WITEL SEMARANG

Report Dashboard Selfi Fulfillment

Data preparation, Cleaning, and Visualization

Access & Operation (ASO) Unit

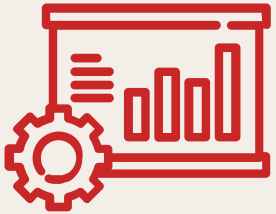
Problem

01

Terdapat kesulitan pada saat melakukan filtering data yang diinginkan untuk pengambilan keputusan.

02

Belum memiliki dashboard yang interaktif, visualisasi data masih berupa tabel biasa.



Solusi

Pengolahan Data

Menggunakan bahasa pemrograman python dalam mengolah data. Penggunaan ini bertujuan agar proses menjadi lebih efisien serta data menjadi lebih siap untuk divisualisasikan.

python™ Google Sheets

Visualisasi Data

Membuat dashboard visualisasi dengan Looker Studio agar dashboard menjadi lebih interaktif dan memudahkan untuk membaca data dalam pengambilan keputusan.

Looker Studio

Data preparation and Data cleaning

Objective

- Membuat program yang dapat dijalankan untuk melakukan data preparation & cleaning
- Dapat melakukan data preparation dan cleaning lebih mudah serta cepat
- Membuat dataset baru sesuai dengan kebutuhan
- Menghasilkan data yang siap untuk divisualisasikan

Data Preparation & Cleaning

Check Dataset

```
dataset.duplicated().sum()
```

```
0
```

Memeriksa dataset apakah terdapat data yang terduplikasi atau tidak

Cek dataset yang akan digunakan untuk mengetahui apakah terdapat missing value dan tipe data yang digunakan.

```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6015 entries, 0 to 6014
Data columns (total 67 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   ORDER_ID_NEW           6015 non-null   int64
 1   REGIONAL               6015 non-null   int64
 2   WITEL                  6015 non-null   object
 3   DATEL                  6015 non-null   object
 4   STO                    6015 non-null   object
 5   UNIT                   6015 non-null   object
 6   JENISPSB               6015 non-null   object
 7   TYPE_TRANS             6015 non-null   object
 8   TYPE_LAYANAN           5965 non-null   object
 9   CHANNEL                6013 non-null   object
 10  GROUP_CHANNEL          5867 non-null   object
 11  STATUS_RESUME          6015 non-null   object
 12  STATUS_MESSAGE         6015 non-null   object
 13  PROVIDER               6015 non-null   object
 14  LAST_ORDER_DATE        6015 non-null   datetime64[ns]
 15  LAST_UPDATED_DATE      6015 non-null   datetime64[ns]
 16  ORDER_ID_OLD           6015 non-null   int64
 17  STATUS_RESUME_OLD      5922 non-null   object
 18  STATUS_MESSAGE_OLD     5922 non-null   object
 19  ORDER_DATE_OLD         5922 non-null   datetime64[ns]
 20  LAST_UPDATED_DATE_OLD  5922 non-null   datetime64[ns]
 21  TGL_PI_AWAL            3872 non-null   datetime64[ns]
 22  TGL_PI_AKHIR           3861 non-null   datetime64[ns]
 23  DEVICE_ID              5995 non-null   object
 24  PACKAGE_NAME           6014 non-null   object
 25  HIDE                   0 non-null      float64
 26  LOC_ID                 6010 non-null   object
```

Membuat kolom Area

```
def apply_area(row):
    if row['STO'] == "BMK":
        return "BANYUMANIK"
    elif row['STO'] == "MJP":
        return "MAJAPAHIT"
    elif row['STO'] == "SLI":
        return "SALATIGA"
    elif row['STO'] == "SSL":
        return "SIMPANG LIMA"
    elif row['STO'] in ["TGU", "SMT", "BOJ", "MJE", "MKG"]:
        return "TUGU"
    elif row['STO'] in ["JHR", "GNK"]:
        return "JOHAR"
    elif row['STO'] in ["CDI", "SMC"]:
        return "CANDI"
    elif row['STO'] in ["KDL", "WLR", "SKR"]:
        return "KENDAL"
    else:
        return "UNGARAN"


dataset['AREA'] = dataset.apply(apply_area, axis=1)
```

Nilai pada kolom Area sesuai dengan kode data pada setiap STO yang ada.

Menghilangkan kolom

Dalam kasus dataset yang digunakan, terdapat beberapa kolom yang tidak akan digunakan. Kolom yang dihapus yaitu 34 kolom.

```
Column_drop = ['DATEL','UNIT','JENISPSB','CHANNEL','GROUP_CHANNEL','PROVIDER','PACKAGE_NAME','HIDE',
               'POTS','CUSTOMER_NAME','CONTACT_HP','INS_ADDRESS','KCONTACT','CATEGORY',
               'UMUR','TINDAK_LANJUT','ISI_COMMENT','TGL_COMMENT','USER_ID_TL','WONUM',
               'DESC_TASK','STATUS_TASK','SCHEDULE_LABOR','AMCREW','OLT_RX','ONU_RX','SNR_UP',
               'SNR_DOWN','FLAG_DEPOSIT','LAST_VIEW','UKUR_TIME','TEKNISI','MSISDN','SN']

dataset = dataset.drop(Column_drop, axis=1)
```

