Neha Agrawal – My responses are inlined below in Bold

# DataEng: Data Validation Activity

Data validation is usually an iterative three-step process. First (part A) you develop assertions about your data as a way to make your assumptions explicit. Second (part B) you write code to evaluate the assertions and test the assumptions. This helps you to refine your existing assertions (part C) before starting the whole process over again by creating new assertions (part A again).

Submit: In-class Activity Submission Form

## A. 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 for this assignment, two or more assertions in each category are enough.

1. Create 2+ *existence* assertions. Example, “Every record has a date field”.

    - ***Every record has a NOT NULL crash ID field***
    - ***Every record has a NOT NULL Record Type field***

    **Revised Assertions:**

    ***CASE 1: Every record in Crash dataframe should have a valid not null crash id and serial number.***
    ***CASE 2: Every record in Vehicle dataframe should have a valid not null vehicle id***
    ***CASE 3: Every record in Participant dataframe should have a valid not null participant id***

2. Create 2+ *limit* assertions. The values of most numeric fields should fall within a valid range. Example: “the date field should be between 1/1/2019 and 12/31/2019 inclusive”

    - ***the date field should be between 1/1/2019 and 12/31/2019 inclusive for all the records with Record Type ==1***

- ***The crash hour should be between 00 and 23 inclusive and 99 for unknown time for all the records with Record Type ==1***
- ***The County Code that identifies the county in which the crash occurred should be in between 01 and 36 both inclusive for all the records with Record Type ==1***
- ***Latitude degrees – 41 to 46 inclusive, Longitude degrees : -116 to -124 inclusive***
- ***Latitude/Longitude Minutes – 0 to 59 inclusive***
- ***Latitude/Longitude Seconds – 0.00 to 59.99 inclusive***

**Revised Assertions:**

- ***the date field should be between 1/1/2019 and 12/31/2019 inclusive for all the records for CRASH dataframe***
- ***The crash hour should be between 00 and 23 inclusive and 99 for unknown time for all the records for CRASH dataframe***
- ***The County Code that identifies the county in which the crash occurred should be in between 01 and 36 both inclusive for all the records for CRASH dataframe***
- ***Latitude degrees – 41 to 46 inclusive, Longitude degrees : -116 to -124 inclusive for CRASH dataframe.***
- ***Latitude/Longitude Minutes – 0 to 59 inclusive for CRASH dataframe.***
- ***Latitude/Longitude Seconds – 0.00 to 59.99 inclusive for CRASH dataframe.***

3. Create 2+ *intra-record check* assertions.
    - **Total Count of Persons Involved: = Total Pedestrian Count + Total Pedalcyclist Count + Total Unknown Count + Total Occupant Count**
    - **Please check the referential integrity assertions**
4. Create 2+ *inter-record check* assertions.
    - Every crash should have a valid vehicle and crash participant involved
5. Create 2+ *summary* assertions. Example: “every crash has a unique ID”
    - **Every crash has a unique crash ID**
    - **Every crash has a unique Vehicle ID**
    - **Every crash has a unique Participant ID**

6. Create 2+ referential integrity insertions. Example “every crash participant has a Crash ID of a known crash”
    - ***every crash participant has a Crash ID of a known crash***
    - ***Every record with record type = 1 has a serial#***
    - ***Every record with record type = 2 has a vehicle ID field***

- ***Every record with record type = 3 should have both Vehicle ID and Participant ID field.***

7. Create 2+ *statistical distribution assertions*. Example: “crashes are evenly/uniformly distributed throughout the year.”
    - **Crashes are uniformly distributed throughout the year.**
    - **Crashes are more likely to occur in nice/dry weather when people prefer to be out or during the icy climate when road surface becomes too slippery.**

## B. Validate the Assertions

1. Now study the data in an editor or browser. If you are anything like me you will be surprised with what you find. The Oregon DOT made a mess with their data!

