PROJECT TITLE:

PUBLIC TRANSPORTATION ANALYSIS USING IBM COGNUS

TEAM MEMBERS:

•Mani@Arun Venkatesh P(2021115060)

•Malarvizhi E (2021115059)

•Mahalakshmi B (2021115058)

•Loksundar (2021115057)

•Sivaananth (2021115310)

PHASE-I:

1.Project Definition:

Project Definition: The project involves analyzing public transportation data to assess service efficiency, on-time performance, and passenger feedback. The objective is to provide insights that support transportation improvement initiatives and enhance the overall public transportation experience. This project includes defining analysis objectives, collecting transportation data, designing relevant visualizations in IBM Cognos, and using code for data analysis.

2.Data Collection:

To proceed with the analysis, we need to acquire the necessary public transportation data. The dataset for this project can be accessed through the following link:

https://www.kaggle.com/datasets/rednivrug/unisys/code

This dataset encompasses a range of information related to public transportation, including schedules, real-time updates, and passenger feedback. Prior to analysis in IBM Cognos, we will download and perform necessary preprocessing. This may involve tasks such as data cleaning, transformation, and handling of missing values, format discrepancies, and potential outliers. This ensures that the dataset is primed for meaningful analysis and visualization.

3.Visualization Technique:

1. Chart Selection: Choose suitable chart types (e.g., bar charts, line charts) based on the nature of the data (e.g., time series, categorical) to effectively represent key performance metrics.

2. Comparative Visuals: Create visualizations that facilitate easy comparison between different aspects of public transportation, such as on-time performance across routes or modes.

3. Interactive Filters: Enable users to interactively filter data by relevant factors like date, route, or mode, allowing for customized views of the information.

4. Clear Labels and Legends: Ensure that visualizations include clear labels and legends to provide context and aid interpretation.

5. Dynamic Elements: Leverage IBM Cognos' capabilities to create dynamic visualizations that can adapt to different data sets and user inputs.

This simplified strategy aims to streamline the visualization process while still providing effective insights into public transportation data.

4.Insights Generation:

The primary aim of this public transportation analysis project is to extract valuable insights that can inform decision-making and improve the overall transportation experience. The insights may encompass:

• Identifying Routes with High or Low Efficiency:

• Determine routes with exceptional or subpar service efficiency metrics to focus on optimization efforts.

• Analyzing Peak Hour Performance:

• Recognize trends in on-time performance during peak travel hours to allocate resources effectively.

• Evaluating Passenger Feedback Trends:

• Understand common feedback themes to address specific concerns and enhance passenger satisfaction.

• Impact of External Factors:

• Investigate how external factors (e.g., weather, holidays) influence transportation efficiency and passenger experience.

• Spotlight on Unusual Patterns:

• Highlight anomalies or irregularities in performance data, which may require special attention.

5.Next Steps:

In the next phase, we'll preprocess the data, ensuring accuracy. We'll then integrate it into IBM Cognos for seamless analysis and visualization. Our focus will be on creating insightful visualizations,

backed by rigorous statistical analysis. Regular collaboration among team members will be pivotal for project success and aligning with defined objectives.

6.Timeline:

A tentative timeline for the project is as follows:

- Data Collection and Preprocessing: 2 weeks
- IBM Cognos Setup and Visualization Design: 3 weeks
- Data Analysis and Insights Generation: 4 weeks
- Documentation and Reporting: 2 weeks
- Review and Finalization: 1 week

Public Transportation Analysis
Phase-2

1. Data Collection and Integration:
- Implement an extensive network of IoT sensors and GPS devices on public transportation vehicles and at stations.
- Collect real-time data on vehicle location, passenger counts, traffic conditions, weather, and maintenance status.
- Utilize APIs to integrate data from various sources, including traffic management systems, weather services, and urban development databases.

2. Data Processing and Analytics:
- Store data in a secure and scalable cloud infrastructure to facilitate real-time processing and analysis.
- Utilize big data analytics and machine learning algorithms to process and extract insights from the collected data.
- Develop data dashboards for transportation authorities to visualize key performance indicators.

3. Predictive Modeling:
- Create predictive models for passenger demand, traffic congestion, and service disruptions.
- Implement machine learning algorithms that continuously update models based on real-time data.
- Utilize predictive modeling to optimize routes, schedules, and vehicle deployment.

4. Passenger-Centric Solutions:
- Develop a user-friendly mobile application for passengers.
- Provide real-time information on vehicle locations, estimated arrival times, and service updates.
- Enable mobile ticketing and contactless payment options.

5. Traffic Management Integration:
- Collaborate with urban traffic management systems to prioritize public transportation vehicles.
- Implement traffic signal synchronization to minimize delays and congestion.

6. Sustainability Initiatives:
- Introduce electric and eco-friendly vehicles into the public transportation fleet.
- Explore renewable energy sources for powering transit systems.
- Monitor and report on the carbon footprint of public transportation.