Membuat kolom flag kendala

```
def apply_flag_kendala(row):
    status_resume = row['STATUS_RESUME']
    if status_resume is not None and 'fallout' in status_resume.lower():
        return "FALLOUT"
    else:
        return None

dataset['FLAG_KENDALA'] = dataset.apply(apply_flag_kendala, axis=1)
```

Nilai yang ada di kolom flag kendala akan dibuat berdasarkan dari kolom status resume dengan kategorisasi :

- Jika nilai yang ada di kolom status resume mempunyai karakter ‘fallout’ maka flag kendala ‘FALLOUT’
- Jika nilai yang ada di kolom status resume tidak mempunyai karakter ‘fallout’ maka flag kendala ‘None’

Membuat kolom status group

Nilai yang ada di kolom Status Resume akan dibuat berdasarkan dari kolom Status Group dengan kategorisasi :

- Jika nilai yang ada di kolom status resume mempunyai karakter ‘cancel ‘ maka status group ‘CANCEL’
- Jika nilai yang ada di kolom status resume mempunyai karakter ‘revoke‘ maka status group ‘REVOKE

```
def map_status(status):

    if status is not None and 'cancel' in status.lower():
        return 'CANCEL'
    if status is not None and 'revoke' in status.lower():
        return 'REVOKE'
    if status is not None and 'un-scs' in status.lower():
        return 'UNSCS'
    if status is not None and 'fallout' in status.lower():
        return 'FALLOUT'
    if status is not None and 'completed' in status.lower():
        return 'PS'
    else:
        return 'PI'

dataset['Status_Group'] = dataset['STATUS_RESUME'].apply(map_status)
```

- Jika nilai yang ada di kolom status resume mempunyai karakter ‘un-scs ‘ maka status group ‘UNSCS’
- Jika nilai yang ada di kolom status resume mempunyai karakter ‘fallout’ maka status group ‘FALLOUT’
- Jika nilai yang ada di kolom status resume mempunyai karakter ‘completed’ maka status group ‘PS’
- Jika nilai yang ada di kolom status resume selain yang di atas maka status group ‘PI’

Membuat kolom VAL dan RE

Nilai yang ada di kolom FLAG Val akan dibuat berdasarkan dari kolom Status Group dengan kategorisasi :

- Jika nilai yang ada di kolom status group mempunyai karakter ‘cancel ‘ atau ‘revoke’ maka flag val ‘None’
- Jika nilai yang ada di kolom status group tidak mempunyai karakter ‘cancel ‘ atau ‘revoke“ maka flag val ‘VAL’

```
def apply_flag_val(status_group):
    if status_group and any(keyword in status_group.lower() for keyword in ['cancel', 'revoke']):
        return None
    else:
        return 'VAL'

dataset['FLAG_VAL'] = dataset['Status_Group'].apply(apply_flag_val)
dataset['FLAG_RE'] = 'RE'
```

Membuat kolom Group Kendala

```
def apply_group_kendala(row):
    status_resume = row['STATUS_RESUME']

    # Check if 'completed' or 'un-scs' is present in STATUS_RESUME
    if 'complete' in status_resume.lower() or 'un-scs' in status_resume.lower():
        return None

    group_kendala = row['SUBERRORCODE_WOA']
    if group_kendala is not None and pd.notna(group_kendala):
        return group_kendala
    else:
        return None

dataset['Group_Kendala'] = dataset.apply(apply_group_kendala, axis=1)
dataset['Group_Kendala'] = dataset['Group_Kendala'].apply(lambda x: str(x).upper() if isinstance(x, str) else x)
dataset.loc[(dataset['Group_Kendala'].isna()) & (dataset['FLAG_KENDALA'].isin(['FALLOUT'])), 'Group_Kendala'] = 'OTHERS'
```

Nilai yang ada di kolom “Group_Kendala” dibuat berdasarkan dari kolom “Status_Resume” dengan kategorisasi :

- Jika terdapat karakter “complete” atau “un-scs” pada “Status_resume” maka nilai kosong.
- Jika tidak terdapat 2 karakter tersebut maka nilai akan sama dengan yang ada di kolom “SUBERRORCODE_WOA”

Membuat kolom Cek Revoke

```
def check_revoke(row):
  order_id_new = row['ORDER_ID_NEW']
  if order_id_new != row['ORDER_ID_OLD']:
    return 'Ada Revoke'
  else:
    return 'No Revoke'

dataset['CEK_REVOKE'] = dataset.apply(check_revoke,axis=1)
```

Nilai yang ada di kolom Cek Revoke akan dibuat berdasarkan dari kolom id order new dan old dengan kategorisasi :

- Jika nilai yang ada di kolom id order new dan old mempunyai nilai yang sama maka cek revoke ‘No Revoke’
- Jika nilai yang ada di kolom id order new dan old mempunyai nilai tidak sama maka cek revoke ‘Ada Revoke’

Merubah format tanggal

```
import datetime
dataset['LAST_ORDER_DATE'] = pd.to_datetime(dataset['LAST_ORDER_DATE'], format='%Y-%m-%d %H:%M:%S', errors='coerce')
dataset['LAST_ORDER_DATE'] = dataset['LAST_ORDER_DATE'].dt.date
dataset['LAST_UPDATED_DATE'] = pd.to_datetime(dataset['LAST_UPDATED_DATE'], format='%Y-%m-%d %H:%M:%S', errors='coerce')
dataset['LAST_UPDATED_DATE'] = dataset['LAST_UPDATED_DATE'].dt.date
dataset['ORDER_DATE_OLD'] = pd.to_datetime(dataset['ORDER_DATE_OLD'], format='%Y-%m-%d %H:%M:%S', errors='coerce')
dataset['ORDER_DATE_OLD'] = dataset['ORDER_DATE_OLD'].dt.date
```

Mengubah tipe data pada kolom ‘LAST_ORDER_DATE’, ‘LAST_UPDATED_DATE’, dan ‘ORDER_DATE_OLD’ menjadi datetime.

