Log of tested solutions

Project BCI Global Data Cleaning

Jeron Kamp, Antonella Portugal, Jakub Marcinek, Maxwell Agbana, Sophie Hu, Max Verwijmeren

Contents

[Create a document for keeping track of our Tested solution 3](#_Toc152580516)

[Test Great Expectations Library 4](#_Toc152580517)

[Test pydqc library 5](#_Toc152580518)

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task/Goal: | Context: (files/scripts) |
| Jeron | 20/11/2023 | Create a document for keeping track of our Tested solution | Log of Tested solutions: Github |
| Key points: | | | |
| * Format * Different kind of information: who, when, what, why. | | | |
| Conclusions: | | | |
| * Using this document will help us keeping track of all solution we have explored | | | |
| Sources: “Apa source” | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
| Antonella and Max | 13-11-2023 | Python Scripts for file Integrity Check | Integrity Check Parcel File+Sales Order Anonymized.ipynb  Product Master + Interfacility - Integrity checks.ipynb |
| Key points: | | | |
| * 2 Python Scripts that both do integrity checks for the different received data files such as:   + Share of negative values   + Report for textual and ID Columns   + Report for Numerical Columns | | | |
| Conclusions: | | | |
| * Both scripts are applicable to any future files. However They do Require user input to know which columns in the dataset might be “ID” columns instead of numerical for example | | | |
| Sources: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
| Antonella | 21-11-23 | Test Great Expectations Library | Test/Great Expectations test script.ipynb |
| Key points: | | | |
| * Tested Great Expectations Library on python notebook. * Didn’t figure out the use of batches * Didn’t figure out how to set expectations | | | |
| Conclusions: | | | |
| * We could re-explore this option if we think it could actually be useful for the client and if it is the simplest of the proposed solutions * It might be too big and complex of a solution for the kind of data we are using | | | |
| Sources:  <https://docs.greatexpectations.io/docs/> | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
| Antonella | 29-11-23 | Test pydqc library | File:test/pydqc\_test |
| Key points: | | | |
| * Generates Friendly Excel Files * The library is able to make plot of numerical columns * It can also make comparisons and compare consistency between tables * Requires a little human intervention for defining the column types in Excel | | | |
| Conclusions: | | | |
| * It’s good for summarizing data, but still haven’t figured out the consistency and comparison checks as it requires more than one table. | | | |
| Sources:  <https://github.com/SauceCat/pydqc/tree/master> | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
| Jeron | 29-11-23 | Test Soda Library |  |
| Key points: | | | |
| * Soda Core executes the checks to find invalid, missing, or unexpected data. * Manual selection * Works with SQL | | | |
| Conclusions: | | | |
| * Its fine for checking for missing data but is a lot of manual input when using different databases. Also is primarily focussed on missing data. | | | |
| Sources:  <https://github.com/sodadata/soda-core> | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
|  |  |  |  |
| Key points: | | | |
|  | | | |
| Conclusions: | | | |
|  | | | |
| Sources: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
|  |  |  |  |
| Key points: | | | |
|  | | | |
| Conclusions: | | | |
|  | | | |
| Sources: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
|  |  |  |  |
| Key points: | | | |
|  | | | |
| Conclusions: | | | |
|  | | | |
| Sources: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
|  |  |  |  |
| Key points: | | | |
|  | | | |
| Conclusions: | | | |
|  | | | |
| Sources: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
|  |  |  |  |
| Key points: | | | |
|  | | | |
| Conclusions: | | | |
|  | | | |
| Sources: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
|  |  |  |  |
| Key points: | | | |
|  | | | |
| Conclusions: | | | |
|  | | | |
| Sources: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
|  |  |  |  |
| Key points: | | | |
|  | | | |
| Conclusions: | | | |
|  | | | |
| Sources: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
|  |  |  |  |
| Key points: | | | |
|  | | | |
| Conclusions: | | | |
|  | | | |
| Sources: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
|  |  |  |  |
| Key points: | | | |
|  | | | |
| Conclusions: | | | |
|  | | | |
| Sources: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
|  |  |  |  |
| Key points: | | | |
|  | | | |
| Conclusions: | | | |
|  | | | |
| Sources: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
|  |  |  |  |
| Key points: | | | |
|  | | | |
| Conclusions: | | | |
|  | | | |
| Sources: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
|  |  |  |  |
| Key points: | | | |
|  | | | |
| Conclusions: | | | |
|  | | | |
| Sources: | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Executed by: | Date: | Task: | Context: (files/scripts) |
|  |  |  |  |
| Key points: | | | |
|  | | | |
| Conclusions: | | | |
|  | | | |
| Sources: | | | |