In [3]:

```python
# Import pandas
import pandas as pd

# Load the dataset
file_path = '/Users/parle/Documents/InnoByte Services INTERN/Amazon Sale Report.csv'
data = pd.read_csv(file_path)

# Display the first few rows to understand the structure
data.head()

# Get an overview of the dataset, including column names and data types
data.info()

# Check for missing values in each column
data.isnull().sum()

# Drop completely empty columns
data = data.drop(['New', 'PendingS'], axis=1)

# Verify columns have been removed
data.columns
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 128976 entries, 0 to 128975
Data columns (total 21 columns):
 #   Column              Non-Null Count   Dtype  
---  ------              --------------   -----  
 0   index               128976 non-null  int64  
 1   Order ID            128976 non-null  object 
 2   Date                128976 non-null  object 
 3   Status              128976 non-null  object 
 4   Fulfilment          128976 non-null  object 
 5   Sales Channel       128976 non-null  object 
 6   ship-service-level  128976 non-null  object 
 7   Category            128976 non-null  object 
 8   Size                128976 non-null  object 
 9   Courier Status      128976 non-null  object 
 10  Qty                 128976 non-null  int64  
 11  currency            121176 non-null  object 
 12  Amount              121176 non-null  float64
 13  ship-city           128941 non-null  object 
 14  ship-state          128941 non-null  object 
 15  ship-postal-code    128941 non-null  float64
 16  ship-country        128941 non-null  object 
 17  B2B                 128976 non-null  bool   
 18  fulfilled-by        39263 non-null   object 
 19  New                 0 non-null       float64
 20  PendingS            0 non-null       float64
dtypes: bool(1), float64(4), int64(2), object(14)
memory usage: 19.8+ MB
```

Out[3]:

```
Index(['index', 'Order ID', 'Date', 'Status', 'Fulfilment', 'Sales Channel',
       'ship-service-level', 'Category', 'Size', 'Courier Status', 'Qty',
       'currency', 'Amount', 'ship-city', 'ship-state', 'ship-postal-code',
       'ship-country', 'B2B', 'fulfilled-by'],
      dtype='object')
```

In [5]:

```python
# Import pandas
import pandas as pd

# Load the dataset
file_path = '/Users/parle/Documents/InnoByte Services INTERN/Amazon Sale Report.csv'
data = pd.read_csv(file_path)

# Display the first few rows to understand the structure
data.head()

# Get an overview of the dataset, including column names and data types
data.info()

# Check for missing values in each column
data.isnull().sum()

# Drop completely empty columns if they exist
columns_to_drop = ['New', 'PendingS']
existing_columns_to_drop = [col for col in columns_to_drop if col in data.columns]
data = data.drop(existing_columns_to_drop, axis=1)

# Verify columns have been removed
data.columns
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 128976 entries, 0 to 128975
Data columns (total 21 columns):
 #   Column              Non-Null Count   Dtype  
---  ------              --------------   -----  
 0   index               128976 non-null  int64  
 1   Order ID            128976 non-null  object 
 2   Date                128976 non-null  object 
 3   Status              128976 non-null  object 
 4   Fulfilment          128976 non-null  object 
 5   Sales Channel       128976 non-null  object 
 6   ship-service-level  128976 non-null  object 
 7   Category            128976 non-null  object 
 8   Size                128976 non-null  object 
 9   Courier Status      128976 non-null  object 
 10  Qty                 128976 non-null  int64  
 11  currency            121176 non-null  object 
 12  Amount              121176 non-null  float64
 13  ship-city           128941 non-null  object 
 14  ship-state          128941 non-null  object 
 15  ship-postal-code    128941 non-null  float64
 16  ship-country        128941 non-null  object 
 17  B2B                 128976 non-null  bool   
 18  fulfilled-by        39263 non-null   object 
 19  New                 0 non-null       float64
 20  PendingS            0 non-null       float64
dtypes: bool(1), float64(4), int64(2), object(14)
memory usage: 19.8+ MB
```

Out[5]:

```
Index(['index', 'Order ID', 'Date', 'Status', 'Fulfilment', 'Sales Channel',
       'ship-service-level', 'Category', 'Size', 'Courier Status', 'Qty',
       'currency', 'Amount', 'ship-city', 'ship-state', 'ship-postal-code',
       'ship-country', 'B2B', 'fulfilled-by'],
      dtype='object')
```

In [6]:

```python
# Rename columns to ensure consistency
data.columns = data.columns.str.strip().str.replace(' ', '_').str.lower()

# Verify updated column names
data.columns
```