Membuat kolom usia PS

Menghitung umur ps dengan kategorisasi :

- Jika terdapat nilai ‘Completed (PS)’ yang ada di kolom ‘STATUS_RESUME’ dan nilai ‘No Revoke’ pada kolom ‘CEK_REVOKE’ maka umur ps akan dihitung dengan cara nilai dari tanggal ‘LAST_UPDATE_DATE’ dikurangi ‘LAST_ORDER_DATE’.

```
from datetime import datetime
def usia(row):
  status = row['STATUS_RESUME']
  revoke = row['CEK_REVOKE']
  if status == 'Completed (PS)' and revoke == 'No Revoke':
    return row['LAST_UPDATED_DATE'] - row['LAST_ORDER_DATE']
  elif status == 'Completed (PS)' and revoke == 'Ada Revoke':
    return row['LAST_UPDATED_DATE'] - row['ORDER_DATE_OLD']
  else:
    return None

dataset['USIA_TO_PS(HARI)'] = dataset.apply(usia,axis=1)
```

- Jika terdapat nilai ‘Completed (PS)’ yang ada di kolom ‘STATUS_RESUME’ dan nilai ‘Ada Revoke’ pada kolom ‘CEK_REVOKE’ maka umur ps akan dihitung dengan cara nilai dari tanggal ‘LAST_UPDATE_DATE’ dikurangi ‘ORDER_DATE_OLD’.

Membuat kolom range usia PS

```
def range_usia(usia):
    usia_days = usia.days
    if usia_days <= 7:
        return "1 - 7H"
    elif 8 <= usia_days <= 14:
        return "8 - 14H"
    elif 15 <= usia_days <= 21:
        return "15 - 21H"
    elif 22 <= usia_days <= 30:
        return "22 - 30H"
    elif usia_days > 30:
        return ">30H"
    else :
        return None

dataset['Range_Usia_PS'] = dataset['USIA_TO_PS(HARI)'].apply(range_usia)
```

- Nilai usia ps lebih dari 30 hari maka range usia >30H

Menghitung rentang nilai pada usia PS dalam hitungan hari dengan kategorisasi :

- Nilai usia ps kurang dari 7 hari maka range usia 1 - 7H
- Nilai usia ps di antara 8 sampai 14 hari maka range usia 8 - 14H
- Nilai usia ps di antara 15 sampai 21 hari maka range usia 15 - 21H
- Nilai usia ps di antara 22 sampai 30 hari maka range usia 22 - 30H

Membuat kolom usia dan range usia wo

```
def usia_wo(row):
  revoke = row['CEK_REVOKE']
  if revoke == 'No Revoke':
    return row['LAST_UPDATED_DATE'] - row['LAST_ORDER_DATE']
  elif revoke == 'Ada Revoke':
    return row['LAST_UPDATED_DATE'] - row['ORDER_DATE_OLD']
  else:
    return None

dataset['USIA WO ALL (HARI)'] = dataset.apply(usia_wo, axis=1)
dataset['Range_Usia_WO'] = dataset['USIA WO ALL (HARI)'].apply(range_usia)
```

Untuk perhitungan nilai pada kolom ‘RANGE_USIA_WO’ menggunakan fungsi yang sama dengan yang digunakan perhitungan nilai pada kolom ‘RANGE_USIA_PS’

Menghitung umur wo dengan kategorisasi :

- Jika terdapat nilai ‘No Revoke’ yang ada di kolom ‘CEK_REVOKE’ maka umur wo akan dihitung dengan cara nilai dari tanggal ‘LAST_UPDATE_DATE’ dikurangi ‘LAST_ORDER_DATE’.
- Jika terdapat nilai Ada Revoke’ yang ada di kolom ‘CEK_REVOKE’ maka umur wo akan dihitung dengan cara nilai dari tanggal ‘LAST_UPDATE_DATE’ dikurangi ‘ORDER_DATE_OLD’.

Merubah format tanggal

```
dataset['TGL_PI_AKHIR'] = pd.to_datetime(dataset['TGL_PI_AKHIR'])
dataset['TGL_PI_AWAL'] = pd.to_datetime(dataset['TGL_PI_AWAL'])
```

Merubah format dari kolom “TGL_PI_AKHIR” dan “TGL_PI_AWAL” menjadi datetime

Membuat kolom durasi PI dan SLA PSB

```
def durasi_pi(row):
  tgl_pi = row['TGL_PI_AKHIR']
  if tgl_pi is not None:
    return tgl_pi - row['TGL_PI_AWAL']
  else:
    return None

dataset['DURASI PI (JAM)'] = dataset.apply(durasi_pi, axis=1)
dataset['DURASI PI (JAM)'] = round(dataset['DURASI PI (JAM)'].dt.total_seconds() / 3600)
```

Menghitung durasi PI dengan kategorisasi:

- Jika terdapat nilai di kolom “TGL_PI_AKHIR” maka durasi umur pi akan dihitung dengan cara “TGL_PI_AKHIR” dikurangi nilai pada kolom “TGL_PI_AWAL”
- Jika tidak ada nilai di kolom “TGL_PI_AKHIR” maka nilai yang dihasilkan “None”.

