ANDREW MADSON

COLLECT

WRANGLE

CLEAN

EXCEL - SQL - PYTHON





CLEANING CYCLE

COLLECT

FILTER

DUPLICATES

REPLACE VALUES

RENAME

DATA TYPES

MISSING VALUES

NORMALIZE

AGGREGATE

MERGE





COLLECT

PURPOSE: Acquire and load the data from various sources into your tool for analysis.

EXCEL	**Use Data tab > Get External Data**
SQL	**Use SELECT statement from a database**
PYTHON	**Use pandas.read_csv(), pandas.read_excel(), pandas.read_sql()**





FILTER

PURPOSE: Select a subset of your data based on certain conditions to focus on specific parts of the dataset.

EXCEL	**Use Sort & Filter option on Home tab**
SQL	**Use WHERE clause in SELECT statement**
PYTHON	**Use DataFrame.query() or DataFrame[DataFrame['col'] > val]**





FILTER - EXAMPLE

We want to filter out the data for those who are from 'Canada'.

ID	Name	Country	Age
1	Alice	USA	25
2	Bob	Canada	30
3	Charlie	UK	35
4	Dave	USA	40
5	Eve	Australia	45
6	Frank	Canada	50

ID	Name	Country	Age
1	Alice	USA	25
2	Bob	Canada	30





FILTER - EXAMPLE

EXCEL:

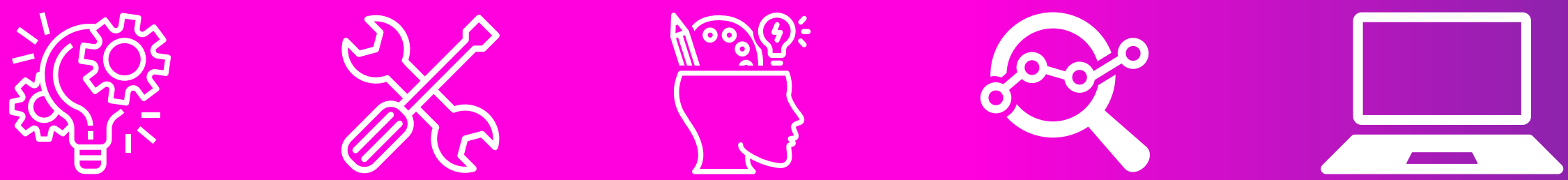
Filter option from the Sort & Filter section in the Home tab and select "Canada" under the Country column.

SQL:

```sql
SELECT * FROM dataset WHERE Country = 'Canada'
```

PYTHON:

```python
filtered_data = dataset[dataset['Country'] == 'Canada']
```



DUPLICATES

PURPOSE: Eliminate redundant data to ensure each data point is unique and meaningful for analysis.

EXCEL	**Use Remove Duplicates in Data tab**
SQL	**Use DISTINCT keyword in SELECT statement**
PYTHON	**Use DataFrame.drop_duplicates()**





DUPLICATES - EXAMPLE

We want to remove the duplicates for 'Alice' and 'Bob'.

ID	Name	Country
1	Alice	USA
2	Bob	Canada
3	Alice	USA
4	Dave	USA
5	Bob	Canada
6	Frank	Canada

ID	Name	Country
1	Alice	USA
2	Bob	Canada
4	Dave	USA
6	Frank	Canada

DUPLICATES - EXAMPLE

EXCEL:

Remove Duplicates option in the Data Tools section on the Data tab, selecting all columns to consider the whole row for duplication.

SQL:

```sql
SELECT DISTINCT * FROM dataset;
```

PYTHON:

```python
unique_data = dataset.drop_duplicates()
```





REPLACE

PURPOSE: Substitute specific values in your data to correct errors or standardize data entries.

EXCEL	**Use Find & Select > Replace on Home tab**
SQL	**Use UPDATE and SET commands**
PYTHON	**Use DataFrame.replace()**





REPLACE - EXAMPLE

we want to replace "Unknown" in the "Country" column with "Not Specified".