7. Accessibility Enhancement:
- Invest in infrastructure improvements to enhance accessibility for people with disabilities.
- Implement low-floor buses, ramps, and tactile information for the visually impaired.

8. Public-Private Partnerships:
- Collaborate with private transportation providers to offer a seamless and integrated multi-modal transportation network.
- Facilitate fare integration and shared data to optimize passenger journeys.

9. Real-time Feedback and Crowdsourcing:
- Develop a feedback system within the mobile application for passengers to report issues and provide suggestions.
- Leverage crowdsourced data to identify and address problems in real time.

10. Open Data and APIs:
- Make data on public transportation systems available to developers through open APIs.
- Encourage third-party developers to create innovative applications that enhance the passenger experience and contribute to data analysis.

11. Autonomous Vehicles:
- Research and implement autonomous vehicles in the public transportation system to improve efficiency and reduce operational costs.

12. Public Awareness Campaigns:
- Launch public awareness campaigns to promote the benefits of public transportation, such as reduced congestion and environmental sustainability.

13. Funding and Policy Support:

- Advocate for government funding and supportive policies to implement the IPTOS system and ensure long-term sustainability.

14. Continuous Improvement:

- Establish a dedicated team for monitoring system performance, conducting regular assessments, and making necessary adjustments based on data and passenger feedback.

The Integrated Public Transportation Optimization System (IPTOS) aims to revolutionize public transportation by making it more efficient, accessible, and sustainable, resulting in improved urban mobility and reduced environmental impact.

Public Transportation Analysis
Phase-3

Data Preprocessing :

Data preprocessing in public transportation analysis is a crucial step that involves cleaning, transforming, and organizing raw transportation data to make it suitable for analysis. Public transportation systems generate vast amounts of data from various sources, such as ticketing systems, GPS trackers, sensors, and schedules. Preprocessing this data is necessary to extract meaningful insights, improve data quality, and ensure that it's ready for analytical and modeling tasks. Here are the key aspects of data preprocessing in public transportation analysis:

1. Data Collection:
   - Data collection involves gathering information from various sources, such as fare collection systems, vehicle sensors, passenger counts, and scheduling systems. This raw data can be in different formats and structures.

2. Data Cleaning:
   - Data cleaning is the process of identifying and correcting errors, inconsistencies, and missing values in the dataset. This can include dealing with duplicated records, removing outliers, and addressing data entry errors.

3. Data Integration:
   - Public transportation data often comes from different sources and in various formats. Data integration involves merging, aligning, and transforming data so that it can be analyzed as a cohesive dataset.

4. Data Transformation:
   - Transformation tasks may include converting data into a standardized format, resampling temporal data, and aggregating data to different time intervals (e.g., hourly or daily) to align with analysis requirements.

5. Geospatial Data Processing:
   - Public transportation analysis often involves geospatial data, including GPS coordinates, routes, and geographical boundaries. Preprocessing may involve geocoding, spatial indexing, and the calculation of distances or travel times between locations.

The code used :

```
#!/usr/bin/env python
# coding: utf-8

# In[2]:


#Importing necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns


# In[3]:


#Loading the dataset
data =
pd.read_csv("C:\\Users\\Maha\\Downloads\\Dataset\\PublicTransportDataset.CSV", low_memory=False)
```

In[4]:

#Displaying the first 20 rows
data.head(20)

In[5]:

# Dropping records which have duplicate values
data.drop_duplicates(inplace=True)

In[6]:

# Filling missing values with mean
data.fillna(data.mean(), inplace=True)

In[7]:

# Printing the first few rows
print(data.head())

In[8]:

# Generating descriptive statistics of the dataset
print(data.describe())

In[9]:

Generating concise summary of the dataset

```
print(data.info())


# In[11]:


# Shape of the dataset
print(data.shape)


# In[12]:


# Displaying first few rows after preprocessing
data.head()
```

In [2]:
```
#Importing necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
#Loading the dataset
data = pd.read_csv("C:\\Users\\AbiramiSV\\Downloads\\Dataset\\PublicT
```

In [4]:
```
#Displaying the first 20 rows
data.head(20)
```

Out[4]:

	TripID	RouteID	StopID	StopName	WeekBeginning	NumberOfBoardings
0	23631	100	14156	181 Cross Rd	2013-06-30 00:00:00	1
1	23631	100	14144	177 Cross Rd	2013-06-30 00:00:00	1
2	23632	100	14132	175 Cross Rd	2013-06-30 00:00:00	1
3	23633	100	12266	Zone A Arndale Interchange	2013-06-30 00:00:00	2
4	23633	100	14147	178 Cross Rd	2013-06-30 00:00:00	1
5	23634	100	13907	9A Marion Rd	2013-06-30 00:00:00	1
6	23634	100	14132	175 Cross Rd	2013-06-30 00:00:00	1
7	23634	100	13335	9A Holbrooks Rd	2013-06-30 00:00:00	1
8	23634	100	13875	9 Marion Rd	2013-06-30 00:00:00	1
9	23634	100	13045	206 Holbrooks Rd	2013-06-30 00:00:00	1
10	23635	100	13335	9A Holbrooks Rd	2013-06-30 00:00:00	1
11	23635	100	13383	8A Marion Rd	2013-06-30 00:00:00	1
12	23635	100	13586	8D Marion Rd	2013-06-30 00:00:00	2
13	23635	100	12726	23 Findon Rd	2013-06-30 00:00:00	1
14	23635	100	13813	8K Marion Rd	2013-06-30 00:00:00	1
15	23635	100	14062	20 Cross Rd	2013-06-30 00:00:00	1
16	23636	100	12780	22A Crittenden Rd	2013-06-30 00:00:00	1
17	23636	100	13383	8A Marion Rd	2013-06-30 00:00:00	1
18	23636	100	14154	180 Cross Rd	2013-06-30 00:00:00	2
19	23636	100	13524	8C Marion Rd	2013-06-30 00:00:00	3

In [5]:
```
# Dropping records which have duplicate values
data.drop_duplicates(inplace=True)
```

In [6]:
```
# Filling missing values with mean
data.fillna(data.mean(), inplace=True)
```

In [7]:
```
# Printing the first few rows
print(data.head())
```

```
   TripID RouteID  StopID                    StopName        WeekBeg
inning  \
0   23631     100   14156                181 Cross Rd  2013-06-30 0
0:00:00
1   23631     100   14144                177 Cross Rd  2013-06-30 0
0:00:00
2   23632     100   14132                175 Cross Rd  2013-06-30 0
0:00:00
3   23633     100   12266  Zone A Arndale Interchange  2013-06-30 0
0:00:00
4   23633     100   14147                178 Cross Rd  2013-06-30 0
0:00:00

   NumberOfBoardings
0                  1
1                  1
2                  1
3                  2
4                  1
```

In [8]:
```
# Generating descriptive statistics of the dataset
print(data.describe())
```

```
             TripID        StopID  NumberOfBoardings
count  1.085723e+07  1.085723e+07       1.085723e+07
mean   2.952100e+04  1.366132e+04       4.743737e+00
std    1.960938e+04  1.971760e+03       9.382286e+00
min    7.900000e+01  1.000100e+04       1.000000e+00
25%    1.191700e+04  1.231100e+04       1.000000e+00
50%    2.747900e+04  1.334600e+04       2.000000e+00
75%    4.885800e+04  1.491600e+04       4.000000e+00
max    6.553500e+04  1.871500e+04       9.770000e+02
```

In [9]:
```
# Generating concise summary of the dataset
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10857234 entries, 0 to 10857233
Data columns (total 6 columns):
 #   Column             Dtype
---  ------             -----
 0   TripID             int64
 1   RouteID            object
 2   StopID             int64
 3   StopName           object
 4   WeekBeginning      object
 5   NumberOfBoardings  int64
dtypes: int64(3), object(3)
memory usage: 579.8+ MB
None
```

In [11]:
```
# Shape of the dataset
print(data.shape)
```

```
(10857234, 6)
```

In [12]:
```
# Displaying first few rows after preprocessing
data.head()
```

Out[12]:

	TripID	RouteID	StopID	StopName	WeekBeginning	NumberOfBoardings
0	23631	100	14156	181 Cross Rd	2013-06-30 00:00:00	1
1	23631	100	14144	177 Cross Rd	2013-06-30 00:00:00	1
2	23632	100	14132	175 Cross Rd	2013-06-30 00:00:00	1
3	23633	100	12266	Zone A Arndale Interchange	2013-06-30 00:00:00	2
4	23633	100	14147	178 Cross Rd	2013-06-30 00:00:00	1

In [ ]:

PUBLIC TRANSPORTATION ANALYSIS
PHASE 4

External Features

Some Important external data fields calculation

- **IsHoliday** Number of public holidays within that week
- **DistanceFromCentre** Distance measure from the city centre

For Calculating Distance between centre with other bus stops by using Longitude and Latitude we have used the Haversine formula

In [8]:

```
from math import sin, cos, sqrt, atan2, radiansdef calc_dist(lat1,lon1):

    ## approximate radius of earth in km

    R = 6373.0

    dlon = radians(138.604801) - radians(lon1)

    dlat = radians(-34.921247) - radians(lat1)

    a = sin(dlat / 2)**2 + cos(radians(lat1)) * cos(radians(-34.921247)) * sin(dl
on / 2)**2

    c = 2 * atan2(sqrt(a), sqrt(1 - a))

    return R * c
```

In [9]:

```
out_geo['dist_from_centre'] = out_geo[['latitude','longitude']].apply(lambda x:
 calc_dist(*x), axis=1)
```

