📊 Sesi 1

Python & Pandas untuk Analisis Data

HARI 1: Fundamental Analisis Data

Durasi: 3 jam (09:00 - 12:00)
Notebook: `01_exploratory_data_analysis_rup.ipynb`

🎯 Tujuan Sesi

Setelah sesi ini, Anda bisa:

- ✅ Memuat dan mengeksplorasi dataset dengan Pandas
- ✅ Melakukan seleksi dan filtering data
- ✅ Agregasi data dengan GroupBy operasi
- ✅ Menangani missing values
- ✅ Membuat statistical summary
- ✅ Visualisasi dasar dengan Pandas

Dataset: RUP 2025 (16,430 paket pengadaan)

📋 Agenda Sesi

Waktu	Topik	Durasi
08:00 - 08:15	Kick-off & Tujuan Sesi	15 min
08:15 - 09:00	Persiapan Environment & Data Loading	45 min
09:00 - 10:00	Exploratory Data Analysis Fundamentals	60 min
10:00 - 11:00	Data Cleaning & Handling Missing Values	60 min
11:00 - 11:45	Aggregasi & GroupBy Lanjutan	45 min
11:45 - 12:00	Visualisasi Cepat & Wrap-up	15 min

🛠️ Part 1: Persiapan Environment

Setup Virtual Environment

```
# Opsi 1: Menggunakan uv (recommended)
uv sync
uv run jupyter notebook

# Opsi 2: Menggunakan venv + pip
python -m venv .venv
source .venv/bin/activate  # Linux/Mac
.venv\Scripts\activate     # Windows
pip install pandas numpy jupyter pyarrow openpyxl
jupyter notebook
```

📦 Import Libraries

```
# Data pengolahan
import pandas as pd
import numpy as np

# Visualization
import matplotlib.pyplot as plt
import seaborn as sns

# Display settings
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', 100)
pd.set_option('display.float_format', '{:.2f}'.format)

# Plotting style
plt.style.use('seaborn-v0_8')
sns.set_palette("husl")

print(f"Pandas version: {pd.__version__}")
print(f"NumPy version: {np.__version__}")
```

📂 Loading Data

Membaca File Parquet

```
# Load dataset RUP 2025
df = pd.read_parquet('../../datasets/rup/RUP-PaketPenyedia-Terumumkan-2025.parquet')

# Quick peek
print(f"Dataset shape: {df.shape}")
print(f"Rows: {df.shape[0]:,}")
print(f"Columns: {df.shape[1]}")
```

Output:

```
Dataset shape: (16430, 35)
Rows: 16,430
Columns: 35
```

Mengapa Parquet? Lebih efisien dari CSV (ukuran & kecepatan) ⚡

🔍 Part 2: Data Inspection

Cara Melihat Data dengan `.head()` dan `.tail()`

Penjelasan: `.head()` untuk lihat data dari atas, `.tail()` untuk lihat dari bawah

```
# Lihat 5 baris pertama
df.head()

# Lihat 10 baris pertama
df.head(10)

# Lihat 5 baris terakhir
df.tail()
```

Gunakan untuk:

- ✅ Cek struktur data
- ✅ Lihat contoh records
- ✅ Identifikasi column names
- ✅ Deteksi format data

📊 Data Info dengan `.info()`

```
df.info()
```

Output menunjukkan:

- Jumlah entries (rows)
- Jumlah columns
- Nama kolom
- Non-null count (missing values)
- Data types
- Memory usage

📊 Data Info dengan `.info()` (lanjutan)

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16430 entries, 0 to 16429
Data columns (total 35 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   kd_rup                   16430 non-null  object
 1   nama_paket               16430 non-null  object
 2   pagu                     16430 non-null  float64
...
```

📈 Statistical Summary dengan `.describe()`

```
# Statistik untuk kolom numerik
df.describe()
```

Menghasilkan:

- **count:** jumlah non-null values
- **mean:** rata-rata
- **std:** standard deviation
- **min:** nilai minimum
- **25%, 50%, 75%:** quartiles
- **max:** nilai maximum

📈 Statistical Summary dengan `.describe()` (lanjutan)

```
# Include semua tipe data
df.describe(include='all')

