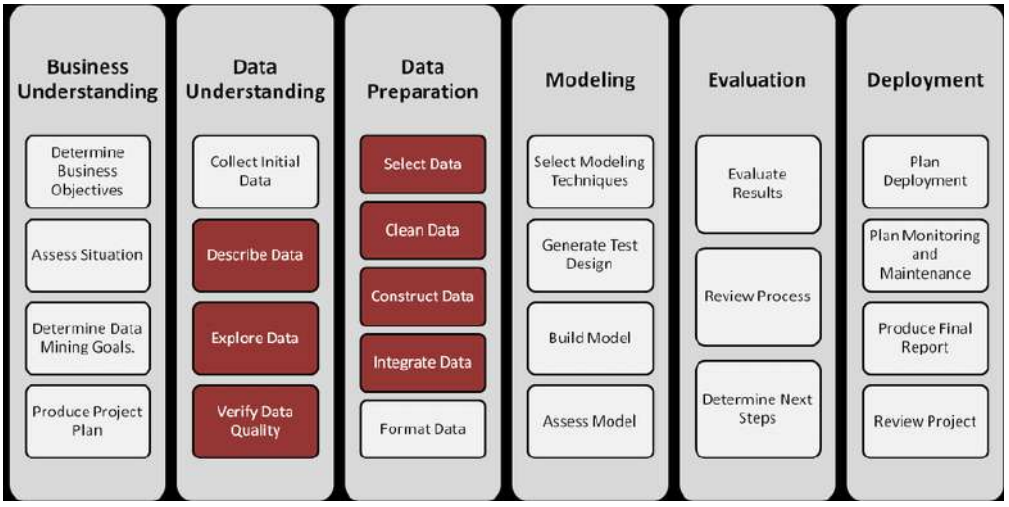
Tugas Mandiri

Tugas mandiri ini digunakan pada kegiatan Kursus Data Science yang merupakan pembekalan bagi mahasiswa Universitas Gunadarma untuk Skema Associate Data Scientist

Pertemuan 1 - Semester 8

Daftar Isi :



1. Definisi Menelaah Data (*Data Understanding*)

Secara garis besar, *data understanding* adalah tahapan setelah setelah problem bisnis telah didefinisikan sebagai hasil tahapan *business understanding.

Tahapan *data understanding* dilakukab untuk memeriksa data sehingga masalah pada data yang didapatkan dapat terindentifikasi.

- Tujuan: mendapatkan gambaran utuh atas data.
- Dilanjutkan ke persiapan data (data preparation), jika pemahaman awal data cukup atau kembali ke business understanding jika definisi permasalahan bisnis harus direvisi.

1.1 Mengapa Perlu Data Understanding?

- Data merupakan bahan mentah solusi AI.
- Data dari masing-masing sumber belum tentu dapat langsung dipakai karena:
- Maksud dan tujuan data berbeda-beda
- Keadaan asal terpisah-pisah atau justru terintegrasi secara ketat.
- Tingkat kekayaan (richness) berbeda-beda
- Tingkat keandalan (reliability) berbeda-beda
- Data understanding memberikan gambaran awal tentang:
    - Kekuatan data
    - Kekurangan dan batasan penggunaan data
    - Tingkat kesesuaian data dengan masalah bisnis yang akan dipecahkan
    - Ketersediaan data (terbuka/tertutup, biaya akses, dsb.)

2. Sumber Data

Sumber data terbagi menjadi dua yaitu sumber data *internal* dan *external* :

1. Sumber Data *Internal*
    - Spreadsheets (Excel, CSV, JSON, etc)
    - Databases : Can be queried via SQL, etc
    - Text documents
    - Multimedia documents (audio & video)
2. Sumber Data *External*
    - Open data repositories
    - Public domain web pages

3. Tipe Data berdasarkan Susunan Data: Structured and Unstructured Data

---

	Data terstruktur (structured data)	Data tak terstruktur (unstructured data)
Sifat	• Model data terdefinisikan sebelumnya • Format butir data (biasanya) teks. • Antar butir data dibedakan dengan jelas. • Ekstraksi/kueri langsung cukup mudah.	• Model data tidak terdefinisikan sebelumnya • Format butir data (biasanya) teks, citra, suara, video, dan format lainnya. • Antar butir data tidak cukup jelas terbedakan karena ketidakteraturan dan ambiguitas. • Ekstraksi/kueri langsung cukup sulit.
Contoh	Data tabular, data berorientasi objek, *time series*	Data teks dalam dokumen teks bebas, data audio, data video.