    ***Totally agreed! Consistent blank spaces which did not made any sense at all. But for the first week kept all the data in unnormalized form i.e. a single table/dataframe. Later in the second week of the in-class assignment, converted the ODOT data into 3 separate dataframes –CRASH which has unique crash id and serial num, VEHICLE which has unique vehicle id and PARTICIPANT which has unique participant id. Normalized data has its several benefits – it is easy to understand, removes redundancy, this way we removed the null valued fields and any duplicates.***

```
CrashesDF = df[df['Record Type'] == 1]
VehiclesDF = df[df['Record Type'] == 2]
ParticipantsDF = df[df['Record Type'] == 3]

CrashesDF = CrashesDF.dropna(axis=1,how='all')
VehiclesDF = VehiclesDF.dropna(axis=1,how='all')
ParticipantsDF = ParticipantsDF.dropna(axis=1,how='all')
```

    ***2 steps taken:***
    ***STEP 1: Created three views – CrashesDF, VehicleDF, ParticipantDF***
    ***STEP 2: Eliminated the columns with null values for each view,***

2. Write python code to read in the test data and parse it into python data structures. You can write your code any way you like, but we suggest that you use pandas’ methods for reading csv files into a pandas Data frame

3. Write python code to validate each of the assertions that you created in part A. Again, pandas make it easy to create and execute assertion validation code.

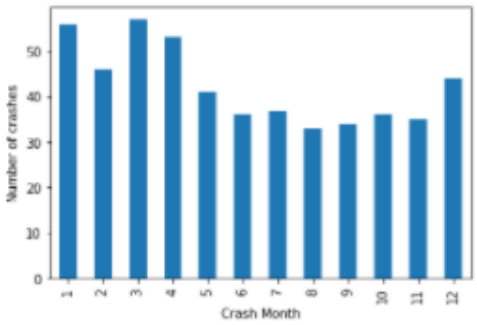
***First week of data validation code can be found here: [https://github.com/neha-psu/Data-Engg/blob/main/Week4-5/week4-dataValidation.py](https://github.com/neha-psu/Data-Engg/blob/main/Week4-5/week4-dataValidation.py)***

***GitHub link to the revised code with normalized data: [https://github.com/neha-psu/Data-Engg/blob/main/Week4-5/week5-dataValidation.py](https://github.com/neha-psu/Data-Engg/blob/main/Week4-5/week5-dataValidation.py)***

4. If you are like me you'll find that some of your assertions don't make sense once you actually understand the structure of the data. So go back and change your assertions if needed to make them sensible.

   **The output of the data validation of the revised assertions is shown below:**

```
-------------Crash DATAFRAME-------------


====================EXISTENCE VALIDATIONS================================

----- CASE 1: Every CRASH record should have not Null crash ID And Serial #
All the CRASH records passed Case 1 check!

====================LIMIT VALIDATIONS================================

----- CASE 2: Check the date field should be between 1/1/2019 and 12/31/2019 inclusive--------
All the CRASH records passed Case 2 check!

----- CASE 3: Check the crash hour should be between 00 and 23 inclusive and 99 for unknown time
All the CRASH records passed Case 3 check!
Time at which crash occured is unknown for 4 records

----- CASE 4: Check the County Code that identifies the county in which the crash occurred should be in between 01 and 36 b
oth inclusive.
All the CRASH records passed Case 4 check!

------CASE 5: Crashes are evenly/uniformly distributed throughout the year-------
```




```
------------Vehicle DATAFRAME-----------


====================EXISTENCE VALIDATIONS================================

----- CASE 6: Every record of VEHICLE DF has a vehicle ID field
All the records passed Case 6 check!

------------Participant DATAFRAME--------


====================EXISTENCE VALIDATIONS================================

------CASE 7: Every record of participant DF should have a NOT NULL Participant ID field.-------
All the records passed Case 7 check!
```

5. Run your code and note any assertion violations. List the violations here.

***I do not see any assertion voilation except one where I found 4 records with invalid crash_hour.***

```
----- CASE 3: Check the crash hour should be between 00 and 23 inclusive and 99 for unknown time
All the CRASH records passed Case 3 check!
Time at which crash occured is unknown for 4 records
```

## C. Evaluate the Violations

For any assertion violations found in part B, describe how you might resolve the violation. Options might include “revise assumptions/assertions”, “discard the violating row(s)”, “ignore”, “add missing values”, “interpolate”, “use defaults”, etc.

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

If you chose to “revise assumptions/assertions” for any of the violations, then briefly explain how you would revise your assertions based on what you learned.