Membuat kolom durasi PI dan SLA PSB

```
dataset['SLA PSB'] = dataset['DURASI PI (JAM)'].apply(lambda x: 'SLA NOT ACHIEVE' if x is not None and x > 72 else 'SLA ACHIEVE')
```

Nilai yang ada di “SLA PSB” akan ditentukan berdasarkan nilai pada “DURASI PI (JAM) dengan kategorisasi:

- Jika nilai yang di “DURASI PI (JAM)” lebih dari 72 maka sla psb “SLA NOT ACHIEVE”.
- Jika nilai yang di “DURASI PI (JAM)” tidak lebih dari 72 maka sla psb “SLA ACHIEVE”.

Membuat kolom usia dan range usia fallout

Untuk perhitungan nilai pada kolom ‘RANGE_USIA_FALLOUT’ menggunakan fungsi yang sama dengan yang digunakan perhitungan nilai pada kolom ‘RANGE_USIA_PS’

```
def usia_fallout(row):
  status_resume = row['STATUS_RESUME']
  if 'fallout' in status_resume.lower():
    return row['LAST_UPDATED_DATE'] - row['LAST_ORDER_DATE']
  else:
    return None

dataset['USIA_TO_FALLOUT(HARI)'] = dataset.apply(usia_fallout,axis=1)
dataset['RANGE_USIA_FALLOUT'] = dataset['USIA_TO_FALLOUT(HARI)'].apply(range_usia)
```

Menghitung umur fallout dengan kategorisasi :

- Jika terdapat karakter ‘fallout’ yang ada di kolom ‘STATUS_RESUME’ maka umur fallout akan dihitung dengan cara nilai dari tanggal ‘LAST_UPDATE_DATE’ dikurangi ‘LAST_ORDER_DATE’.
- Jika tidak ada, maka nilai akan ‘None’.

Export Dataset to File Excel

```
import time

fileDate = time.strftime("%d-%m-%Y")
fileExcel = '/content/drive/MyDrive/Report ASO - LEVEL UP/Results/Result_Selfi_' + fileDate + '.xlsx'
# fileExcel = '/content/drive/MyDrive/Report ASO - LEVEL UP/Results/Result_Selfi_Oktober-Desember.xlsx'
dataset.to_excel(fileExcel, index=False)
```

Setelah semua dilakukan dalam proses cleaning data, langkah terakhir yaitu mengekspor hasilnya menjadi file excel

Akses untuk file ipynb

Cleaning Data

Data Visualization

Objective

Memberikan visualisasi data yang mudah untuk dipahami audience

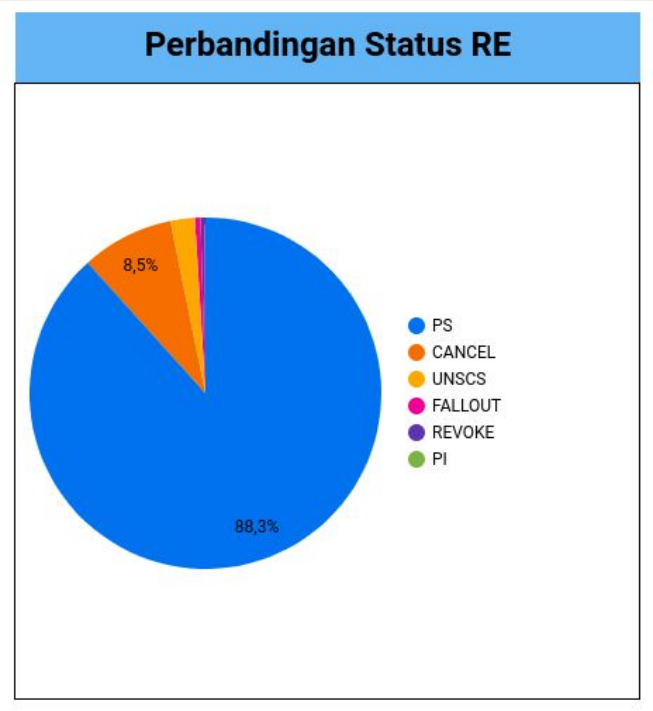
Memudahkan untuk mendapatkan insight melalui visualisasi yang dibuat