Data semi-terstruktur (semi-structured data): Data terstruktur yang tidak mengikuti model struktur tabular yang seperti pada basis data relasional, namun tetap mengandung tags atau penanda lainnya yang dapat memisahkan elemen-elemen semantik pada data serta mengatur hierarki antara butir-butir datanya.

4. Tipe Data berdasarkan Sifat, Cara Pengumpulan, Waktu

4.1 Tipe Data Berdasarkan Sifatnya

- Data dikotomi, merupakan data yang bersifat pilah satu sama lain, misalnya suku, agama, jenis kelamin, pendidikan, dan lain sebagainya.
- Data diskrit, merupakan data yang proses pengumpulan datanya dijalankan dengan cara menghitung atau membilang. Seperti, jumlah anak, jumlah penduduk, jumlah kematian dan sebagainya.
- Data kontinum, merupakan data pengumpulan datanya didapatkan dengan cara mengukur dengan alat ukur yang memakai skala tertentu. Seperti misalnya, Suhu, berat, bakat, kecerdasan, dan lainnya.

4.2 Tipe Data Tipe Data Berdasarkan Cara Pengumpulan

- Data Primer, merupakan data yang didapatkan dari sumber pertama, atau dapat dikatakan pengumpulannya dilakukan sendiri oleh si peneliti secara langsung, seperti hasil wawancara dan hasil pengisian kuesioner (angket).
- Data Sekunder, merupakan data yang didapatkan dari sumber kedua. Menurut Purwanto (2007), data sekunder yaitu data yang dikumpulkan oleh orang atau lembaga lain. Data sekunder adalah data yang digunakan atau diterbitkan oleh organisasi yang bukan pengolahnya (Soeratno dan Arsyad (2003;76).

4.3 Tipe Data Tipe Data Berdasarkan Waktu

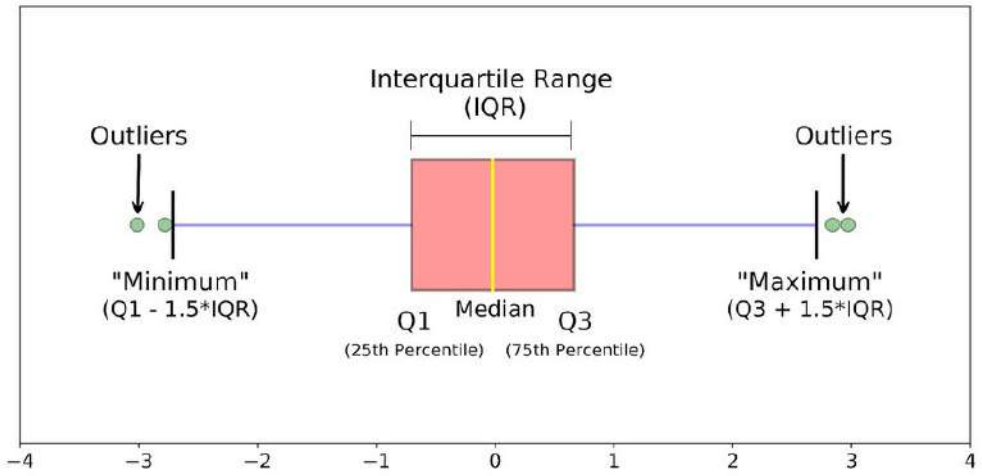
- Data Cross Section, merupakan data yang menunjukkan titik waktu tertentu. Contohnya laporan keuangan per 31 Desember 2020, data pelanggan PT. Data Indah bulan mei 2004, dan lain sebagainya.
- Data Sekunder, merupakan data yang datanya menggambarkan sesuatu dari waktu ke waktu atau periode secara historis. Contoh data time series adalah data perkembangan nilai tukar dollar amerika terhadap rupiah tahun 2016 – 2020.

5. Metode Pengambilan Data

Pengambilan data terbagi ke dalam beberapa jenis yaitu :

- Pengambilan data secara manual.
- Pengambilan data melalui API
- Pengambilan data melalui web scraping
- Pengambilan data melalui akses langsung ke basis data relasional yang ada.

