DataEng S24: Data Validation Activity

High quality data is crucial for any data project. This week you'll gain experience with validating a real data set provided by the Oregon Department of Transportation.

Due: this Friday at 10pm PT
Submit: Make a copy of this document and use it to record your results. Store a PDF copy of the document in your git repository along with any needed code before submitting using the in-class activity submission form.

A. [MUST] Initial Discussion Question

Discuss the following question among your working group members at the beginning of the week and place your own response(s) in this space. Or, if you have no such experience with invalid data then indicate this in the space below.

Have you ever worked with a set of data that included errors? Describe the situation, including how you discovered the errors and what you did about them.

Response:
While working with Oracle EBS data, I encountered errors during extraction. Conducted investigation, collaborated for resolution, and documented findings for future reference.

Background

The data set for this week is a listing of all Oregon automobile crashes on the Mt. Hood Hwy (Highway 26) during 2019. This data is provided by the Oregon Department of Transportation and is part of a larger data set that is often utilized for studies of roads, traffic and safety.

Here is the available documentation for this data: description of columns, Oregon Crash Data Coding Manual

Data validation is usually an iterative multi-step process.

B. Create assertions about the data
C. Write code to evaluate your assertions.
D. Run the code, analyze the results
E. Write code to transform the data and resolve any validation errors

B. [MUST] Create Assertions

Access the crash data, review the associated documentation of the data (ignore the data itself for now). Based on the documentation, create English language assertions for various properties of the data. No need to be exhaustive. Develop one or two assertions in each of the following categories during your first iteration through the ABC process.

1. ***existence* assertions.** Example: “Every crash occurred on a date”
    - **Response:**
      For each Crash ID in the dataset, there must exist at least one distinct Vehicle ID associated with it.
2. ***limit* assertions**. Example: “Every crash occurred during year 2019”
   Response:
    - For each crash ID where Record Type is 1, the corresponding Week Day Code must fall within the range of 1 to 7
    - Road Control must be in 02C and 02B
    - Median type must be in the range of 0 to 2 or blank
3. *intra-record* assertions. Example: “If a crash record has a latitude coordinate then it should also have a longitude coordinate”
   Response:
    - The Crash Month, Crash Day, and Crash Year columns must contain valid values representing a valid date within each record. Specifically, Crash Month should range from 1 to 12, Crash Day should range from 1 to 31 based on the month, and Crash Year should fall within a reasonable range, such as from 1900 to the current year.
    - When the record as longitude and latitude, it must have the impact location code with value less than or equal to14 or blank
4. **Create 2+ *inter-record check* assertions**. Example: “Every vehicle listed in the crash data was part of a known crash”
    - **Response:**
      Participant-vehicle associations within the crash dataset must be accurate and complete
    - Each vehicle involved in a crash has a corresponding Driver License Status.

5. Create 2+ *summary* assertions. Example: “There were thousands of crashes but not millions”
   Response:

    - Crash ID is unique across all the records
    - For every crash, the number of participants must be more than 1 for every crash
6. **Create 2+ *statistical distribution assertions***. Example: “crashes are evenly/uniformly distributed throughout the months of the year.”
   Response:
    - The distribution of crashes by hour of day follows a bimodal distribution, with peaks in the morning and evening rush hours.
    - Crashes are evenly/uniformly distributed throughout the months of the year.

These are just examples. You may use these examples, but you should also create new ones of your own.

C. [MUST] Validate the Assertions

1. Study the data in an editor or browser. Study it carefully, this data set is non-intuitive!.
2. Write python code to read in the test data. You are free to write your code any way you like, but we suggest that you use pandas’ methods for reading csv files into a pandas Dataframe.
3. Write python code to validate each of the assertions that you created in part A. The pandas package eases the task of creating data validation code.
4. If needed, update your assertions or create new assertions based on your analysis of the data.

D. [MUST] Run Your Code and Analyze the Results

In this space, list any assertion violations that you encountered:
- noticed there are many records with median type non numeric and more the 2 (checked in the document as the median type will be between 0 to 2)
- Impact Location should be numeric value in the range of 0 to 14, there are over 1500 records which are non numeric

For each assertion violation, describe how to resolve the violation. Options might include:
- revise assumptions/assertions
- discard the violating row(s)
- Ignore
- add missing values
- Interpolate
- use defaults
- abandon the project because the data has too many problems and is unusable

No need to write code to resolve the violations at this point, you will do that in step E.

> **Update the Dataset**: Once you have determined the appropriate action for each violating record, update the dataset accordingly. This may involve correcting impact location codes, removing records, or leaving impact location codes blank as needed.

E. [SHOULD] Resolve the Violations and Transform the Data

For each assertion violation write python code to resolve the violation according to your entry in the "how to resolve" section above.

Output the validated/transformed data to new files. There is no need to keep the same, awkward, single file format for the data. Consider outputting three files containing information about (respectively) crashes, vehicles and participants.

F. [ASPIRE] Learn and Iterate

The process of validating data usually gives us a better understanding of any data set. What have you learned about the data set that you did not know at the beginning of the current ABC iteration?

Next, iterate through the process again by going back through steps B, C, D and E at least one more time.