Membantu pada saat pengambilan keputusan

Menyajikan data yang sesuai dengan yang diolah

Data Visualization

Visualization Item 1 : Perbandingan Status RE

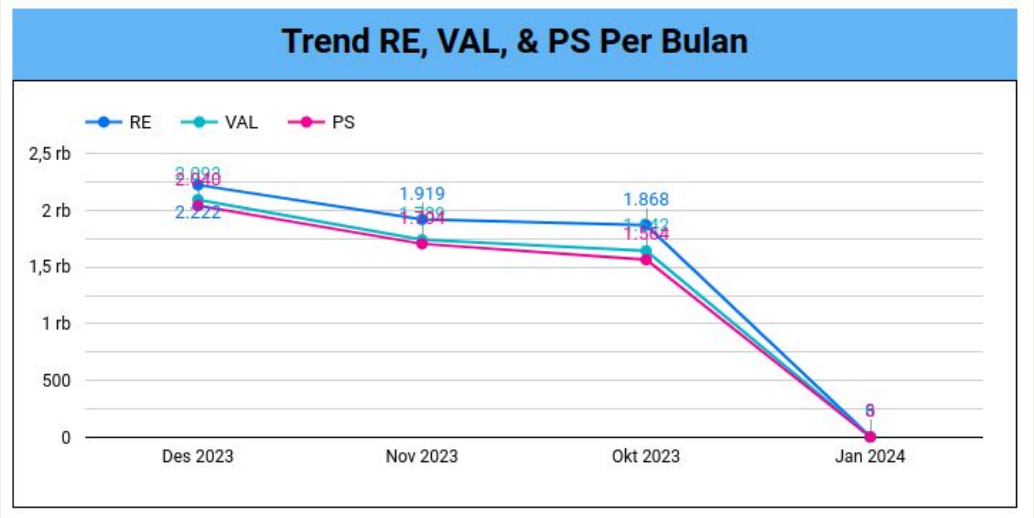


Insight :
Status PS lebih banyak dengan yang lainnya. Secara persentase, status PS terjadi sebanyak 88.3% dari total status RE yang didapatkan dari data yang ada.

Tindak Lanjut :
Melakukan penanggulangan untuk kendala yang masih terjadi agar performance meningkat.

Data Visualization

Visualization Item 2 : Trend RE, VAL, & PS Per Bulan

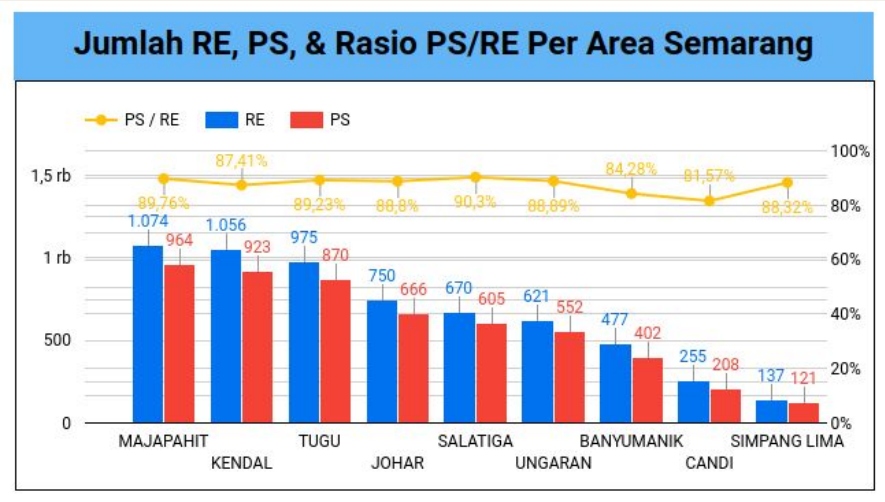


Insight :
Terjadi peningkatan pada performansi RE, VAL dan PS di setiap bulannya.

Tindak Lanjut :
Melakukan diskusi terkait performansi per bulan dengan setiap bagian yang bertanggung jawab agar tidak ada kendala dalam pelaksanaan tugas.

Data Visualization

Visualization Item 3 : Jumlah RE, PS, & Rasio PS/RE



Insight :

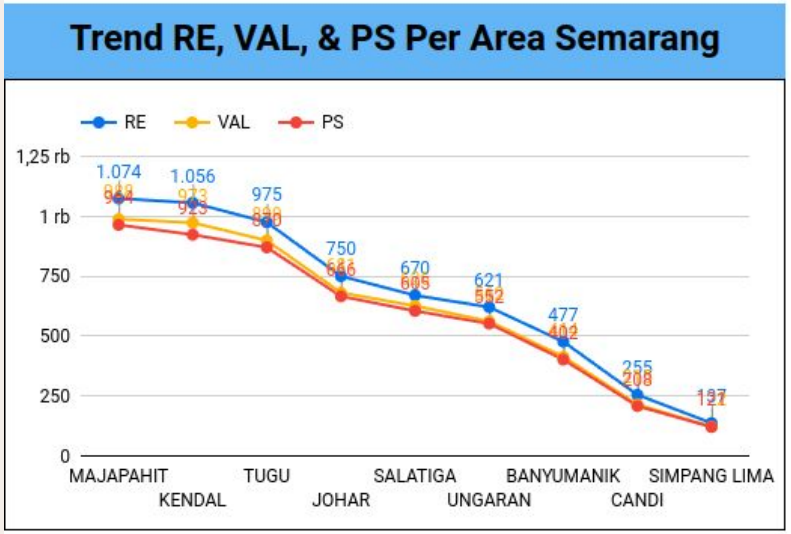
Jumlah RE, PS serta perbandingannya di area semarang memiliki perbedaan yang variatif. Untuk rasio perbandingan PS/RE paling tinggi dimiliki oleh Salatiga dengan nilai persentase 90.3%. Sedangkan yang terendah dimiliki oleh Candi dengan nilai persentase 81.54%.

Tindak Lanjut :

Melakukan konfirmasi mengenai STO yang bersangkutan dan mendiskusikan apakah terdapat masalah yang terjadi. Karena jumlah yang didapatkan tidak sama dengan area lainnya.