6. Pencilan (Tukey's fences)



Tukey's fences memakai rentang antarkuartil (interquartile range) -> IQR = Q3-Q1

- Xi adalah pencilan jika Xi < Q1 - 1.5(IQR) atau Xi > Q3 + 1.5(IQR)
- Xi adalah pencilan ekstrim jika Xi < Q1 - 3(IQR) atau Xi > Q3 + 3(IQR)

Tugas Mandiri Menelaah Data

Import Library

In [1]:
```
# 1. Import Library Pandas
import pandas as pd
```

In [2]:
```
# 2. Memuat Data
df = pd.read_csv('delay_maskapai.csv')
```

Menelaah Data dengan Metode Statistik

In [3]:
```
# 3. Menampilkan 5 baris pertama dari dataset
print(df.head(5))
```

```
   Flight   Time  Length Airline AirportFrom AirportTo  DayOfWeek  Class
0   320.0  870.0   180.0      CO          IAH       PHX          5      0
1  3045.0  365.0    66.0      OO          DSM       MKE          2      0
2  1560.0  740.0   257.0      DL          SEA       CVG          7      0
3  1156.0  595.0   235.0      WN          SNA       MDW          7      1
4  1873.0  530.0   133.0      CO          IAH       MCO          6      1
```

In [4]:
```
# 4. Menampilkan 10 baris pertama dari dataset
(df.head(10))
```

Out[4]:

	Flight	Time	Length	Airline	AirportFrom	AirportTo	DayOfWeek	Class
0	320.0	870.0	180.0	CO	IAH	PHX	5	0
1	3045.0	365.0	66.0	OO	DSM	MKE	2	0
2	1560.0	740.0	257.0	DL	SEA	CVG	7	0
3	1156.0	595.0	235.0	WN	SNA	MDW	7	1
4	1873.0	530.0	133.0	CO	IAH	MCO	6	1
5	3641.0	1045.0	160.0	MQ	MIA	CMH	1	0
6	138.0	1160.0	185.0	WN	BDL	MCO	7	0
7	586.0	700.0	142.0	AS	PDX	SNA	2	1
8	5220.0	895.0	102.0	EV	AEX	ATL	2	0
9	5122.0	516.0	137.0	EV	DTW	MCI	7	0

In [5]:
```
# 5. enampilkan 5 baris terakhir dari dataset
df.tail(5)
```

Out[5]:

	Flight	Time	Length	Airline	AirportFrom	AirportTo	DayOfWeek	Class
10782	2821.0	610.0	50.0	MQ	DFW	SPS	2	0
10783	2618.0	839.0	118.0	XE	SDF	EWR	2	0
10784	1973.0	1160.0	124.0	DL	ATL	MIA	2	0
10785	528.0	1275.0	112.0	US	PHX	RNO	4	0
10786	2254.0	970.0	95.0	DL	ATL	CMH	4	1

In [6]:
```
# 6. Menampilkan 20 baris terakhir dari dataset
df.tail(20)
```

Out[6]:

	Flight	Time	Length	Airline	AirportFrom	AirportTo	DayOfWeek	Class
10767	4200.0	355.0	64.0	9E	TYS	CVG	2	0
10768	796.0	360.0	95.0	WN	IND	BWI	2	1
10769	6527.0	370.0	82.0	OH	DCA	JFK	3	1
10770	541.0	1120.0	130.0	WN	SJC	SEA	5	0
10771	6317.0	531.0	49.0	OO	SBA	LAX	2	0
10772	3486.0	450.0	245.0	WN	MDW	PHX	3	1
10773	897.0	1019.0	135.0	FL	LGA	IND	1	1
10774	1294.0	1075.0	140.0	AA	IAH	MIA	7	1
10775	6474.0	961.0	109.0	OO	LAX	ABQ	3	0
10776	6509.0	571.0	92.0	OO	TUS	LAX	3	0
10777	1679.0	505.0	57.0	CO	IAH	SAT	1	1
10778	508.0	480.0	325.0	WN	MCO	LAS	3	1
10779	4961.0	999.0	80.0	EV	FLO	ATL	6	0
10780	2297.0	560.0	65.0	XE	MKE	CLE	6	1
10781	1227.0	645.0	75.0	WN	RDU	PHL	2	0
10782	2821.0	610.0	50.0	MQ	DFW	SPS	2	0
10783	2618.0	839.0	118.0	XE	SDF	EWR	2	0
10784	1973.0	1160.0	124.0	DL	ATL	MIA	2	0
10785	528.0	1275.0	112.0	US	PHX	RNO	4	0
10786	2254.0	970.0	95.0	DL	ATL	CMH	4	1