Out[6]:

```
Index(['index', 'order_id', 'date', 'status', 'fulfilment', 'sales_channel',
       'ship-service-level', 'category', 'size', 'courier_status', 'qty',
       'currency', 'amount', 'ship-city', 'ship-state', 'ship-postal-code',
       'ship-country', 'b2b', 'fulfilled-by'],
      dtype='object')
```

In [9]:

```python
# Fill missing values for 'currency' with 'Unknown'
data['currency'] = data['currency'].fillna('Unknown')

# Fill missing values in numeric columns like 'amount' with the mean
data['amount'] = data['amount'].fillna(data['amount'].mean())

# Rename column and then fill
data = data.rename(columns={'ship-city': 'ship_city'})
data['ship_city'] = data['ship_city'].fillna('Unknown')

# Verify changes
data.isnull().sum()
```

Out[9]:

```
index                     0
order_id                  0
date                      0
status                    0
fulfilment                0
sales_channel             0
ship-service-level        0
category                  0
size                      0
courier_status            0
qty                       0
currency                  0
amount                    0
ship_city                 0
ship-state               35
ship-postal-code         35
ship-country             35
b2b                       0
fulfilled-by          89713
dtype: int64
```

In [11]:

```python
# Convert 'date' column to datetime
data['date'] = pd.to_datetime(data['date'], format='%m-%d-%y', errors='coerce')

# Convert 'ship-postal-code' to numeric (using correct column name with hyphen)
data['ship-postal-code'] = pd.to_numeric(data['ship-postal-code'], errors='coerce')

# Convert 'b2b' column to boolean if not already
data['b2b'] = data['b2b'].astype(bool)

# Verify data types
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 128976 entries, 0 to 128975
Data columns (total 19 columns):
 #   Column              Non-Null Count   Dtype         
---  ------              --------------   -----         
 0   index               128976 non-null  int64         
 1   order_id            128976 non-null  object        
 2   date                73867 non-null   datetime64[ns]
 3   status              128976 non-null  object        
 4   fulfilment          128976 non-null  object        
 5   sales_channel       128976 non-null  object        
 6   ship-service-level  128976 non-null  object        
 7   category            128976 non-null  object        
 8   size                128976 non-null  object        
 9   courier_status      128976 non-null  object        
 10  qty                 128976 non-null  int64         
 11  currency            128976 non-null  object        
 12  amount              128976 non-null  float64       
 13  ship_city           128976 non-null  object        
 14  ship-state          128941 non-null  object        
 15  ship-postal-code    128941 non-null  float64       
 16  ship-country        128941 non-null  object        
 17  b2b                 128976 non-null  bool          
 18  fulfilled-by        39263 non-null   object        
dtypes: bool(1), datetime64[ns](1), float64(2), int64(2), object(13)
memory usage: 17.8+ MB
```

In [12]:

```python
# Check for duplicates
data.duplicated().sum()

# Drop duplicate rows
data = data.drop_duplicates()

# Verify duplicates are removed
data.duplicated().sum()
```

Out[12]:

```
np.int64(0)
```

In [13]:

```python
# Example: Handle outliers in 'amount' column using IQR method
Q1 = data['amount'].quantile(0.25)
Q3 = data['amount'].quantile(0.75)
IQR = Q3 - Q1

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

# Remove outliers
data = data[(data['amount'] >= lower_bound) & (data['amount'] <= upper_bound)]

# Verify changes
data.describe()
```

Out[13]:

	index	date	qty	amount	ship-postal-code
count	124634.000000	71368	124634.000000	124634.000000	124602.000000
mean	64344.009387	2022-05-17 06:32:53.321376512	0.900204	623.039222	464354.019454
min	0.000000	2022-03-31 00:00:00	0.000000	0.000000	110001.000000
25%	32033.250000	2022-04-23 00:00:00	1.000000	455.000000	382481.000000
50%	64249.500000	2022-05-18 00:00:00	1.000000	625.000000	500034.000000
75%	96572.750000	2022-06-14 00:00:00	1.000000	759.000000	600026.000000
max	128974.000000	2022-06-29 00:00:00	15.000000	1238.000000	989898.000000
std	37232.319229	NaN	0.311697	235.415289	191002.518968

In [14]:

```python
# Standardize text columns (example: 'category')
data['category'] = data['category'].str.lower().str.strip()

# Ensure 'size' column has consistent formatting
data['size'] = data['size'].str.upper()

# Verify formatting
data.head()
```