# Hanya untuk kolom kategorikal
df.describe(include='object')
```

🔎 Explorasi Kolom Dataset RUP

Kolom-Kolom Penting

```
# Lihat semua nama kolom
print(df.columns.tolist())

# Contoh kolom penting:
# - kd_rup: Kode RUP
# - nama_paket: Nama paket pengadaan
# - pagu: Pagu anggaran (Rupiah)
# - metode_pengadaan: Tender/Penunjukan Langsung/dll
# - jenis_pengadaan: Barang/Jasa Konsultansi/Pekerjaan Konstruksi
# - nama_satker: Nama satuan kerja
# - tgl_pengumuman_paket: Tanggal pengumuman
```

🎯 Part 3: Seleksi & Penyaringan Data

Selecting Columns

```python
# Satu kolom (Series)
nama_paket = df['nama_paket']

# Multiple kolom (DataFrame)
subset = df[['nama_paket', 'pagu',
             'metode_pengadaan']]

# Lihat hasil
print(subset.head())
```

Tips:

- Single `[]` → Series
- Double `[[]]` → DataFrame
- Gunakan list untuk multiple columns

🔍 Selecting Rows dengan `.loc[]` dan `.iloc[]`

`.loc[]` - Berbasis label

```
# Rows 0 sampai 4
df.loc[0:4]

# Specific rows & columns
df.loc[0:4,
       ['nama_paket', 'pagu']]

# Boolean indexing
high_value = df.loc[
    df['pagu'] > 1_000_000_000
]
```

`.iloc[]` - Berbasis integer

```
# Rows 0 sampai 4 (exclusive)
df.iloc[0:5]

# Specific rows & columns
df.iloc[0:5, [0, 1, 2]]

# Pertama 100 rows
df.iloc[:100]

# Terakhir 50 rows
df.iloc[-50:]
```

Boolean Indexing (Filtering)

Filter Data Berdasarkan Kondisi

```
# Filter pagu > 1 Miliar
high_value_packages = df[df['pagu'] > 1_000_000_000]
print(f"Paket dengan pagu > 1M: {len(high_value_packages):,}")

# Filter metode pengadaan = Tender
tender_only = df[df['metode_pengadaan'] == 'Tender']

# Multiple conditions dengan &(AND) atau |(OR)
# Pagu > 1M DAN metode = Tender
result = df[(df['pagu'] > 1_000_000_000) &
            (df['metode_pengadaan'] == 'Tender')]

# Pagu > 1M ATAU metode = Penunjukan Langsung
result = df[(df['pagu'] > 1_000_000_000) |
            (df['metode_pengadaan'] == 'Penunjukan Langsung')]
```

⚠️ **Penting:** Gunakan `&` (AND) dan `|` (OR), bukan `and`/`or`

🔎 Query Method untuk Filter Kompleks

```python
# Lebih readable untuk kondisi rumit
result = df.query('pagu > 1_000_000_000 and metode_pengadaan == "Tender"')

# Dengan variabel (tempat simpan data)
threshold = 1_000_000_000
result = df.query('pagu > @threshold')

# String contains
result = df.query('nama_paket.str.contains("Pengadaan", case=False)')

# Multiple conditions
result = df.query('''
    pagu > 1_000_000_000 and
    metode_pengadaan == "Tender" and
    jenis_pengadaan == "Barang"
''')
```

Keuntungan: Lebih mudah dibaca, mirip SQL WHERE clause

Part 4: Data Aggregation

Cara Kelompokkan Data dengan GroupBy

Penjelasan: GroupBy seperti membuat ringkasan data berdasarkan kategori, misalnya total per metode

Konsep GroupBy

1. **Split** - Pisahkan data berdasarkan kategori
2. **Apply** - Terapkan fungsi agregasi
3. **Combine** - Gabungkan hasil

Part 4: Data Aggregation (lanjutan)

```
# Group by metode pengadaan, hitung total pagu
grouped = df.groupby('metode_pengadaan')['pagu'].sum()
print(grouped)