In [10]:

```
##Fill the missing values with modeout_geo['type'].fillna('street_address',inpl
ace=True)out_geo['type'] = out_geo['type'].apply(lambda x: str(x).split(',')[-1])
```

In [11]:

```
out_geo['type'].unique()
```

Out[11]:

```
array(['street_address', 'transit_station', 'premise', 'political',

       'school', 'route', 'intersection', 'point_of_interest',
```

'subpremise', 'real_estate_agency', 'university', 'travel_agency',

'restaurant', 'supermarket', 'store', 'post_office'], dtype=object)

Adding the details regarding the Public holidays from June 2013 to June 2014

In [12]:

```
'''Holidays--2013-09-01,Father's Day2013-10-07,Labour day2013-12-25,Christmas day2013-12-26,Proclamation Day2014-01-01,New Year2014-01-27,Australia Day2014-03-10,March Public Holiday2014-04-18,Good Friday2014-04-19,Easter Saturday2014-04-21,Easter Monday2014-04-25,Anzac Day2014-06-09,Queen's Birthday'''
```

Out[12]:

"Holidays--\n2013-09-01,Father's Day\n2013-10-07,Labour day\n2013-12-25,Christmas day\n2013-12-26,Proclamation Day\n2014-01-01,New Year\n2014-01-27,Australia Day\n2014-03-10,March Public Holiday\n2014-04-18,Good Friday\n2014-04-19,Easter Saturday\n2014-04-21,Easter Monday\n2014-04-25,Anzac Day\n2014-06-09,Queen's Birthday"

In [13]:

```
def holiday_label (row):
    if row == datetime.date(2013, 9, 1) :
        return '1'
    if row == datetime.date(2013, 10, 6) :
        return '1'
    if row == datetime.date(2013, 12, 22) :
        return '2'
    if row == datetime.date(2013, 12, 29):
        return '1'
    if row  == datetime.date(2014, 1, 26):
        return '1'
    if row == datetime.date(2014, 3, 9):
        return '1'
    if row == datetime.date(2014, 4, 13) :
```

```
        return '2'
    if row == datetime.date(2014, 4, 20):
        return '2'
    if row == datetime.date(2014, 6, 8):
        return '1'
    return '0'
```

In [14]:

```
data['WeekBeginning'] = pd.to_datetime(data['WeekBeginning']).dt.date
```

In [15]:

```
data['holiday_label'] = data['WeekBeginning'].apply (lambda row: holiday_la
bel(row))
```

Data Aggregation

Combine the Geolocation,Routes and main input file to get final Output File.

In [16]:

```
data= pd.merge(data,out_geo,how='left',left_on = 'StopName',right_on = 'inp
ut_string')
```

In [17]:

```
data = pd.merge(data, route, how='left', left_on = 'RouteID', right_on = 'route
_id')
```

Columns to keep for further analysis

In [18]:

```
col = ['TripID', 'RouteID', 'StopID', 'StopName', 'WeekBeginning','NumberOf
Boardings','formatted_address',
    'latitude', 'longitude','postcode','type','route_desc','dist_from_centre','holid
ay_label']
```

In [19]:

```
data = data[col]
```

In [20]:

```
##saving the final datasetdata.to_csv('Weekly_Boarding.csv',index=False)
```

In [21]:

```
## getting the addresses for geolocation api.# Address data['StopName'].uniq
ue()# sub = pd.DataFrame({'Address': Address})# sub=sub.reindex(columns
=["Address"])# sub.to_csv('addr.csv')
```

Aggregate the Data According to Weeks and Stop names

- **NumberOfBoardings_sum** Number of Boardings within particular week for each Bus stop
- **NumberOfBoardings_count** Number of times data is recorded within week
- **NumberOfBoardings_max** Maximum number of boarding done at single time within week

In [22]:

```
# st_week_grp1 = pd.DataFrame(data.groupby(['StopName','WeekBeginning
','type']).agg({'NumberOfBoardings': ['sum', 'count']})).reset_index()grouped
= data.groupby(['StopName','WeekBeginning','type']).agg({'NumberOfBoardi
ngs': ['sum', 'count','max']})grouped.columns = ["_".join(x) for x in grouped.c
olumns.ravel()]
```

In [23]:

```
st_week_grp = pd.DataFrame(grouped).reset_index()st_week_grp.shapest_w
eek_grp.head()
```

Out[23]:

(207864, 6)

Out[23]:

	Stop Name	WeekBeginning	type	NumberOfBoardings_sum	NumberOfBoardings_count	NumberOfBoardings_max
0	1 Anzac Hwy	2013-06-30	street_address	1003	378	51
1	1 Anzac Hwy	2013-07-07	street_address	783	360	28
2	1 Anzac Hwy	2013-07-14	street_address	843	343	45
3	1 Anzac	2013-07-21	street_address	710	356	28