Out[14]:

	index	order_id	date	status	fulfilment	sales_channel	ship-service-level	category	size	courier_status	qty	currency	amount	ship_city	ship-state	ship-postal-code	ship-country	b2b	fulfilled-by
0	0	405-8078784-5731545	2022-04-30	Cancelled	Merchant	Amazon.in	Standard	t-shirt	S	On the Way	0	INR	647.62	MUMBAI	MAHARASHTRA	400081.0	IN	False	Easy Ship
1	1	171-9198151-1101146	2022-04-30	Shipped - Delivered to Buyer	Merchant	Amazon.in	Standard	shirt	3XL	Shipped	1	INR	406.00	BENGALURU	KARNATAKA	560085.0	IN	False	Easy Ship
2	2	404-0687676-7273146	2022-04-30	Shipped	Amazon	Amazon.in	Expedited	shirt	XL	Shipped	1	INR	329.00	NAVI MUMBAI	MAHARASHTRA	410210.0	IN	True	NaN
3	3	403-9615377-8133951	2022-04-30	Cancelled	Merchant	Amazon.in	Standard	blazzer	L	On the Way	0	INR	753.33	PUDUCHERRY	PUDUCHERRY	605008.0	IN	False	Easy Ship
4	4	407-1069790-7240320	2022-04-30	Shipped	Amazon	Amazon.in	Expedited	trousers	3XL	Shipped	1	INR	574.00	CHENNAI	TAMIL NADU	600073.0	IN	False	NaN

In [15]:

```python
# Final overview of cleaned data
data.info()

# Summary statistics
data.describe()

# Display the first few rows
data.head()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 124634 entries, 0 to 128975
Data columns (total 19 columns):
 #   Column              Non-Null Count   Dtype         
---  ------              --------------   -----         
 0   index               124634 non-null  int64         
 1   order_id            124634 non-null  object        
 2   date                71368 non-null   datetime64[ns]
 3   status              124634 non-null  object        
 4   fulfilment          124634 non-null  object        
 5   sales_channel       124634 non-null  object        
 6   ship-service-level  124634 non-null  object        
 7   category            124634 non-null  object        
 8   size                124634 non-null  object        
 9   courier_status      124634 non-null  object        
 10  qty                 124634 non-null  int64         
 11  currency            124634 non-null  object        
 12  amount              124634 non-null  float64       
 13  ship_city           124634 non-null  object        
 14  ship-state          124602 non-null  object        
 15  ship-postal-code    124602 non-null  float64       
 16  ship-country        124602 non-null  object        
 17  b2b                 124634 non-null  bool          
 18  fulfilled-by        37972 non-null   object        
dtypes: bool(1), datetime64[ns](1), float64(2), int64(2), object(13)
memory usage: 18.2+ MB
```

Out[15]:

	index	order_id	date	status	fulfilment	sales_channel	ship-service-level	category	size	courier_status	qty	currency	amount	ship_city	ship-state	ship-postal-code	ship-country	b2b	fulfilled-by
0	0	405-8078784-5731545	2022-04-30	Cancelled	Merchant	Amazon.in	Standard	t-shirt	S	On the Way	0	INR	647.62	MUMBAI	MAHARASHTRA	400081.0	IN	False	Easy Ship
1	1	171-9198151-1101146	2022-04-30	Shipped - Delivered to Buyer	Merchant	Amazon.in	Standard	shirt	3XL	Shipped	1	INR	406.00	BENGALURU	KARNATAKA	560085.0	IN	False	Easy Ship
2	2	404-0687676-7273146	2022-04-30	Shipped	Amazon	Amazon.in	Expedited	shirt	XL	Shipped	1	INR	329.00	NAVI MUMBAI	MAHARASHTRA	410210.0	IN	True	NaN
3	3	403-9615377-8133951	2022-04-30	Cancelled	Merchant	Amazon.in	Standard	blazzer	L	On the Way	0	INR	753.33	PUDUCHERRY	PUDUCHERRY	605008.0	IN	False	Easy Ship
4	4	407-1069790-7240320	2022-04-30	Shipped	Amazon	Amazon.in	Expedited	trousers	3XL	Shipped	1	INR	574.00	CHENNAI	TAMIL NADU	600073.0	IN	False	NaN

In [17]:

```python
# Save the cleaned dataset to a CSV file
cleaned_file_path = 'Cleaned_Amazon_Sale_Report.csv'  # Save in the current working directory
data.to_csv(cleaned_file_path, index=False)

print(f"Cleaned dataset saved to: {cleaned_file_path}")
```

```
Cleaned dataset saved to: Cleaned_Amazon_Sale_Report.csv
```