# Hasilnya:
# metode_pengadaan
# Penunjukan Langsung              5,234,567,890
# Tender                           45,678,901,234
# Tender Cepat                     12,345,678,901
# Name: pagu, dtype: int64
```

🎯 GroupBy: Agregasi Multiple Functions

```python
# Gabung dengan multiple functions
agg_result = df.groupby('metode_pengadaan')['pagu'].agg([
    'count',  # Jumlah paket
    'sum',    # Total pagu
    'mean',   # Rata-rata pagu
    'median', # Median pagu
    'min',    # Pagu minimum
    'max'     # Pagu maksimum
])

print(agg_result)
```

Output:

```
                      count           sum         mean    median  ...
metode_pengadaan                                                  ...
Penunjukan Langsung    1234  5.23e+09    4.24e+06   2.5e+06    ...
Tender                 5678  4.57e+10    8.05e+06   5.0e+06    ...
```

📈 GroupBy: Multiple Columns

```
# Group by multiple columns
grouped = df.groupby(['metode_pengadaan', 'jenis_pengadaan'])['pagu'].agg([
    'count',
    'sum',
    'mean'
]).round(2)

# Reset index untuk DataFrame yang lebih mudah dipakai
grouped_reset = grouped.reset_index()

# Rename columns
grouped_reset.columns = ['Metode', 'Jenis', 'Jumlah_Paket',
                         'Total_Pagu', 'Rata_Rata_Pagu']

print(grouped_reset.head(10))
```

🏆 Top 10 Satker dengan Pagu Terbesar

```python
# Group by satker, sum pagu, sort descending
top_satker = (
    df.groupby('nama_satker')['pagu']
    .sum()
    .sort_values(ascending=False)
    .head(10)
)

# Konversi ke DataFrame dan format
top_satker_df = top_satker.reset_index()
top_satker_df.columns = ['Satuan Kerja', 'Total Pagu']
top_satker_df['Total Pagu (Miliar)'] = (
    top_satker_df['Total Pagu'] / 1_000_000_000
).round(2)

print(top_satker_df)
```

Method Chaining: Lebih pythonic dan efisien! 🐍

🔄 Part 5: Manipulasi Data

Sorting Data

```
# Sort by single column
df_sorted = df.sort_values('pagu', ascending=False)

# Sort by multiple columns
df_sorted = df.sort_values(
    ['metode_pengadaan', 'pagu'],
    ascending=[True, False]
)

# Sort dan reset index
df_sorted = df.sort_values('pagu', ascending=False).reset_index(drop=True)

# Lihat top 10
print(df_sorted.head(10))
```

🕵️ Handling Missing Values

Deteksi Missing Values

```python
# Cek missing values per kolom
missing = df.isnull().sum()
print(missing[missing > 0])

# Persentase missing
missing_pct = (df.isnull().sum() / len(df) * 100)
print(missing_pct[missing_pct > 0])

# Visualisasi missing data
import matplotlib.pyplot as plt
missing_pct[missing_pct > 0].plot(kind='barh')
plt.title('Percentage of Missing Values')
plt.xlabel('Percentage (%)')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

🛠️ Treatment Missing Values

Drop Missing

```python
# Drop rows dengan missing
df_clean = df.dropna()

# Drop jika kolom tertentu null
df_clean = df.dropna(
    subset=['pagu',
            'metode_pengadaan']
)

# Drop kolom dengan >50% missing
threshold = 0.5
df_clean = df.dropna(
    thresh=len(df) * threshold,
    axis=1
)
```

Fill Missing

```python
# Fill dengan value tertentu
df_filled = df.fillna(0)

# Fill dengan mean/median
df['pagu'].fillna(
    df['pagu'].mean()
)

# Forward/Backward fill
df.fillna(method='ffill')
df.fillna(method='bfill')

# Fill berbeda per kolom
df.fillna({
    'pagu': 0,
    'metode': 'Unknown'
})
```

📝 String Operations

```
# Mengakses string methods dengan .str
df['nama_paket_upper'] = df['nama_paket'].str.upper()
df['nama_paket_lower'] = df['nama_paket'].str.lower()

# String contains (untuk filtering)
konstruksi = df[df['nama_paket'].str.contains('Konstruksi',
                                              case=False,
                                              na=False)]


# String split
df[['kata1', 'kata2']] = df['nama_paket'].str.split(' ', n=1, expand=True)

# String replace
df['nama_paket_clean'] = df['nama_paket'].str.replace('/', '-')

# String length
df['nama_paket_length'] = df['nama_paket'].str.len()
```