ID	Name	Country
1	Alice	USA
2	Bob	Canada
3	Alice	UK
4	Dave	USA
5	Bob	Unknown
6	Frank	Canada

ID	Name	Country
1	Alice	USA
2	Bob	Canada
3	Alice	UK
4	Dave	USA
5	Bob	Not Specified
6	Frank	Canada

REPLACE - EXAMPLE

EXCEL:

Find and Replace tool (Ctrl+H) to replace "Unknown" with "Not Specified".

SQL:

```sql
UPDATE dataset SET Country = 'Not Specified'
WHERE Country = 'Unknown';
```

PYTHON:

```python
dataset['Country'] =
dataset['Country'].replace('Unknown', 'Not
Specified')
```

RENAME

PURPOSE: Change the names of columns to something more meaningful or appropriate for analysis.

EXCEL	**Use Format > Rename on Home tab**
SQL	**Use AS keyword in SELECT statement**
PYTHON	**Use DataFrame.rename()**





RENAME - EXAMPLE

We want to rename the column "Country" to "Nationality".

ID	Name	Country
1	Alice	USA
2	Bob	Canada
3	Alice	UK
4	Dave	USA
5	Bob	Unknown
6	Frank	Canada

ID	Name	Nationality
1	Alice	USA
2	Bob	Canada
3	Alice	UK
4	Dave	USA
5	Bob	Unknown
6	Frank	Canada

RENAME - EXAMPLE

EXCEL:

Manually click on the header of the column and change its name.

SQL:

```sql
ALTER TABLE dataset RENAME COLUMN Country TO
Nationality;
```

PYTHON:

```python
dataset.rename(columns={'Country':
'Nationality'}, inplace=True)
```





DATA TYPE

PURPOSE: Convert data types to meet the requirements of specific calculations, operations, or functions.

EXCEL	**Use Text to Columns in Data tab**
SQL	**Use CAST or CONVERT function**
PYTHON	**Use DataFrame.astype()**





DATA TYPE - EXAMPLE

"Age" is currently an integer, but let's say we want to convert it to a string data type.

ID	Name	Age
1	Alice	25
2	Bob	30
3	Carol	35

ID	Name	Age
1	Alice	"25"
2	Bob	"30"
3	Carol	"35"

DATA TYPE - EXAMPLE

EXCEL:

Use the TEXT function:

1. **Create a new column next to "Age" and label it "Age (String)".**
2. **In the first cell under "Age (String)", write =TEXT(B2, "0") where B2 refers to the first cell in "Age" column.**
3. **Drag the bottom right corner of this cell down to apply this formula to the entire column.**

SQL:

```sql
SELECT ID, Name, CAST(Age AS VARCHAR) AS Age FROM
dataset;
```

PYTHON:

```python
dataset['Age'] = dataset['Age'].astype(str)
```

MISSING VALUES

PURPOSE: Deal with gaps or null values in your data by filling them with a specific value or removing the entries altogether.

EXCEL	**Use IFERROR function**
SQL	**Use ISNULL or COALESCE function**
PYTHON	**Use DataFrame.fillna() or DataFrame.dropna()**





MISSING - EXAMPLE

"Age" for Bob is missing. Replace missing values with a 0

ID	Name	Age
1	Alice	25
2	Bob	
3	Carol	35

ID	Name	Age
1	Alice	25
2	Bob	0
3	Carol	35

MISSING - EXAMPLE

EXCEL:

1. Create a new column next to "Age" and label it "Age (Cleaned)".
2. In the first cell under "Age (Cleaned)", write =IF(ISBLANK(B2), 0, B2) where B2 refers to the first cell in "Age" column. Here we replace the missing value with 0.
3. Drag the bottom right corner of this cell down to apply this formula to the entire column.

SQL:

```sql
SELECT ID, Name, COALESCE(Age, 0) AS Age FROM
dataset;
```

PYTHON:

```python
dataset['Age'] = dataset['Age'].fillna(0)
```

NORMALIZE

PURPOSE: Scale numerical data to standardize it, useful for certain statistical techniques and machine learning models.