In [7]:
```
# 7. Mengetahui jumlah baris dan kolom memanfaatkan fungsi shape
df.shape
```

Out[7]:
```
(10787, 8)
```

In [8]:
```
# 8. Mengetahui jumlah baris dan kolom memanfaatkan fungsi shape
print("Jumlah Baris : ", df.shape[0])
print("Jumlah Kolom : ", df.shape[1])
```

```
Jumlah Baris :  10787
Jumlah Kolom :  8
```

In [9]:
```
# 9. Mengetahui tipe data pada dataset
df.dtypes
```

Out[9]:
```
Flight         float64
Time           float64
Length         float64
Airline         object
AirportFrom     object
AirportTo       object
DayOfWeek        int64
Class            int64
dtype: object
```

In [10]:
```
# 10. Menampilkan statistik kolom
df.describe()
```

Out[10]:

	Flight	Time	Length	DayOfWeek	Class
count	10787.000000	10787.000000	10787.000000	10787.000000	10787.000000
mean	2435.506999	803.193566	133.346621	3.895430	0.447576
std	2071.935213	276.976027	71.598884	1.923294	0.497267
min	1.000000	15.000000	29.000000	1.000000	0.000000
25%	720.000000	567.000000	81.000000	2.000000	0.000000
50%	1798.000000	795.000000	116.000000	4.000000	0.000000
75%	3785.500000	1035.000000	164.000000	5.000000	1.000000
max	7812.000000	1439.000000	655.000000	7.000000	1.000000

In [11]:
```
# 11. Menampilkan statistik kolom baik yang bertipe numerik dan non numerik meng
df.describe(include='all')
```

Out[11]:

	Flight	Time	Length	Airline	AirportFrom	AirportTo	DayOfWee
count	10787.000000	10787.000000	10787.000000	10787	10787	10787	10787.00000
unique	NaN	NaN	NaN	18	270	265	Na
top	NaN	NaN	NaN	WN	ATL	ATL	Na
freq	NaN	NaN	NaN	1887	687	708	Na
mean	2435.506999	803.193566	133.346621	NaN	NaN	NaN	3.89543
std	2071.935213	276.976027	71.598884	NaN	NaN	NaN	1.92329
min	1.000000	15.000000	29.000000	NaN	NaN	NaN	1.00000
25%	720.000000	567.000000	81.000000	NaN	NaN	NaN	2.00000
50%	1798.000000	795.000000	116.000000	NaN	NaN	NaN	4.00000
75%	3785.500000	1035.000000	164.000000	NaN	NaN	NaN	5.00000
max	7812.000000	1439.000000	655.000000	NaN	NaN	NaN	7.00000

Mendeteksi Outliers

In [12]:
```
q1 = df.select_dtypes(exclude=['object']).quantile (0.25)
q3 = df.select_dtypes(exclude=['object']).quantile (0.75)
iqr = q3-q1
iqr
```

Out[12]:
```
Flight       3065.5
Time          468.0
Length         83.0
DayOfWeek       3.0
Class           1.0
dtype: float64
```

In [13]:
```
# 12. Melihat batas bawah/minimum dari Interquartile Range (IQR)
batas_bawah = q1 - (1.5 * iqr)
batas_bawah
```

Out[13]:
```
Flight      -3878.25
Time         -135.00
Length        -43.50
DayOfWeek      -2.50
Class          -1.50
dtype: float64
```

In [14]:
```
# 13. Melihat batas atas/maksimum dari Iterquartile Range (IQR)
batas_atas = q3 + (1.5 * iqr)
batas_atas
```

Out[14]:
```
Flight       8383.75
Time         1737.00
Length        288.50
DayOfWeek       9.50
Class           2.50
dtype: float64
```

Dari kode di atas dapat diketahui berapa nilai untuk batas atas dan batas bawah dari setiap kolom

In [15]:
```
# 14. Menampung kolom dengan tipe data selain object
df_ex_object = df.select_dtypes(exclude=['object'])
```

In [16]:
```
# 15. Filter outlier pada dataset dengan batas bawah dan atas
outlier_filter = ((df_ex_object < batas_bawah) | (df_ex_object > batas_atas))
outlier_filter
```

Out[16]:

	Flight	Time	Length	DayOfWeek	Class
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
...	...	...	...	...	...
10782	False	False	False	False	False
10783	False	False	False	False	False
10784	False	False	False	False	False
10785	False	False	False	False	False
10786	False	False	False	False	False

10787 rows × 5 columns

In [17]:
```
## 16. Menghitung Jumlah Outlier Pada Data
for col in outlier_filter.columns :

    # Exclude kolom dengan tipe data object
    if df[col].dtype != object :
        print('Nama Kolom:',col)
        print(outlier_filter[col].value_counts())
        print('--------------------')
```

```
Nama Kolom: Flight
Flight
False    10787
Name: count, dtype: int64
--------------------
Nama Kolom: Time
Time
False    10787
Name: count, dtype: int64
--------------------
Nama Kolom: Length
Length
False    10249
True       538
Name: count, dtype: int64
--------------------
Nama Kolom: DayOfWeek
DayOfWeek
False    10787
Name: count, dtype: int64
--------------------
Nama Kolom: Class
Class
False    10787
Name: count, dtype: int64
--------------------
```

In [18]:
```
#Persentase Outliers
length_outliers_percentage = len(outlier_filter[outlier_filter.Length==True]) /
length_outliers_percentage
```

Out[18]:
```
4.987484935570595
```

In [19]:
```
# 18. Analisa standar deviasi panjang penerbangan masing-masing maskapai dengan
df.groupby('Airline')['Length'].std()
```

Out[19]:
```
Airline
9E      25.137381
AA      79.964064
AS      87.750062
B6      83.566714
CO      81.894476
DL      73.829095
EV      27.749909
F9      45.220751
FL      43.172867
HA     108.017397
MQ      36.596474
OH      32.884057
OO      41.957837
UA      93.667622
US      77.367830
WN      60.583769
XE      37.130555
YV      39.197155
Name: Length, dtype: float64
```

In [20]:
```
# 19. Analisa jumlah delay masing-masing maskapai dengan Groupby
Airline_Delay_Percentage = df.groupby(['Airline','Class']).count().unstack('Clas
Airline_Delay_Percentage
```

Out[20]:

Class	0	1
Airline		
9E	264	163
AA	569	375
AS	149	77
B6	176	151
CO	183	253
DL	659	531
EV	370	239
F9	76	50
FL	293	134
HA	82	36
MQ	461	248
OH	181	74
OO	531	445
UA	356	154
US	450	250
WN	547	1340
XE	408	229
YV	204	79

In [21]:
```
# 20. Analisa jumlah delay dari masing-masing hari dengan Groupby
for i in range(Airline_Delay_Percentage.shape[0]):
    numOfFlights = Airline_Delay_Percentage.iloc[i,0]+Airline_Delay_Percentage.i
    for j in [0,1]:
        Airline_Delay_Percentage.iloc[i,j] = Airline_Delay_Percentage.iloc[i,j]/
Airline_Delay_Percentage.sort_values(by=Airline_Delay_Percentage.columns[1],asce
```