DateTime Operations

```python
# Convert ke datetime
df['tgl_pengumuman_paket'] = pd.to_datetime(
    df['tgl_pengumuman_paket'],
    errors='coerce'
)

# Extract date components
df['tahun'] = df['tgl_pengumuman_paket'].dt.year
df['bulan'] = df['tgl_pengumuman_paket'].dt.month
df['hari'] = df['tgl_pengumuman_paket'].dt.day
df['day_of_week'] = df['tgl_pengumuman_paket'].dt.day_name()

# Filter by date range
mask = (df['tgl_pengumuman_paket'] >= '2025-01-01') & \
       (df['tgl_pengumuman_paket'] <= '2025-03-31')
q1_data = df[mask]
```

Part 6: Ringkasan Statistik

Statistik Deskriptif

```
# Statistik untuk kolom pagu
print(f"Mean:      {df['pagu'].mean():,.2f}")
print(f"Median:    {df['pagu'].median():,.2f}")
print(f"Std Dev:   {df['pagu'].std():,.2f}")
print(f"Min:       {df['pagu'].min():,.2f}")
print(f"Max:       {df['pagu'].max():,.2f}")

# Quartiles
print(f"Q1 (25%): {df['pagu'].quantile(0.25):,.2f}")
print(f"Q2 (50%): {df['pagu'].quantile(0.50):,.2f}")
print(f"Q3 (75%): {df['pagu'].quantile(0.75):,.2f}")

# IQR (Interquartile Range)
q1 = df['pagu'].quantile(0.25)
q3 = df['pagu'].quantile(0.75)
iqr = q3 - q1
print(f"IQR:       {iqr:,.2f}")
```

Deteksi Outliers dengan IQR Method

```
# Calculate IQR
Q1 = df['pagu'].quantile(0.25)
Q3 = df['pagu'].quantile(0.75)
IQR = Q3 - Q1

# Define outlier boundaries
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Detect outliers
outliers = df[(df['pagu'] < lower_bound) | (df['pagu'] > upper_bound)]
print(f"Number of outliers: {len(outliers):,}")
print(f"Percentage: {len(outliers)/len(df)*100:.2f}%")

# Remove outliers
df_no_outliers = df[(df['pagu'] >= lower_bound) &
                    (df['pagu'] <= upper_bound)]
```

🔗 Correlation Analysis

```
# Pilih kolom numerik saja
numeric_cols = df.select_dtypes(include=[np.number]).columns.tolist()

# Calculate correlation matrix
correlation_matrix = df[numeric_cols].corr()

# Lihat korelasi dengan pagu
pagu_corr = correlation_matrix['pagu'].sort_values(ascending=False)
print(pagu_corr)

# Heatmap (akan dipelajari lebih lanjut di sesi visualisasi)
import seaborn as sns
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', center=0)
plt.title('Correlation Matrix')
plt.show()
```

📊 Part 7: Data Visualization dengan Pandas

Basic Plotting

```
# Bar chart - Count per metode
df['metode_pengadaan'].value_counts().plot(kind='bar')
plt.title('Jumlah Paket per Metode Pengadaan')
plt.xlabel('Metode')
plt.ylabel('Jumlah Paket')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# Horizontal bar chart
df['metode_pengadaan'].value_counts().plot(kind='barh')
```

📈 Histogram untuk Distribusi

```python
# Histogram untuk distribusi pagu
df['pagu'].plot(kind='hist', bins=50, edgecolor='black')
plt.title('Distribusi Pagu Pengadaan')
plt.xlabel('Pagu (Rupiah)')
plt.ylabel('Frekuensi')
plt.show()

# Dengan log scale untuk data skewed
df['pagu'].plot(kind='hist', bins=50, edgecolor='black', logy=True)

# Tambahkan mean dan median lines
plt.axvline(df['pagu'].mean(), color='red',
            linestyle='--', label='Mean')
plt.axvline(df['pagu'].median(), color='green',
            linestyle='--', label='Median')
plt.legend()
```