	Stop Name	WeekBeginning	type	NumberOfBoardings_sum	NumberOfBoardings_count	NumberOfBoardings_max
	Hwy					
4	1 Anzac Hwy	2013-07-28	street_address	898	379	41

Gathering only the Stop Name which having all 54 weeks of Data

In [24]:

```
st_week_grp1 = pd.DataFrame(st_week_grp.groupby('StopName')['WeekBeginning'].count()).reset_index()
```

In [25]:

```
aa=list(st_week_grp1[st_week_grp1['WeekBeginning'] == 54]['StopName'])
```

In [26]:

```
bb = st_week_grp[st_week_grp['StopName'].isin(aa)]
```

In [27]:

```
## save the aggregate databb.to_csv('st_week_grp.csv', index=False)
```

Data Exploration

Total Having 1 Year of Data from date 2013-06-30 till 2014-07-06 in a Weekly interval based.

Having Total of 4165 Stops in South Australian Metropolitan Area.

In [28]:

```
data.nunique()
```

Out[28]:

```
TripID              39282
RouteID               619
StopID               7397
StopName             4165
WeekBeginning          54
NumberOfBoardings     400
formatted_address    3242
latitude             3029
longitude            3008
postcode              207
type                   16
route_desc            440
dist_from_centre     3033
holiday_label           3
dtype: int64
```

In [29]:

```
data.shapedata.columnsdata.head(3)
```

Out[29]:

```
(10857234, 14)
```

Out[29]:

```
Index(['TripID', 'RouteID', 'StopID', 'StopName', 'WeekBeginning',
       'NumberOfBoardings', 'formatted_address', 'latitude', 'longitude',
```

'postcode', 'type', 'route_desc', 'dist_from_centre', 'holiday_label'],

dtype='object')

Out[29]:

	TripID	RouteID	StopID	StopName	WeekBeginning	NumberOfBoardings	formatted_address	latitude	longitude	postcode	type	route_desc	dist_from_centre	holiday_label
0	236631	100	141156	181 Cross Rd	2013-06-30	1	181 Cross Rd, Westbourne Park SA 5041, Australia	-34.966656	138.595948	5041	street_address	via Woodville Road, Holbrooks Road, Marion Roa...	5.180961	0
1	236631	100	141144	177 Cross Rd	2013-06-30	1	177 Cross Rd, Westbourne Park SA 5041, Australia	-34.966607	138.595948	5041	street_address	via Woodville Road, Holbrooks Road, Marion Roa...	5.172525	0
2	2	10	1	17	2013	1	175	-	138	50	stre	via	5.18	0

	TripID	RouteID	StopID	StopName	WeekBeginning	NumberOfBoardings	formatted_address	latitude	longitude	postcode	type	route_desc	dist_from_centre	holiday_label
	3632	0	4132	5 Cross Rd	-06-30		Cross Rd, Westbourne Park SA 5041, Australia	34.9667758	.592715	41	et_address	Woodville Road, Holbrooks Road, Marion Roa...	0709	

In [30]:

```
data.isnull().sum()
```

Out[30]:

```
TripID                  0
RouteID                 0
StopID                  0
StopName                0
WeekBeginning           0
NumberOfBoardings       0
formatted_address    3506
latitude                0
longitude               0
postcode           425081
type                    0
route_desc         2106618
```

dist_from_centre 0

holiday_label 0

dtype: int64

How Many different type of Unique Data in the dataset

In [31]:

```
data['WeekBeginning'].unique()
```

Out[31]:

```
array([datetime.date(2013, 6, 30), datetime.date(2013, 7, 7),
       datetime.date(2013, 7, 14), datetime.date(2013, 7, 21),
       datetime.date(2013, 7, 28), datetime.date(2013, 8, 4),
       datetime.date(2013, 8, 11), datetime.date(2013, 8, 18),
       datetime.date(2013, 8, 25), datetime.date(2013, 9, 1),
       datetime.date(2013, 9, 8), datetime.date(2013, 9, 15),
       datetime.date(2013, 9, 22), datetime.date(2013, 9, 29),
       datetime.date(2013, 10, 6), datetime.date(2013, 10, 13),
       datetime.date(2013, 10, 20), datetime.date(2013, 10, 27),
       datetime.date(2013, 11, 3), datetime.date(2013, 11, 10),
       datetime.date(2013, 11, 17), datetime.date(2013, 11, 24),
       datetime.date(2013, 12, 1), datetime.date(2013, 12, 8),
       datetime.date(2013, 12, 15), datetime.date(2013, 12, 22),
       datetime.date(2013, 12, 29), datetime.date(2014, 1, 5),
       datetime.date(2014, 1, 12), datetime.date(2014, 1, 19),
       datetime.date(2014, 1, 26), datetime.date(2014, 2, 2),
       datetime.date(2014, 2, 9), datetime.date(2014, 2, 16),
       datetime.date(2014, 2, 23), datetime.date(2014, 3, 2),
       datetime.date(2014, 3, 9), datetime.date(2014, 3, 16),
       datetime.date(2014, 3, 23), datetime.date(2014, 3, 30),
       datetime.date(2014, 4, 6), datetime.date(2014, 4, 13),
```

datetime.date(2014, 4, 20), datetime.date(2014, 4, 27),

datetime.date(2014, 5, 4), datetime.date(2014, 5, 11),

datetime.date(2014, 5, 18), datetime.date(2014, 5, 25),

datetime.date(2014, 6, 1), datetime.date(2014, 6, 8),

datetime.date(2014, 6, 15), datetime.date(2014, 6, 22),

datetime.date(2014, 6, 29), datetime.date(2014, 7, 6)],

dtype=object)