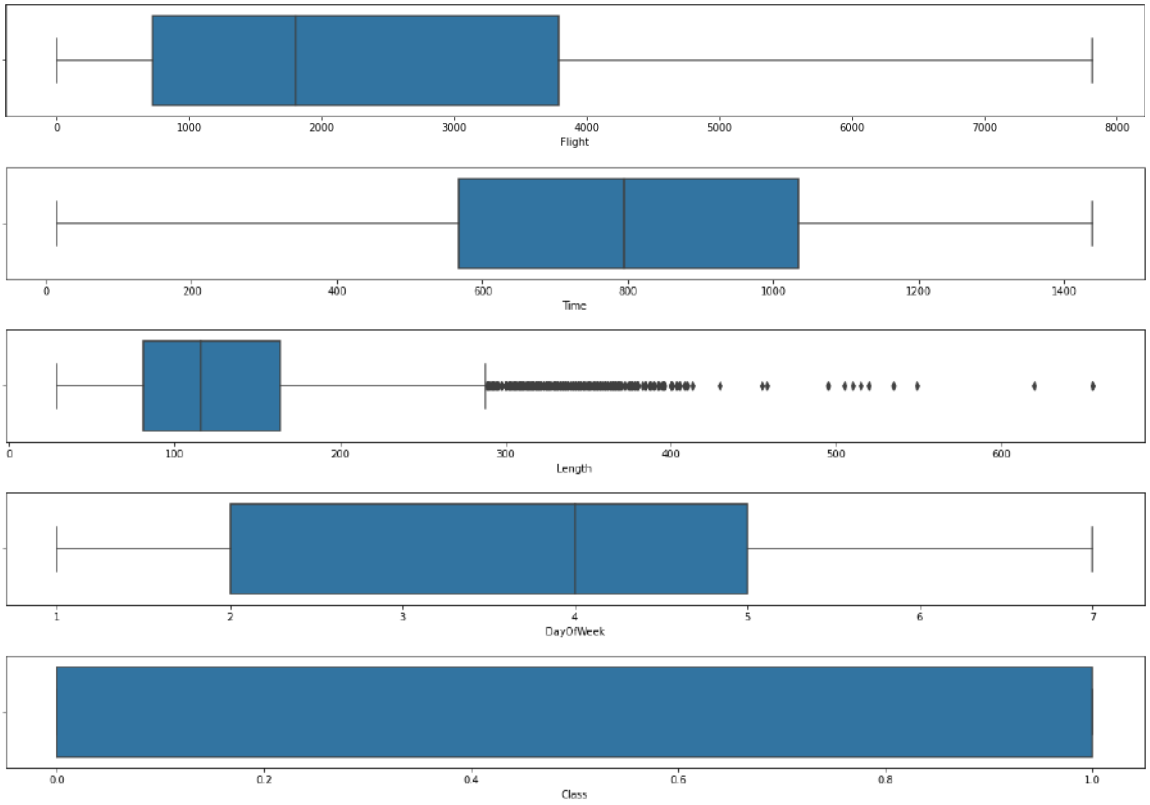
Out[21]:

Class	0	1
Airline		
WN	0.289878	0.710122
CO	0.419725	0.580275
B6	0.538226	0.461774
OO	0.544057	0.455943
DL	0.553782	0.446218
AA	0.602754	0.397246
F9	0.603175	0.396825
EV	0.607553	0.392447
9E	0.618267	0.381733
XE	0.640502	0.359498
US	0.642857	0.357143
MQ	0.650212	0.349788
AS	0.659292	0.340708
FL	0.686183	0.313817
HA	0.694915	0.305085
UA	0.698039	0.301961
OH	0.709804	0.290196
YV	0.720848	0.279152

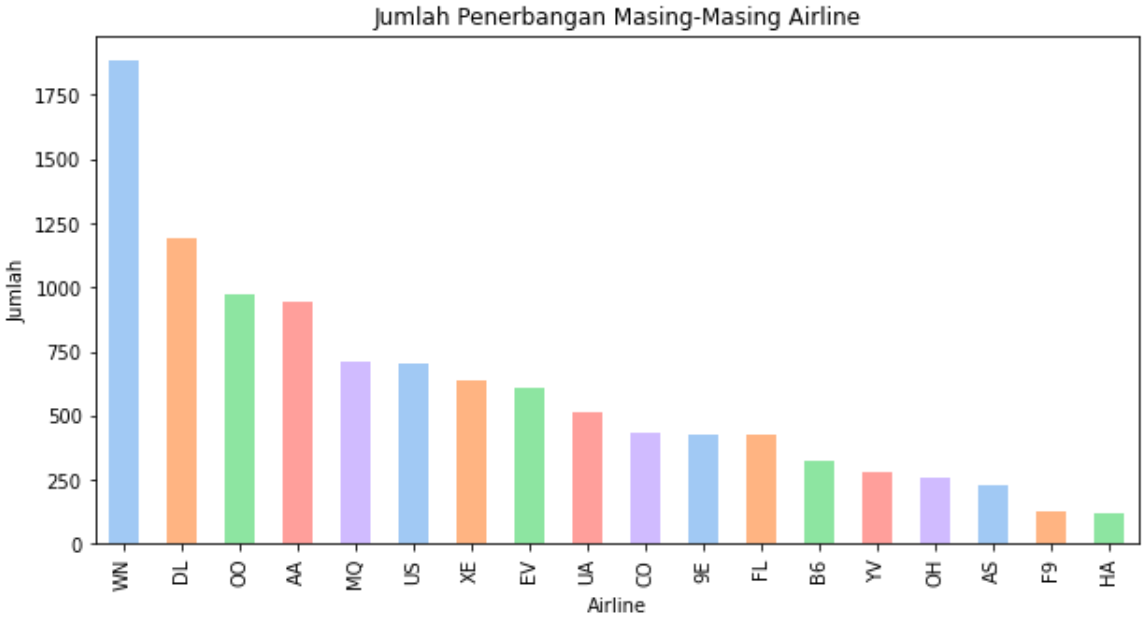
Menelaah Data dengan Metode Visualisasi

In [22]:
```
# Import Library

# 21. Library Numpy
import numpy as np
# 22. Pyplot
import matplotlib.pyplot as plt
# 23. Seaborn
import seaborn as sns
```