🥧 Pie Chart untuk Komposisi

```
# Pie chart - Distribusi metode pengadaan
metode_counts = df['metode_pengadaan'].value_counts()

plt.figure(figsize=(8, 8))
metode_counts.plot(kind='pie', autopct='%1.1f%%')
plt.title('Distribusi Metode Pengadaan')
plt.ylabel('')  # Remove ylabel
plt.show()

# Dengan explode untuk highlight
explode = (0.1, 0, 0, 0)  # explode 1st slice
metode_counts.plot(kind='pie', autopct='%1.1f%%',
                   explode=explode, shadow=True)
```

Line Chart untuk Time Series

```
# Convert date column
df['tgl_pengumuman_paket'] = pd.to_datetime(
    df['tgl_pengumuman_paket']
)

# Count pengumuman per hari
daily_counts = df.set_index('tgl_pengumuman_paket').resample('D').size()

# Plot
daily_counts.plot(kind='line', figsize=(12, 6))
plt.title('Trend Pengumuman Paket Pengadaan')
plt.xlabel('Tanggal')
plt.ylabel('Jumlah Paket')
plt.grid(True, alpha=0.3)
plt.show()

# Agregasi per bulan
monthly_counts = df.set_index('tgl_pengumuman_paket').resample('M').size()
monthly_counts.plot(kind='line', marker='o')
```

🎨 Styling dan Customization

```
# Multiple plots dengan subplot
fig, axes = plt.subplots(2, 2, figsize=(15, 10))

# Plot 1: Distribution
df['pagu'].plot(kind='hist', bins=50, ax=axes[0,0],
                edgecolor='black')
axes[0,0].set_title('Distribusi Pagu')

# Plot 2: Top Metode
df['metode_pengadaan'].value_counts().plot(kind='bar',
                                           ax=axes[0,1])
axes[0,1].set_title('Metode Pengadaan')

# Plot 3: Top Jenis
df['jenis_pengadaan'].value_counts().plot(kind='barh',
                                          ax=axes[1,0])
axes[1,0].set_title('Jenis Pengadaan')
```

🎨 Styling dan Customization (lanjutan)

```
# Plot 4: Box plot
df.boxplot(column='pagu', by='metode_pengadaan', ax=axes[1,1])

plt.tight_layout()
plt.show()
```

🎯 Praktik Hands-On

Latihan untuk Anda Coba

1. **Analisis Distribusi Pagu**
    - Hitung mean, median, std dev
    - Buat histogram dengan mean/median lines
    - Identifikasi outliers
2. **Identifikasi Top Satker**
    - Find top 10 satker by total pagu
    - Find top 10 satker by jumlah paket
    - Compare keduanya

🎯 Praktik Hands-On (lanjutan)

Latihan tambahan

3. **Analisis Metode Pengadaan**
    - Hitung distribusi per metode
    - Rata-rata pagu per metode
    - Visualisasi dengan bar chart
4. **Trend Analysis**
    - Plot pengumuman paket per bulan
    - Hitung pertumbuhan month-over-month
    - Identifikasi bulan dengan aktivitas tertinggi

🎯 Praktik Hands-On (lanjutan)

Latihan tambahan

5. **Data Quality Check**
    - Identify missing values
    - Find duplicate records
    - Detect outliers
    - Clean dataset
6. **Custom Analysis**
    - Filter paket konstruksi dengan pagu > 1M
    - Analisis per kategori pengadaan
    - Buat laporan ringkasan

💡 Praktik Terbaik

Tips untuk Pemula

- 💡 **Jangan takut error!** Error adalah bagian dari belajar coding
- 💡 **Coba-coba!** Gak akan rusak kok, eksperimen aja
- 💡 **Google adalah teman** Gak apa-apa search solusi di Google

Tips Analisis Data

1. **Selalu mulai dengan EDA** 🔍
    - Pahami data Anda sebelum analisis
2. **Dokumentasikan kode Anda** 📝
    - Gunakan comments dan markdown cells

💡 Praktik Terbaik (lanjutan)