In [18]:

```python
# First save your cleaned data
data.to_csv(cleaned_file_path, index=False)

# Then export to PDF
import os
notebook_path = 'Untitled-1.ipynb'  # your notebook name
pdf_path = 'Amazon_Sales_Analysis.pdf'  # desired PDF name

# Export to PDF
!jupyter nbconvert --to pdf {notebook_path} --output {pdf_path}

print(f"PDF saved as: {pdf_path}")
```

```
[NbConvertApp] WARNING | pattern 'Untitled-1.ipynb' matched no files
This application is used to convert notebook files (*.ipynb)
        to various other formats.

        WARNING: THE COMMANDLINE INTERFACE MAY CHANGE IN FUTURE RELEASES.

Options
=======
The options below are convenience aliases to configurable class-options,
as listed in the "Equivalent to" description-line of the aliases.
To see all configurable class-options for some <cmd>, use:
    <cmd> --help-all

--debug
    set log level to logging.DEBUG (maximize logging output)
    Equivalent to: [--Application.log_level=10]
--show-config
    Show the application's configuration (human-readable format)
    Equivalent to: [--Application.show_config=True]
--show-config-json
    Show the application's configuration (json format)
    Equivalent to: [--Application.show_config_json=True]
--generate-config
    generate default config file
    Equivalent to: [--JupyterApp.generate_config=True]
-y
    Answer yes to any questions instead of prompting.
    Equivalent to: [--JupyterApp.answer_yes=True]
--execute
    Execute the notebook prior to export.
    Equivalent to: [--ExecutePreprocessor.enabled=True]
--allow-errors
    Continue notebook execution even if one of the cells throws an error and include the error message in the cell output (the default behaviour is to abort conversion). This flag is only relevant if '--execute' was specifie
d, too.
    Equivalent to: [--ExecutePreprocessor.allow_errors=True]
--stdin
    read a single notebook file from stdin. Write the resulting notebook with default basename 'notebook.*'
    Equivalent to: [--NbConvertApp.from_stdin=True]
--stdout
    Write notebook output to stdout instead of files.
    Equivalent to: [--NbConvertApp.writer_class=StdoutWriter]
--inplace
    Run nbconvert in place, overwriting the existing notebook (only
            relevant when converting to notebook format)
    Equivalent to: [--NbConvertApp.use_output_suffix=False --NbConvertApp.export_format=notebook --FilesWriter.build_directory=]
--clear-output
    Clear output of current file and save in place,
            overwriting the existing notebook.
    Equivalent to: [--NbConvertApp.use_output_suffix=False --NbConvertApp.export_format=notebook --FilesWriter.build_directory= --ClearOutputPreprocessor.enabled=True]
--coalesce-streams
    Coalesce consecutive stdout and stderr outputs into one stream (within each cell).
    Equivalent to: [--NbConvertApp.use_output_suffix=False --NbConvertApp.export_format=notebook --FilesWriter.build_directory= --CoalesceStreamsPreprocessor.enabled=True]
--no-prompt
    Exclude input and output prompts from converted document.
    Equivalent to: [--TemplateExporter.exclude_input_prompt=True --TemplateExporter.exclude_output_prompt=True]
--no-input
    Exclude input cells and output prompts from converted document.
            This mode is ideal for generating code-free reports.
    Equivalent to: [--TemplateExporter.exclude_output_prompt=True --TemplateExporter.exclude_input=True --TemplateExporter.exclude_input_prompt=True]
--allow-chromium-download
    Whether to allow downloading chromium if no suitable version is found on the system.
    Equivalent to: [--WebPDFExporter.allow_chromium_download=True]
--disable-chromium-sandbox
    Disable chromium security sandbox when converting to PDF..
    Equivalent to: [--WebPDFExporter.disable_sandbox=True]
--show-input
    Shows code input. This flag is only useful for dejavu users.
    Equivalent to: [--TemplateExporter.exclude_input=False]
--embed-images
    Embed the images as base64 dataurls in the output. This flag is only useful for the HTML/WebPDF/Slides exports.
    Equivalent to: [--HTMLExporter.embed_images=True]
--sanitize-html
    Whether the HTML in Markdown cells and cell outputs should be sanitized..
    Equivalent to: [--HTMLExporter.sanitize_html=True]
--log-level=<Enum>
    Set the log level by value or name.
    Choices: any of [0, 10, 20, 30, 40, 50, 'DEBUG', 'INFO', 'WARN', 'ERROR', 'CRITICAL']
    Default: 30
    Equivalent to: [--Application.log_level]
--config=<Unicode>
    Full path of a config file.
    Default: ''
    Equivalent to: [--JupyterApp.config_file]