In [23]:
```
# Mencari Outliers - Boxplot

# 24. Exclude tipe data object
df_outlier = df.select_dtypes(exclude=['object'])
for column in df_outlier:
        plt.figure(figsize=(20,2))
        sns.boxplot(data=df_outlier, x=column)
```



In [24]:
```
# Analisa jumlah delay masing-masing maskapai

# 25. Menjumlahkan nilai dalam kolom 'Class' untuk setiap kelompok maskapai pene
Airline_Flight_Count = df.groupby('Airline').count()['Time'].sort_values(ascendi

plt.figure(figsize=(10,5))

myColors = sns.color_palette('pastel')[0:5]
Airline_Flight_Count.plot(kind='bar',color=myColors)
plt.ylabel('Jumlah')
plt.title('Jumlah Penerbangan Masing-Masing Airline')
plt.show()
```



In [25]:
```
# 26. Analisa Hari dengan delay - Groupby
delay_per_day = df.groupby(['DayOfWeek', 'Class'])['Time'].count().unstack('Clas
delay_per_day
```

Out[25]:

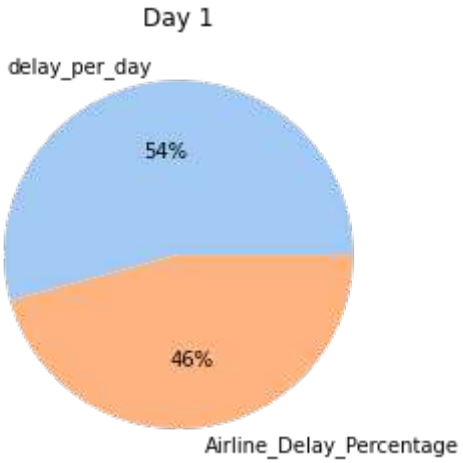
Class	0	1
DayOfWeek		
1	813	686
2	815	661
3	954	859
4	970	834
5	953	701
6	705	438
7	749	649

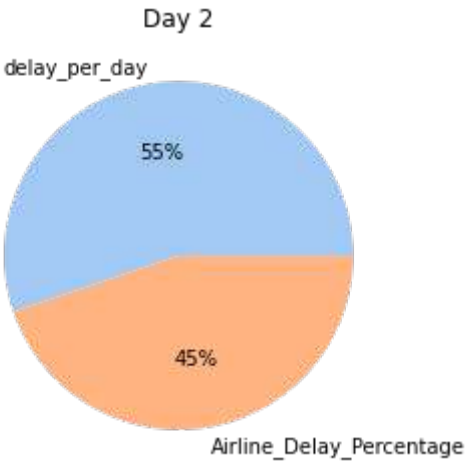
In [26]:
```
# 27. Membuat for loops untuk menghitung persentase delay per hari
for i in range(delay_per_day.shape[0]):
    numOfFlights = delay_per_day.iloc[i,0] + delay_per_day.iloc[i,1]
    for j in [0,1]:
        delay_per_day.iloc[i,j] = delay_per_day.iloc[i,j]/numOfFlights

delay_per_day.sort_values(by=delay_per_day.columns[1],ascending=False)
```

Out[26]:

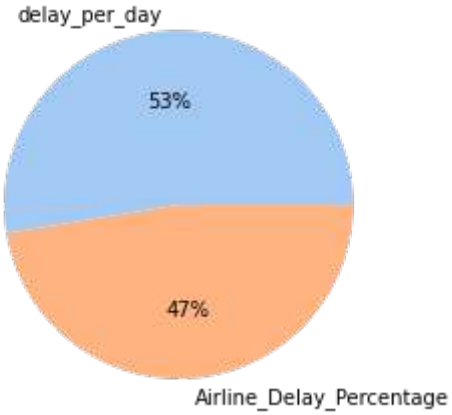
Class	0	1
DayOfWeek		
3	0.526200	0.473800
7	0.535765	0.464235
4	0.537694	0.462306
1	0.542362	0.457638
2	0.552168	0.447832
5	0.576179	0.423821
6	0.616798	0.383202