Data Visualization

In [32]:

```
##can assign the each chart to one axes at a timefig,axrr=plt.subplots(3,2,figsize=(18,18))

data['NumberOfBoardings'].value_counts().sort_index().head(20).plot.bar(ax=axrr[0][0])data['WeekBeginning'].value_counts().plot.area(ax=axrr[0][1])data['RouteID'].value_counts().head(20).plot.bar(ax=axrr[1][0])data['RouteID'].value_counts().tail(20).plot.bar(ax=axrr[1][1])data['type'].value_counts().head(5).plot.bar(ax=axrr[2][0])data['type'].value_counts().tail(10).plot.bar(ax=axrr[2][1])
```

Out[32]:

<matplotlib.axes._subplots.AxesSubplot at 0x7f1726f9e860>

Out[32]:

<matplotlib.axes._subplots.AxesSubplot at 0x7f1615adbb38>

Out[32]:

<matplotlib.axes._subplots.AxesSubplot at 0x7f1645050f28>

Out[32]:

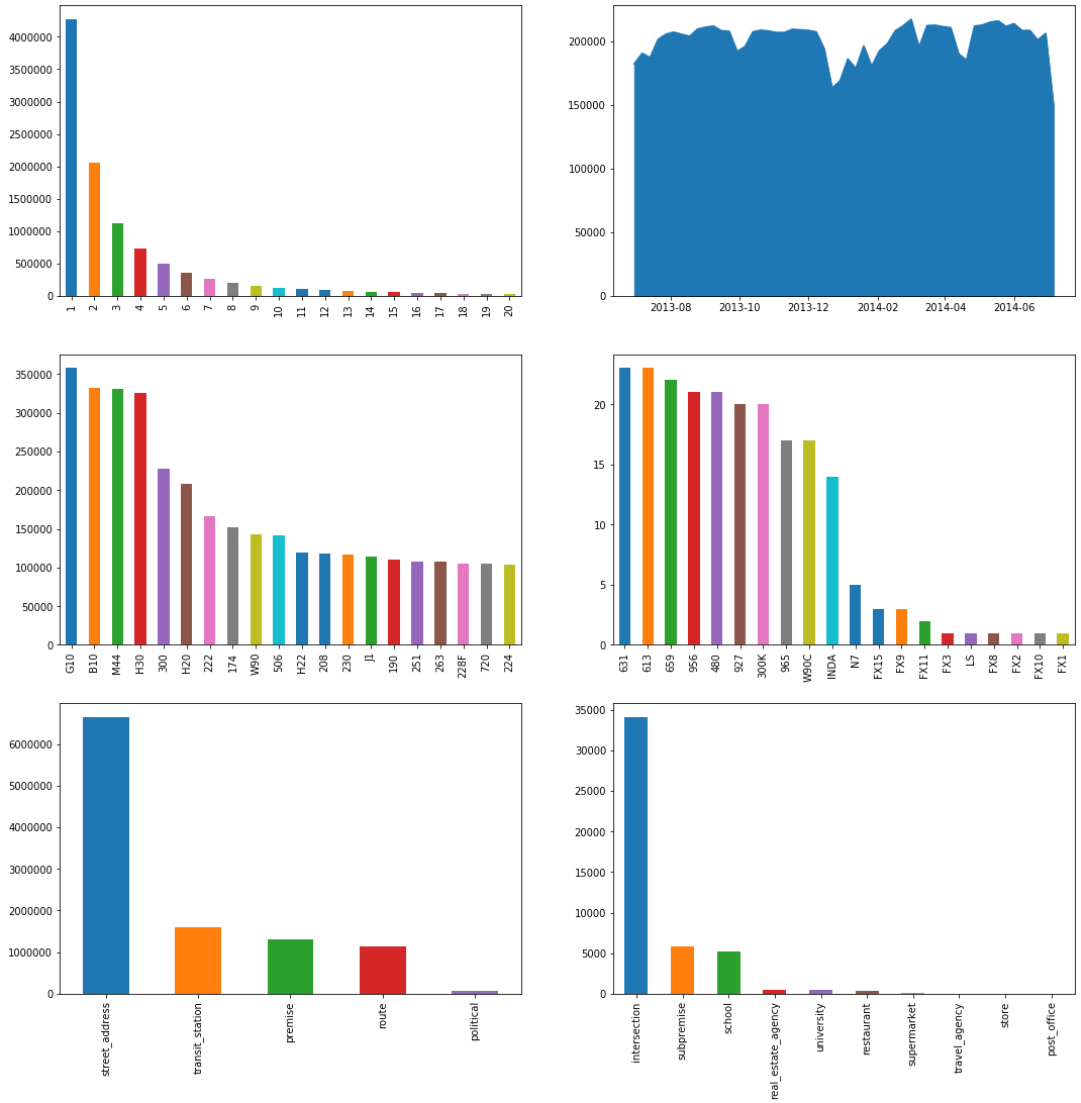
<matplotlib.axes._subplots.AxesSubplot at 0x7f171ef36588>