Data Visualization

Visualization Item 4 : Trend RE, VAL, & PS Per Area Semarang

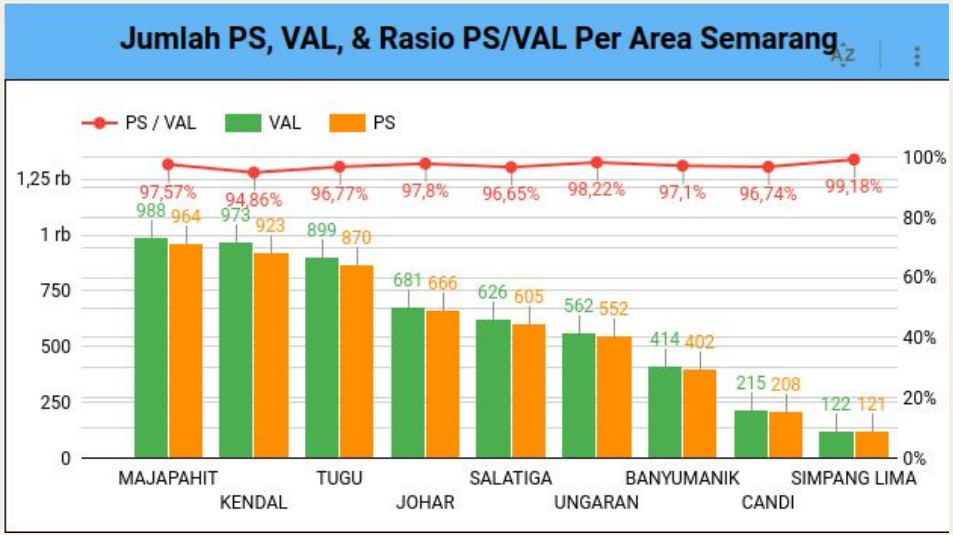


Insight :
Jumlah dari kuantitas status RE, VAL dan PS yang di area semarang memiliki perbedaan yang variatif. Jumlah paling tinggi terdapat di area Majapahit, sedangkan yang paling rendah di area Simpang Lima.

Tindak Lanjut :
Melakukan evaluasi pada STO yang bersangkutan untuk menemukan solusi untuk menyelesaikan status permohonan yang masih terdapat kendala sehingga membuat perbedaan pada kuantitas di VAL dan PS.

Data Visualization

Visualization Item 5 : Jumlah PS, VAL, & Rasio PS/VAL



Insight :

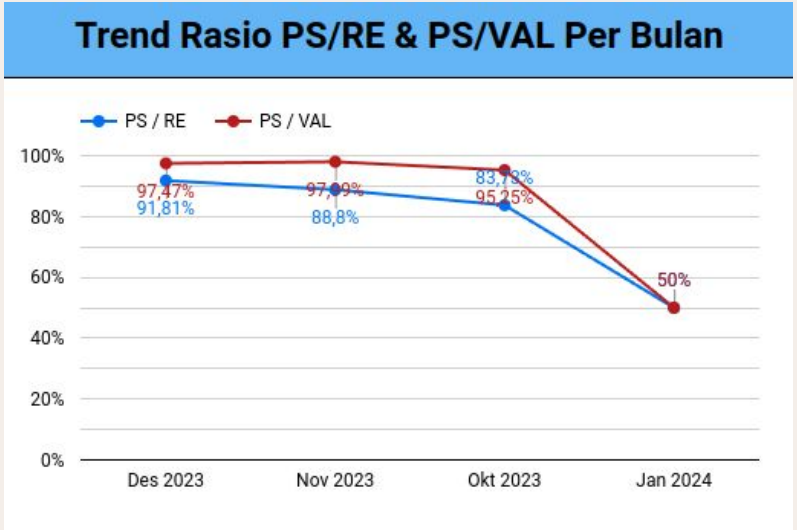
Jumlah PS, VAL serta perbandingannya di area semarang memiliki perbedaan yang variatif. Untuk rasio perbandingan PS/RE paling tinggi dimiliki oleh Simpang Lima dengan nilai persentase 99.18%. Sedangkan yang terendah dimiliki oleh Kendal dengan nilai persentase 94.86%.

Tindak Lanjut :

Melakukan konfirmasi mengenai STO yang bersangkutan dan mendiskusikan apakah terdapat masalah yang terjadi. Karena jumlah yang didapatkan tidak sama dengan area lainnya.

Data Visualization

Visualization Item 6 : Trend Rasio PS/RE & PS/VAL

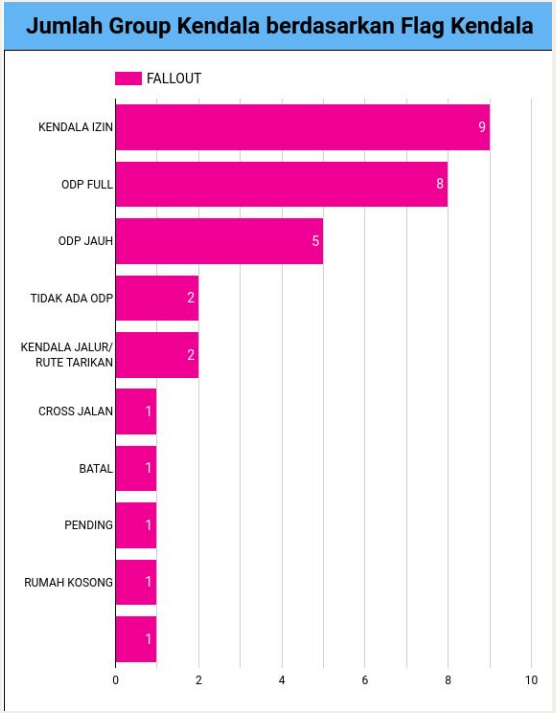


Insight :
Terdapat penurunan performansi dari PS/VAL pada bulan Desember. Pada bulan november memiliki nilai 97.99% dan turun pada bulan Desember menjadi 97.47%. Terjadi penurunan sebesar 00.52%.