--to=<Unicode>
    The export format to be used, either one of the built-in formats
            ['asciidoc', 'custom', 'html', 'latex', 'markdown', 'notebook', 'pdf', 'python', 'qtpdf', 'qtpng', 'rst', 'script', 'slides', 'webpdf']
            or a dotted object name that represents the import path for an
            ``Exporter`` class
    Default: ''
    Equivalent to: [--NbConvertApp.export_format]
--template=<Unicode>
    Name of the template to use
    Default: ''
    Equivalent to: [--TemplateExporter.template_name]
--template-file=<Unicode>
    Name of the template file to use
    Default: None
    Equivalent to: [--TemplateExporter.template_file]
--theme=<Unicode>
    Template specific theme(e.g. the name of a JupyterLab CSS theme distributed
    as prebuilt extension for the lab template)
    Default: 'light'
    Equivalent to: [--HTMLExporter.theme]
--sanitize_html=<Bool>
    Whether the HTML in Markdown cells and cell outputs should be sanitized.This
    should be set to True by nbviewer or similar tools.
    Default: False
    Equivalent to: [--HTMLExporter.sanitize_html]
--writer=<DottedObjectName>
    Writer class used to write the
                                        results of the conversion
    Default: 'FilesWriter'
    Equivalent to: [--NbConvertApp.writer_class]
--post=<DottedOrNone>
    PostProcessor class used to write the
                                        results of the conversion
    Default: ''
    Equivalent to: [--NbConvertApp.postprocessor_class]
--output=<Unicode>
    Overwrite base name use for output files.
                Supports pattern replacements '{notebook_name}'.
    Default: '{notebook_name}'
    Equivalent to: [--NbConvertApp.output_base]
--output-dir=<Unicode>
    Directory to write output(s) to. Defaults
                                  to output to the directory of each notebook. To recover
                                  previous default behaviour (outputting to the current
                                  working directory) use . as the flag value.
    Default: ''
    Equivalent to: [--FilesWriter.build_directory]
--reveal-prefix=<Unicode>
    The URL prefix for reveal.js (version 3.x).
            This defaults to the reveal CDN, but can be any url pointing to a copy
            of reveal.js.
            For speaker notes to work, this must be a relative path to a local
            copy of reveal.js: e.g., "reveal.js".
            If a relative path is given, it must be a subdirectory of the
            current directory (from which the server is run).
            See the usage documentation
            (https://nbconvert.readthedocs.io/en/latest/usage.html#reveal-js-html-slideshow)
            for more details.
    Default: ''
    Equivalent to: [--SlidesExporter.reveal_url_prefix]
--nbformat=<Enum>
    The nbformat version to write.
            Use this to downgrade notebooks.
    Choices: any of [1, 2, 3, 4]
    Default: 4
    Equivalent to: [--NotebookExporter.nbformat_version]

Examples
--------

    The simplest way to use nbconvert is

            > jupyter nbconvert mynotebook.ipynb --to html

            Options include ['asciidoc', 'custom', 'html', 'latex', 'markdown', 'notebook', 'pdf', 'python', 'qtpdf', 'qtpng', 'rst', 'script', 'slides', 'webpdf'].

            > jupyter nbconvert --to latex mynotebook.ipynb

            Both HTML and LaTeX support multiple output templates. LaTeX includes
            'base', 'article' and 'report'.  HTML includes 'basic', 'lab' and
            'classic'. You can specify the flavor of the format used.

            > jupyter nbconvert --to html --template lab mynotebook.ipynb

            You can also pipe the output to stdout, rather than a file

            > jupyter nbconvert mynotebook.ipynb --stdout

            PDF is generated via latex

            > jupyter nbconvert mynotebook.ipynb --to pdf

            You can get (and serve) a Reveal.js-powered slideshow

            > jupyter nbconvert myslides.ipynb --to slides --post serve

            Multiple notebooks can be given at the command line in a couple of
            different ways:

            > jupyter nbconvert notebook*.ipynb
            > jupyter nbconvert notebook1.ipynb notebook2.ipynb

            or you can specify the notebooks list in a config file, containing::

                c.NbConvertApp.notebooks = ["my_notebook.ipynb"]

            > jupyter nbconvert --config mycfg.py

To see all available configurables, use `--help-all`.

PDF saved as: Amazon_Sales_Analysis.pdf
```