3. **Method chaining agar mudah dibaca** ⛓️

```
result = (df
          .groupby('metode')['pagu']
          .sum()
          .sort_values(ascending=False)
          .head(10))
```

4. **Tangani nilai yang hilang dengan hati-hati** ⚠️
    - Jangan asal drop atau fill

💡 Praktik Terbaik (lanjutan)

5. **Validasi hasil Anda** ✅
    - Periksa ulang dengan berbagai metode
    - Periksa kewajaran (apakah masuk akal?)
6. **Visualisasi untuk insight** 📊
    - A picture is worth a thousand numbers
7. **Save intermediate results** 💾

```
df_clean.to_parquet('cleaned_data.parquet')
df_clean.to_csv('cleaned_data.csv', index=False)
df_clean.to_excel('report.xlsx', index=False)
```

💡 Praktik Terbaik (lanjutan)

8. **Version control your notebooks** 🔄
    - Use Git for tracking changes

⚠️ Kesalahan Umum

Kesalahan Pemula (Normal Kok!)

⚠️ **Semua pernah alami ini!** Jangan frustasi

Kesalahan yang Sering Terjadi

1. **SettingWithCopyWarning**

```
# ❌ Bad
df[df['pagu'] > 1000000]['new_col'] = value

# ✅ Baik
df.loc[df['pagu'] > 1000000, 'new_col'] = value
```

⚠️ Kesalahan Umum (lanjutan)

2. **Chained indexing**

```
# ❌ Bad
df['col'][0] = value

# ✅ Baik
df.loc[0, 'col'] = value
```

3. **Tidak handle missing values**
    - Selalu check sebelum calculation
4. **Lupa reset index setelah filter/sort**

```
df_filtered = df[df['pagu'] > 1000000].reset_index(drop=True)
```

⚠️ Kesalahan Umum (lanjutan)

5. **Menggunakan `and`/`or` daripada `&`/`|` untuk boolean**

```
# ❌ Bad
df[(df['pagu'] > 1000) and (df['metode'] == 'Tender')]

# ✅ Baik
df[(df['pagu'] > 1000) & (df['metode'] == 'Tender')]
```

📚 Poin Penting

Yang Harus Anda Ingat

- ✅ Pandas adalah tool powerful untuk data analysis
- ✅ Selalu mulai dengan `.head()`, `.info()`, `.describe()`
- ✅ `.loc[]` untuk label-based, `.iloc[]` untuk integer-based
- ✅ Boolean indexing untuk filtering
- ✅ GroupBy untuk agregasi dan summary
- ✅ Tangani nilai yang hilang dengan strategi yang tepat
- ✅ Visualisasi untuk memahami data
- ✅ Method chaining untuk code yang clean

Practice makes perfect! 💪

🔗 Sumber Daya Resources & Selanjutnya Steps

Langkah Selanjutnya

Dokumentasi & Cheat Sheets

- **Pandas Documentation:** https://pandas.pydata.org/docs/
- **Pandas Cheat Sheet:** https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf
- **10 Minutes to Pandas:** https://pandas.pydata.org/docs/user_guide/10min.html

Sesi Selanjutnya

Sesi 2: DuckDB untuk Query Analitik

- SQL queries pada dataset besar
- Window functions & CTEs
- Integration dengan Pandas
- Kecepatan comparison

BREAK sampai 13:00 ☕

🎉 Selesai Sesi 1!

Great Job! 👏

Istirahat sampai 13:00

Jangan lupa save notebook Anda!

Pertanyaan?? 🙋

📞 Resources

Jika Butuh Bantuan

- 📚 **Notebook:** `day1/session1_python_pandas/notebooks/01_exploratory_data_analysis_rup.ipynb`
- 📖 **README:** Lihat main README untuk reference
- 💬 **Ask:** Jangan ragu bertanya!
- 🐛 **Issues:** GitHub issues untuk bug reports

Sampai jumpa di Sesi 2! 🚀

📞 Tetap Terhubung

Resources & Support

- 👤 **Nama:** [Kurnia Ramadhan,ST.,M.Eng]
- 📧 **Email:** [kurnia@ramadhan.me]

We're here to support your journey! 🤝