Out[32]:

<matplotlib.axes._subplots.AxesSubplot at 0x7f171ef5dc50>

Out[32]:

<matplotlib.axes._subplots.AxesSubplot at 0x7f171ef0d2e8>



Inferences:

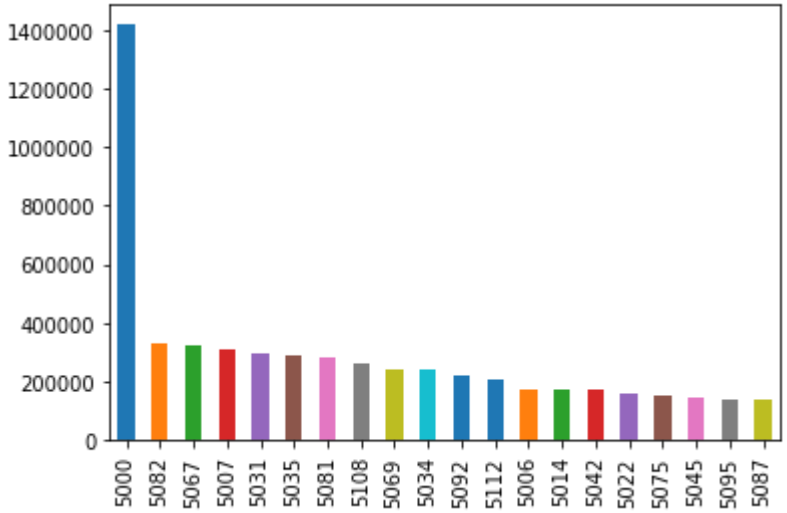
- More than 40 lakhs times only single person board from the bus stop.
- There are average of 1.8 lakhs people travel every week by bus in adelaide metropolitan area.
- G10,B10,M44,H30 are the most busiest routes in the city while FX8,FX3,FX10,FX1,FX2 are the least.
- Most of the Bus stops are Street_Address Type while there are very few which are store or post office.

In [33]:

```
data['postcode'].value_counts().head(20).plot.bar()
```

Out[33]:

<matplotlib.axes._subplots.AxesSubplot at 0x7f171b4c0c50>



In [34]:

```
# data['dist_from_centre'].nunique()bb_grp = data.groupby(['dist_from_centre']).agg({'NumberOfBoardings': ['sum']}).reset_index()bb_grp.columns = bb_grp.columns.get_level_values(0)bb_grp.head()bb_grp.columns
```

Out[34]:

	dist_from_centre	NumberOfBoardings
0	0.000018	1892443
1	0.131368	167535
2	0.309089	356518
3	0.314937	1484824
4	0.326005	120061

Out[34]:

Index(['dist_from_centre', 'NumberOfBoardings'], dtype='object')

In [35]:

```
trace0 = go.Scatter(

   x = bb_grp['dist_from_centre'],

   y = bb_grp['NumberOfBoardings'],mode = 'lines+markers',name = 'X2 King William St')

data1 = [trace0]layout = dict(title = 'Distance Vs Number of boarding',

          xaxis = dict(title = 'Distance from centre'),

          yaxis = dict(title = 'Number of Boardings'))fig = dict(data=data1, layout=layout)iplot(fig)
```

05k10k15k00.5M1M1.5M2MExport to plot.ly »Distance Vs Number of boardingDistance from centreNumber of Boardings

Inferences:

- As we move away from centre the number of Boarding decreases
- There are cluster of bus stops near to the main Adelaide city as oppose to outside.so that's why most of boardings are near to center

Using Bokeh

Plot the Bus stop on the Google Map using the latitude and longitude of the bus stop address

In [36]:

```
lat = out_geo['latitude'].tolist()long = out_geo['longitude'].tolist()nam = out_g
eo['input_string'].tolist()
```