EXCEL	**Use various functions like MIN, MAX**
SQL	**Use mathematical operations in SELECT**
PYTHON	**Use sklearn's preprocessing module**





NORMALIZE - EXAMPLE

Change the values of numeric columns in the dataset to use a common scale, without distorting differences in the ranges of values or losing information.

ID	Age	Salary
1	22	50000
2	35	60000
3	29	80000

ID	Age	Salary
1	0.00	0.00
2	0.86667	0.33333
3	0.46667	1.00

NORMALIZE - EXAMPLE

EXCEL & SQL

It's important to mention that Excel and SQL are not typically used for this process in the field of data analysis, as they lack the necessary built-in functionality. This process is generally completed using statistical software or programming languages such as Python or R, equipped with powerful libraries specifically designed for these tasks.

PYTHON:

```python
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

dataset[['Age', 'Salary']] =
scaler.fit_transform(dataset[['Age', 'Salary']])
```

AGGREGATE

PURPOSE: Summarize data by calculating aggregate metrics (like sum, average, count) over specific groups.

EXCEL	**Use PivotTable feature**
SQL	**Use GROUP BY along with aggregate functions like SUM, AVG**
PYTHON	**Use DataFrame.groupby().agg()**





AGGREGATE - EXAMPLE

We want to aggregate this data to understand the total sales of each product regardless of region

Product	Region	Sales
A	North	120
B	North	150
A	South	100
B	South	200
A	East	180
B	East	210
A	West	140
B	West	130

Product	Region	Sales
A	North	120
B	North	150

AGGREGATE - EXAMPLE

EXCEL:

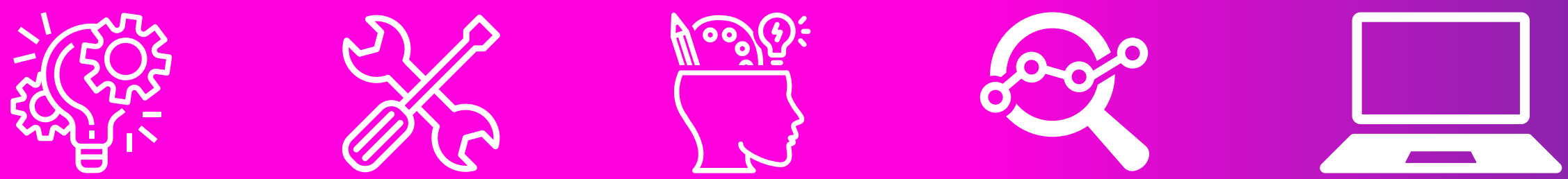
Excel: You can use Pivot Tables or the SUMIF function.

SQL:

```sql
SELECT Product, SUM(Sales)
FROM table_name
GROUP BY Product;
```

PYTHON:

```python
import pandas as pd
df = pd.DataFrame(table_name)
df.groupby('Product')['Sales'].sum()
```



MERGE

PURPOSE: Combine data from different sources or add new data to the existing dataset based on a common key.

EXCEL	**Use XLOOKUP or HLOOKUP function**
SQL	**Use JOIN clause**
PYTHON	**Use pandas.merge() or DataFrame.join()**





MERGE - EXAMPLE

We want to merge these tables to have a comprehensive table that shows each book with its author and the author's nationality

Book_ID	Title	Author_ID
1	"Moby Dick"	A1
2	"War and Peace"	A2
3	"1984"	A3

Author_ID	Author_Name	Nationality
A1	"Herman Melville"	American
A2	"Leo Tolstoy"	Russian
A3	"George Orwell"	British

MERGE - EXAMPLE

EXCEL:

You can use XLOOKUP or INDEX & MATCH functions.

SQL:

```sql
SELECT Books.Title, Authors.Author_Name,
Authors.Nationality
FROM Books
JOIN Authors
ON Books.Author_ID = Authors.Author_ID;
```

PYTHON:

```python
import pandas as pd
df_books = pd.DataFrame(Books)
df_authors = pd.DataFrame(Authors)
df_merged = pd.merge(df_books, df_authors,
on='Author_ID')
```

YOU DID IT!

HOORAY!

Save this post, and tag me as you develop these data analytics core skills.

HAPPY LEARNING!

ANDREW MADSON