**For the above assertion violation, after making a note of it, I’m choosing to ignore it.**

## D. 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 ABCD iteration?

Next, iterate through the process again by going back to Step A. Add more assertions in each of the categories before moving to steps B and C again. Go through the full loop twice before moving to step E.

## E. Resolve the Violations

For each assertion violation found during the two loops of the process, write python code to resolve the assertions. This might include dropping rows, dropping columns, adding default values, modifying values or other operations depending on the nature of the violation.

Note that I realize that this data set is somewhat awkward and that it might be best to “resolve the violations” by restructuring the data into proper tables. However, for this week, I ask that you keep the data in its current overall structure. Later (next week) we will have a chance to separate vehicle data and participant data properly.

## E. Retest

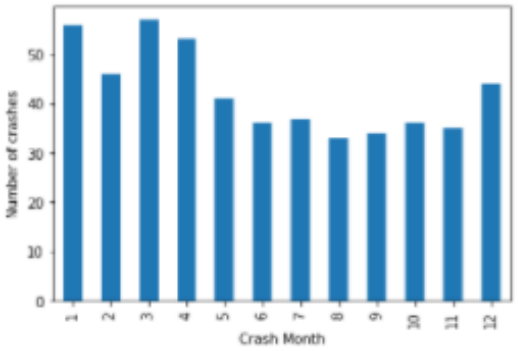
After modifying the dataset/stream to resolve the assertion violations you should have produced a new set of data. Run this data through your validation code (Step B) to make sure that it validates cleanly.
APPENDIX:

Output of data validation code after first week of in-class assignment:

```
====================EXISTENCE VALIDATIONS================================

----- CASE 1: Check if any record has NULL crash ID--------
All the records passed Case 1 check!

----- CASE 2: Check if any record has NULL record type--------
All the records passed Case 2 check!

====================LIMIT VALIDATIONS================================

----- CASE 3: Check the date field should be between 1/1/2019 and 12/31/2019 inclusive--------
All the CRASH records passed Case 3 check!

----- CASE 4: Check the crash hour should be between 00 and 23 inclusive and 99 for unknown time
All the CRASH records passed Case 4 check!
Time at which crash occured is unknown for 4 records

----- CASE 5: Check the County Code that identifies the county in which the crash occurred should be in between 01 and 36 bo
th inclusive.
All the CRASH records passed Case 5 check!

====================REFRENTIAL INTEGRITY VALIDATIONS================================

----- CASE 6: Every record with record type = 1 has a serial#
All the CRASH records passed Case 6 check!

----- CASE 7: Every record with record type = 2 has a vehicle ID field
All the records passed Case 7 check!

------CASE 8: Every record with record type = 3 should have both Vehicle ID and Participant ID field.-------
All the records passed Case 8 check!
```

Output of data validation code after second week of in-class assignment:

```
--------------Crash DATAFRAME-------------


=====================EXISTENCE VALIDATIONS================================

----- CASE 1: Every CRASH record should have not Null crash ID And Serial #
All the CRASH records passed Case 1 check!

=====================LIMIT VALIDATIONS================================

----- CASE 2: Check the date field should be between 1/1/2019 and 12/31/2019 inclusive--------
All the CRASH records passed Case 2 check!

----- CASE 3: Check the crash hour should be between 00 and 23 inclusive and 99 for unknown time
All the CRASH records passed Case 3 check!
Time at which crash occured is unknown for 4 records

----- CASE 4: Check the County Code that identifies the county in which the crash occurred should be in between 01 and 36 b
oth inclusive.
All the CRASH records passed Case 4 check!

------CASE 5: Crashes are evenly/uniformly distributed throughout the year-------
```




```
------------Vehicle DATAFRAME-----------


=====================EXISTENCE VALIDATIONS================================

----- CASE 6: Every record of VEHICLE DF has a vehicle ID field
All the records passed Case 6 check!

------------Participant DATAFRAME--------


=====================EXISTENCE VALIDATIONS================================

------CASE 7: Every record of participant DF should have a NOT NULL Participant ID field.-------
All the records passed Case 7 check!
```

Submit: [In-class Activity Submission Form]