In [37]:

```
map_options = GMapOptions(lat=-34.96, lng=138.592, map_type="roadmap
", zoom=9)key = open('../input/geolockey/api_key.txt').read()p = gmap(key,
map_options, title="Adelaide South Australia")source = ColumnDataSource
(data=dict(lat=lat,lon=long,nam=nam))

p.circle(x="lon", y="lat", size=5, fill_color="blue", fill_alpha=0.8, source=so
urce)TOOLTIPS = [("Place", "@nam")]p.add_tools( HoverTool(tooltips=TO
OLTIPS))output_notebook()show(p)
```

Out[37]:

Inferences:

- It has Geospatial coverage Area from Lat: 34.3862 to -35.3655 and Lon: 138.4126 to 139.1089. Which is Total 152 KM long Area from Daniel Road to Mosquito Creek Road on one side and Total 162 KM Stretch from Truro to Myponga Beach on the other side.
- There are cluster of bus stops near to the main Adelaide city as oppose to outside.

```
source_6 = bb[bb['StopName'] == '57A Hancock Rd'].reset_index(drop = Tru
e)source_7 = bb[bb['StopName'] == '37 Muriel Dr'].reset_index(drop = True)
source_8 = bb[bb['StopName'] == '18B Springbank Rd'].reset_index(drop = T
rue)source_9 = bb[bb['StopName'] == '27E Sir Ross Smith Av'].reset_index(d
```

```
rop = True)source_10 = bb[bb['StopName'] == '46A Baldock Rd'].reset_index
(drop = True)
```

In [42]:

```
trace0 = go.Scatter(

    x = source_6['WeekBeginning'],

    y = source_6['NumberOfBoardings_sum'],mode = 'lines+markers',name = '
57A Hancock Rd')trace1 = go.Scatter(

    x = source_7['WeekBeginning'],

    y = source_7['NumberOfBoardings_sum'],mode = 'lines+markers',name = '
37 Muriel Dr')trace2 = go.Scatter(

    x = source_8['WeekBeginning'],

    y = source_8['NumberOfBoardings_sum'],mode = 'lines+markers',name = '
18B Springbank Rd')trace3 = go.Scatter(

    x = source_9['WeekBeginning'],

    y = source_9['NumberOfBoardings_sum'],mode = 'lines+markers',name = '
27E Sir Ross Smith Av')trace4 = go.Scatter(

    x = source_10['WeekBeginning'],

    y = source_10['NumberOfBoardings_sum'],mode = 'lines+markers',name =
 '46A Baldock Rd')

data = [trace0,trace1,trace2,trace3,trace4]layout = dict(title = 'Weekly Boardi
ng Total',

              xaxis = dict(title = 'Week Number'),

              yaxis = dict(title = 'Number of Boardings'),

              shapes = [{# Holidays Record: 2013-09-01'type': 'line','x0': '2013-09-
01','y0': 0,'x1': '2013-09-02','y1': 80,'line': {

        'color': 'rgb(55, 128, 191)','width': 1,'dash': 'dashdot'},},

              {# 2013-10-07'type': 'line','x0': '2013-10-07','y0': 0,'x1': '2013-10-07','
y1': 80,'line': {

        'color': 'rgb(55, 128, 191)','width': 1,'dash': 'dashdot'},},
```

```
            {# 2013-12-25'type': 'line','x0': '2013-12-25','y0': 0,'x1': '2013-12-26','
y1': 80,'line': {

        'color': 'rgb(55, 128, 191)','width': 3,'dash': 'dashdot'},},

            {# 2014-01-27'type': 'line','x0': '2014-01-27','y0': 0,'x1': '2014-01-28','
y1': 80,'line': {

        'color': 'rgb(55, 128, 191)','width': 1,'dash': 'dashdot'},},

            {# 2014-03-10'type': 'line','x0': '2014-03-10','y0': 0,'x1': '2014-03-11','
y1': 80,'line': {

        'color': 'rgb(55, 128, 191)','width': 1,'dash': 'dashdot'},},

            {# 2014-04-18'type': 'line','x0': '2014-04-18','y0': 0,'x1': '2014-04-19','
y1': 80,'line': {

        'color': 'rgb(55, 128, 191)','width': 3,'dash': 'dashdot'},},

            {# 2014-06-09'type': 'line','x0': '2014-06-09','y0': 0,'x1': '2014-06-10','
y1': 80,'line': {

        'color': 'rgb(55, 128, 191)','width': 1,'dash': 'dashdot'},},])fig = dict(data=
data, layout=layout)iplot(fig)
```

Jul 2013Sep 2013Nov 2013Jan 2014Mar 2014May 2014Jul 2014020406080100120Export to plot.ly »57A Hancock Rd37 Muriel Dr18B Springbank Rd27E Sir Ross Smith Av46A Baldock RdWeekly Boarding TotalWeek NumberNumber of Boardings

Inferences:

- Same decreasing affect of Holidays on number of people travelling through bus can be seen in other city bus stops also.
- The width of vertical blue line shows the number of holidays come within that week period.
- Two thickest blue lines shows Christmas and New year period while other one was easter & Good friday period.on both the occassion number of public holidays within week period was 3.