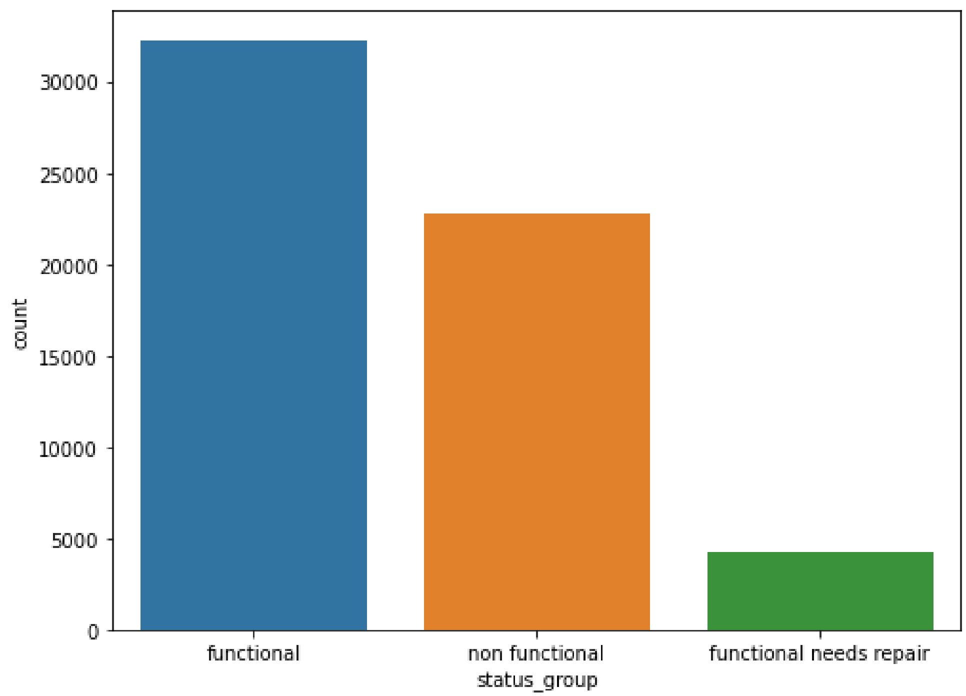
Out[295]:
```
Index(['id', 'amount_tsh', 'funder', 'gps_height', 'installer', 'longitude',
       'latitude', 'basin', 'region', 'district_code', 'lga', 'population',
       'public_meeting', 'permit', 'extraction_type_group', 'management',
       'payment', 'water_quality', 'quantity', 'source', 'waterpoint_type',
       'status_group', 'operational_year'],
      dtype='object')
```

In [296]:
```
df.shape
```

Out[296]:
```
(59400, 23)
```

3.5.2.23 Column "status_group"

In [297]:
```
plt.figure(figsize=(8,6))
ax = sns.countplot(x="status_group", data=df)
```



From above plot of class labels i.e. "status_group", it is understood that the data is highly imbalanced.

In [298]:
```
df.head(5)
```
Out[298]:

	id	amount_tsh	funder	gps_height	installer	longitude	latitude	basin	region	district_code	lga	population	public_meeting	permit	e
0	69572	6000.0	other	1390	other	34.938093	-9.856322	Lake Nyasa	Iringa	5	Ludewa	109.000000	True	False	
1	8776	0.0	other	1399	other	34.698766	-2.147466	Lake Victoria	Mara	2	Serengeti	280.000000	True	True	
2	34310	25.0	other	686	World Vision	37.460664	-3.821329	Pangani	Manyara	4	Simanjiro	250.000000	True	True	
3	67743	0.0	Unicef	263	UNICEF	38.486161	-11.155298	Ruvuma / Southern Coast	Mtwara	63	Nanyumbu	58.000000	True	True	
4	19728	0.0	other	0	other	31.130847	-1.825359	Lake Victoria	Kagera	1	Karagwe	281.087167	True	True	

In [299]:
```
df.shape
```
Out[299]:
```
(59400, 23)
```

3.6 exporting cleaned data to CSV

In [300]:
```
df.to_csv('clean_df.csv') #exporting clean data to csv
```

In [ ]:
```

```