Tindak Lanjut :
Melakukan diskusi dengan bagian yang bertanggung jawab terkait performansi PS dan VAL pada bulan desember untuk mencari tahu penyebabnya. Agar tidak terulang untuk bulan kedepannya

Data Visualization

Visualization Item 7 : Jumlah Group Kendala

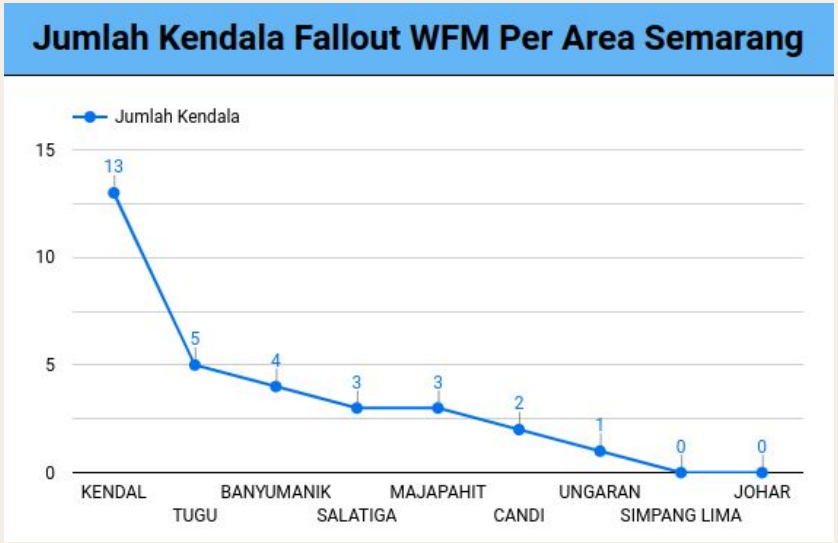


Insight :
Kendala izin menjadi kendala yang paling sering terjadi dibandingkan dengan yang lain.

Tindak Lanjut :
Melakukan diskusi kepada yang bersangkutan terkait pemasangan agar dapat mencari cara menanggulangi kendala ketika kendala tersebut terjadi.

Data Visualization

Visualization Item 8 : Jumlah Kendala Fallout WFM

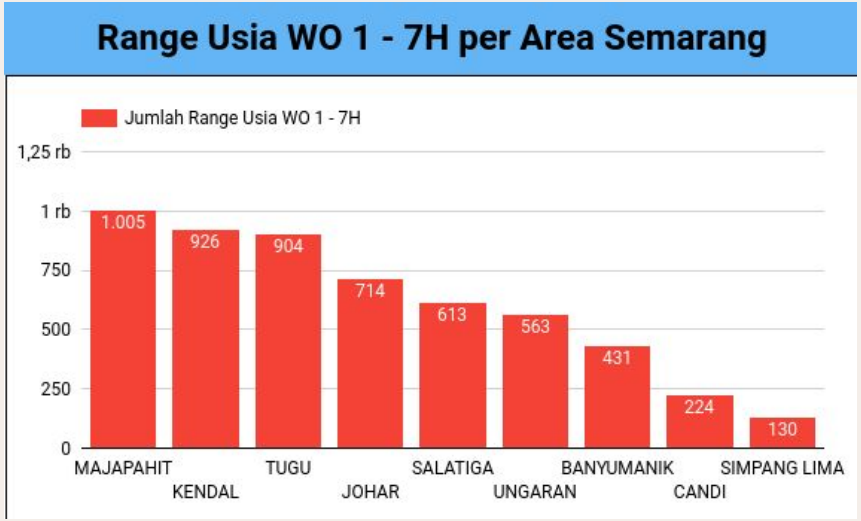


Insight :
Area Kendal memiliki kendala paling banyak dibandingkan dengan yang lainnya. Selain itu, terdapat area yang tidak memiliki kendala yaitu Simpang Lima dan Johar.

Tindak Lanjut :
Melakukan evaluasi di setiap area yang memiliki jumlah kendala khususnya yang jumlah kendalanya banyak untuk mencari tahu apakah terdapat faktor tertentu yang menyebabkan kendala tersebut.

Data Visualization

Visualization Item 9 : Range Usia WO

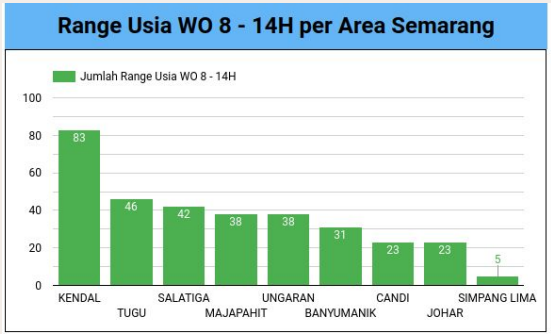


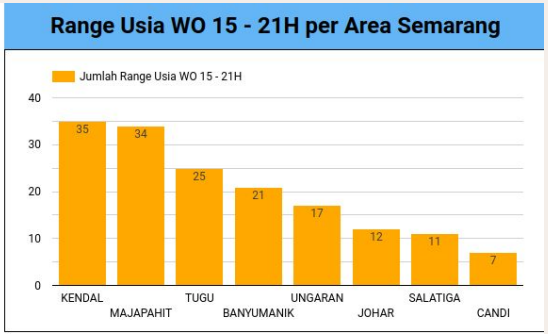
Insight :
Majapahit memiliki kuantitas rentang usia WO 1-7H paling banyak, mencapai 1005. Berdasarkan hal tersebut berarti area tersebut banyak menerima order baru.