In [27]:
```
# 28. Label Pie Chart
pieChartLabels = ['delay_per_day','Airline_Delay_Percentage']

#Menentukan warna pale
myColors = sns.color_palette('pastel')

#Menampilkan pie chart untuk 1 minggu
for i in range(1,8):
    b = delay_per_day.iloc[i-1,:]
    #create pie chart
    plt.pie(b, labels = pieChartLabels, colors = myColors, autopct='%.0f%%')
    plt.title('Day ' + str(i))
    plt.show()
```

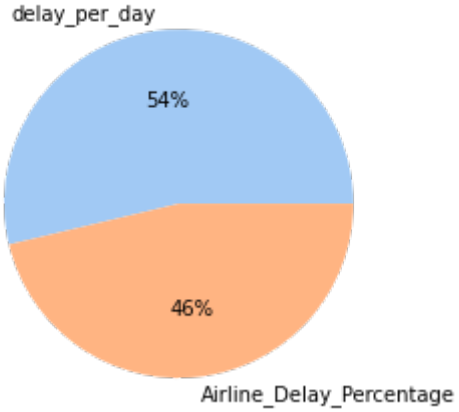




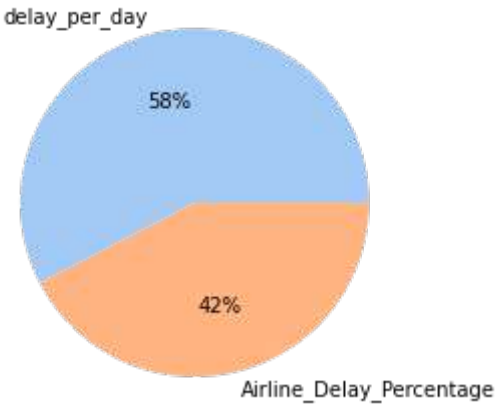
Day 3



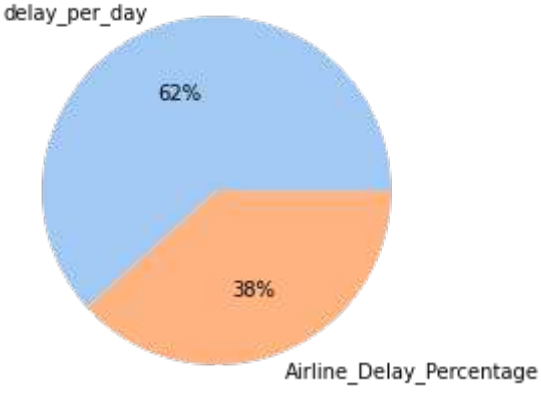
Day 4

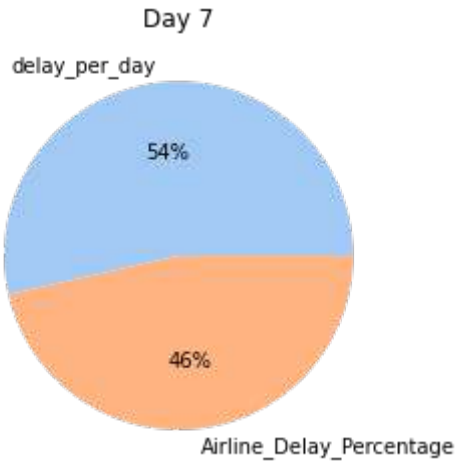


Day 5



Day 6





In [28]:
```
# 29. Melihat korelasi antara kolom
df.select_dtypes(exclude=['object']).corr()
```

Out[28]:

	Flight	Time	Length	DayOfWeek	Class
Flight	1.000000	-0.003814	-0.339813	0.011387	-0.046805
Time	-0.003814	1.000000	-0.023854	0.001654	0.133890
Length	-0.339813	-0.023854	1.000000	0.002270	0.041028
DayOfWeek	0.011387	0.001654	0.002270	1.000000	-0.020370
Class	-0.046805	0.133890	0.041028	-0.020370	1.000000

In [29]:
```
## 30. Melihat korelasi dengan Seaborn heatmap
sns.heatmap(df.select_dtypes(exclude=['object']).corr())
```

Out[29]:
```
<Axes: >
```