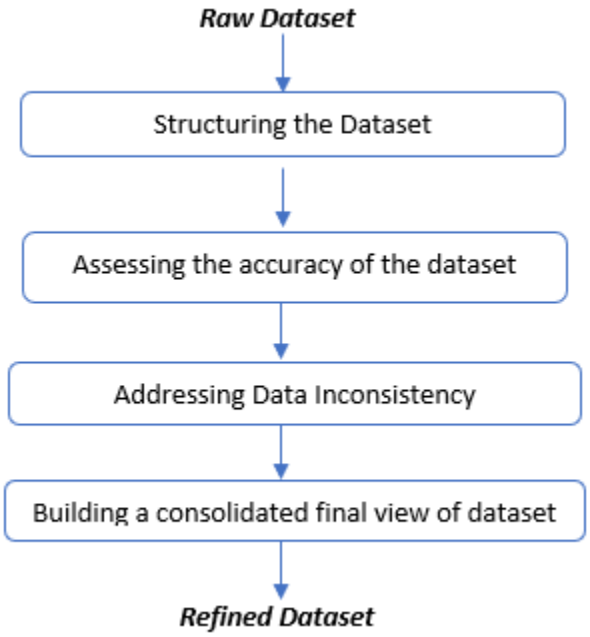
1. Describe your design on how you would onboard the dataset



Structuring the dataset:

- Data can be in different formats such as APIs, JSON files, or weblogs, which come in machine language or complex hierarchy and are therefore unreadable by humans.
- Data like this must first be put into readable form so the assessment can begin.
- Finally, converting into CSV format for further processing

Assessing the accuracy of the dataset:

- Identifying a dataset's flaws and reviewing for accuracy is critical to effective data onboarding.
- Missing values, invalid or mismatched data, and value distribution anomalies are all flaws that must be fixed in the data.

Addressing Data Inconsistency:

- Depending on a company's business needs, specific formatting and standardization, as well as a cross-reference, is needed.

Building a consolidated final view of dataset:

- Once the data is cleaned, using Excel or Access data can be combined in a consistent view.
- Using tools perform operation operations such as joining datasets as an array of functions, pivot and unpivot data, aggregate values, and derive key indicators.

2. Describe what tools (XSV, Python) will be used for data cleanup

Data cleaning or cleansing is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set, table, or database and refers to identifying incomplete, incorrect, inaccurate, or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data.

Some of the anomalies in the dataset is as follows:

1.Missing Data
2.Irregular Data
3.Unnecessary Data – Repetitive data, duplicate data
4.Inconsistant Data – Capitalization, Addresses and more

In python we have been provided with libraries like NumPy and pandas to handle data, where in which we can make use of such libraries to manipulate and clean the data.

Functions that we can perform is as follows.

1. Dropping inappropriate columns in a Data Frame.
2. Changing Index of the Data Frame.
3. Changing the data type of the Columns.
4. Combining String methods to modify the data present in the column
5. Renaming columns and skipping rows
6. Replacing Null columns with Appropriate values

xsv is a command line program for indexing, slicing, analyzing, splitting, and joining CSV files. Commands should be simple, fast and composable:

1. Simple tasks should be easy.
2. Performance tradeoffs should be exposed in the CLI interface.
3. Composition should not come at the expense of performance.

We basically can make use of commands to clean and modify the data.

- **cat** - Concatenate CSV files by row or by column.
- **count** - Count the rows in a CSV file. (Instantaneous with an index.)
- **fixlengths** - Force a CSV file to have same-length records by either padding or truncating them.
- **flatten** - A flattened view of CSV records. Useful for viewing one record at a time.
  e.g., xsv slice -i 5 data.csv | xsv flatten.
- **fmt** - Reformat CSV data with different delimiters, record terminators or quoting rules. (Supports ASCII delimited data.)

- **frequency** - Build frequency tables of each column in CSV data. (Uses parallelism to go faster if an index is present.)
- **headers** - Show the headers of CSV data. Or show the intersection of all headers between many CSV files.
- **index** - Create an index for a CSV file. This is very quick and provides constant time indexing into the CSV file.
- **input** - Read CSV data with exotic quoting/escaping rules.
- **join** - Inner, outer and cross joins. Uses a simple hash index to make it fast.
- **partition** - Partition CSV data based on a column value.
- **sample** - Randomly draw rows from CSV data using reservoir sampling (i.e., use memory proportional to the size of the sample).
- **reverse** - Reverse order of rows in CSV data.
- **search** - Run a regex over CSV data. Applies the regex to each field individually and shows only matching rows.
- **select** - Select or re-order columns from CSV data.
- **slice** - Slice rows from any part of a CSV file. When an index is present, this only has to parse the rows in the slice (instead of all rows leading up to the start of the slice).
- **sort** - Sort CSV data.
- **split** - Split one CSV file into many CSV files of N chunks.
- **stats** - Show basic types and statistics of each column in the CSV file. (i.e., mean, standard deviation, median, range, etc.)
- **table** - Show aligned output of any CSV data using elastic tabstops.