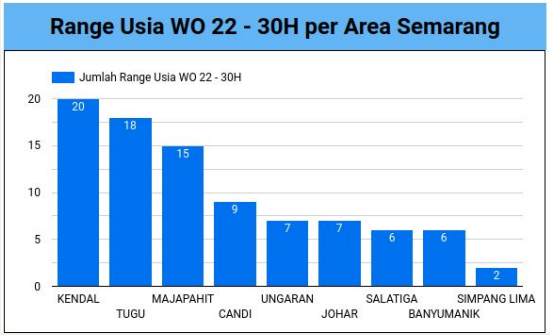
Tindak Lanjut :
Melakukan koordinasi dengan bagian yang bertanggung jawab di setiap STO untuk mencari tahu apakah sering terjadi kendala pada area tersebut.

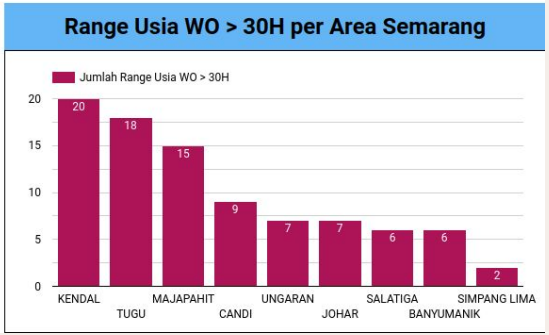
Data Visualization

Visualization Item 10 : Range Usia WO







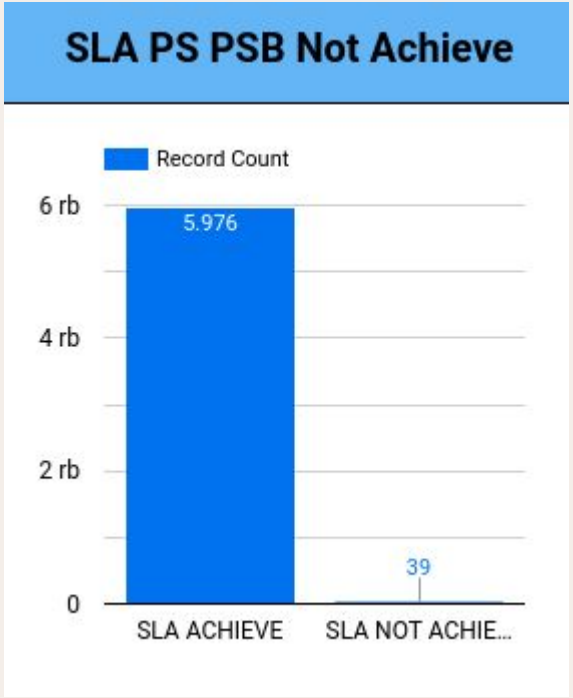


Insight :
Kendal selalu mendapatkan kuantitas tertinggi dengan range usia wo yang berbeda beda.

Tindak Lanjut :
Melakukan koordinasi dengan bagian yang bertanggung jawab di setiap STO untuk mencari tahu apakah sering terjadi kendala pada area tersebut.

Data Visualization

Visualization Item 11 : SLA PS PSB Not Achieve



Insight :
Terdapat 39 SLA yang tidak terselesaikan dari 6015.

Tindak Lanjut :
Melakukan evaluasi kepada pihak yang bertanggung jawab. Apakah yang menyebabkan tidak terselesaikannya SLA PS PSB ini.

Data Visualization

Visualization Item 12 : Tabel Selfi

Tabel Selfi

	ORDER_ID_NEW	STATUS_MESSAGE	LAST_ORDER_DATE	LAST_UPDATED_DATE	ORDER_ID_OLD	STATUS_RESUME_OLD
1.	543354365	Completed	14 Okt 2023	14 Okt 2023	543354365	Completed (PS)
2.	543500513	Completed	27 Okt 2023	27 Okt 2023	543500513	Completed (PS)
3.	543499619	Completed	27 Okt 2023	28 Okt 2023	543499619	Completed (PS)
4.	543502395	Completed	27 Okt 2023	28 Okt 2023	543502395	Completed (PS)
5.	543395004	Completed	18 Okt 2023	18 Okt 2023	543395004	Completed (PS)
6.	543502512	Completed	27 Okt 2023	27 Okt 2023	543502512	Completed (PS)
7.	543392905	Inprogress Alpro, SC ...	18 Okt 2023	18 Okt 2023	543361124	98 | OSS - CANCEL COMP.
8.	543503177	Cancel Completed	27 Okt 2023	7 Nov 2023	543503177	98 | OSS - CANCEL COMP.

1 - 100 / 6015

Dalam tabel ini berisikan data yang telah divisualkan. Tabel ini memudahkan pengecekan data dari secara manual. Seperti mencari ORDER_ID_NEW tertentu, dsb.

Akses untuk dashboard selfi

Dashboard Selfi