

# Coffee Quality Institute: Q Certified Arabica Coffees

CS 513: Theory and Practice of Data Cleaning, Summer 2019

*Elmar H. Langholz ([elmarhl2@illinois.edu](mailto:elmarhl2@illinois.edu))*

*July 21 2019*

## Contents

Introduction and Overview . . . . .	2
Initial assessment of the dataset . . . . .	3
Coffee information . . . . .	3
Grades/Cupping scores . . . . .	4
Green analysis . . . . .	5
Data Cleaning methods and process . . . . .	5
Data cleaning with OpenRefine . . . . .	5
Coffee information . . . . .	6
Grades/Cupping scores . . . . .	6
Green analysis . . . . .	6
Data cleaning with R . . . . .	6
Coffee information . . . . .	6
Grades/Cupping scores . . . . .	6
Green analysis . . . . .	6
Relational schema and integrity constraint checks . . . . .	6
Coffee completion date should be after creation . . . . .	6
Coffee altitude should be larger or equal than zero . . . . .	7
Coffee cupping scores/grades should be between zero and ten . . . . .	7
Coffee cupping scores/grades should be have approximate total scores . . . . .	7
Coffee cupping scores/grades should correctly report passing vs. failing . . . . .	8
Coffee green analysis count should be larger than zero . . . . .	8
Coffee green analysis category totals should equal . . . . .	8
Coffee green analysis category defects should equal . . . . .	8
Every coffee entry should have a at least one set of grades/cupping scores and green analysis . . . . .	8
Every coffee entry should have at most one set of grades/cupping scores and green analysis . . . . .	9
Data Cleaning Results . . . . .	10
Conceptual workflows . . . . .	10
Overview diagram . . . . .	10
Retrieve, save and normalize coffee data diagrams . . . . .	11
Clean data with OpenRefine diagrams . . . . .	12
Clean data with R diagrams . . . . .	14
Clean constraint violations diagrams . . . . .	16
OpenRefine workflows . . . . .	17
Coffee information diagram . . . . .	18
Grades/Cupping scores diagram . . . . .	19
Green analysis diagram . . . . .	20
Changes summary . . . . .	21
By type and stages . . . . .	21
Totals . . . . .	21
Conclusions and Future Work . . . . .	22
Appendix . . . . .	23

Retrieve data . . . . .	23
Normalize data . . . . .	24
Save data . . . . .	25
Coffee information altitude normalization . . . . .	25
Coffee information altitude unit standardization . . . . .	25
Coffee information weight unit standardization . . . . .	25
Coffee information green analysis pass normalization . . . . .	25
Coffee grades/cupping scores pass conversion . . . . .	26
Setup clean data . . . . .	26
Save clean data . . . . .	26
Setup and open relational database . . . . .	26
Write data and list tables . . . . .	26
Coffee completion integrity constraint . . . . .	26
Coffee altitude integrity constraint . . . . .	27
Coffee grades integrity constraint . . . . .	27
Coffee grades total integrity constraint . . . . .	27
Coffee grades pass integrity constraint . . . . .	28
Coffee green analysis values integrity constraint . . . . .	28
Coffee green analysis totals . . . . .	28
Coffee green analysis defects . . . . .	29
Coffee at least one grade or greens integrity constraint . . . . .	29
Coffee at most one grades or greens integrity constraint . . . . .	29
Setup final clean data without constaint violation entries . . . . .	30
Remove integrity constraint violation entries . . . . .	30
Save final clean data . . . . .	30
Create OpenRefine YesWorkflow's . . . . .	31
Create OpenRefine diagrams . . . . .	31
Calculate CSV difference summaries . . . . .	32
Calculate total CSV difference summaries . . . . .	33
Create total CSV difference files . . . . .	34

---

## Introduction and Overview

The Coffee Quality Institute<sup>1</sup> (CQI) is an international non-profit organization dedicated to improving the quality of coffee. As such, they provide tools and support for people that lack the understanding on how to measure the quality of the coffee they produce. They have compiled and shared their database<sup>2</sup> of Q graded arabica<sup>3</sup> and robusta<sup>4</sup> coffees which contain details about the producer, its farm, the coffee cupping scores and green coffee analysis (GCA).

As a use case, assume we want to open up our coffee sales business from wholesale to retail. In order to do this we want to leverage the data from CQI, our primary coffee provider, to implement a feature in our platform to recommend coffee. Since our target market consists of specialty coffee enthusiasts, roasters and baristas we are able to construct an advanced coffee profile with their preferences. For this to work, we must clean the CQI data and then leverage it to build a recommender system that will proactively perform these recommendations over time.

---

<sup>1</sup>Coffee Quality Institute. In *Coffee Quality Institute*. Retrieved May 27, 2019 from <https://www.coffeeinstitute.org/>

<sup>2</sup>Arabica Coffees - Q Coffee. In *Coffee Quality Institute*. Retrieved May 27, 2019 from <https://database.coffeeinstitute.org/coffees/arabica>

<sup>3</sup>Coffea arabica. (2019, May 15). In *Wikipedia, The Free Encyclopedia*. Retrieved May 27, 2019 from [https://en.wikipedia.org/wiki/Coffea\\_arabica](https://en.wikipedia.org/wiki/Coffea_arabica)

<sup>4</sup>Robusta coffee. (2019, May 16). <http://openrefine.org/http://openrefine.org/> Retrieved May 27, 2019 from [https://en.wikipedia.org/wiki/Robusta\\_coffee](https://en.wikipedia.org/wiki/Robusta_coffee)

Unlike with regular coffee which is normally dark roasted to the maximum level possible, in specialty coffee the quality matters a lot more since the flavors and notes are obtained through a light roast. However, depending on preference, one might find that one is willing to sacrifice different types of quality for price (e.g. you don't care as much about aroma but you want flavor and find the presence of slight insect issues or broken beans irrelevant).

In order to make this data fit for use, we will focus on accuracy, completeness and consistency of the measured coffee quality entries. Specifically, we would like to focus on the macro and micro-location details, coffee details and quality which consists of grading/cupping scores as well as green coffee analysis.

The data is sufficiently clean and granular enough that we can correlate the farm and coffee lot to its grading/cupping scores and green coffee analysis. It can also allow us to perform analysis at the farm, producer or regional level with the intent of uncovering trends at the harvest year. With enough historical data, it can help identify patterns and predict quality based on any of these parameters. Adding other sources of data (e.g. historical weather conditions, air moisture, etc...), can potentially improve and make these predictions even more accurate.

## Initial assessment of the dataset

The dataset was **acquired** using CQI's undocumented API, then it **normalized** and **saved**. At the time of obtaining this data there were only 3 robusta coffee records. Therefore, the focus was determined to fall fully on arabica coffees. In general the data is divided into three different areas and therefore files: coffee information, grades/cupping scores and green analysis.

### Coffee information

The coffee information subset consists of 104 observations and has the following structure:

```
Classes 'tbl_df', 'tbl' and 'data.frame':  104 obs. of  67 variables:
 $ species_short      : chr  "arabica" "arabica" "arabica" "arabica" ...
 $ species_title      : chr  "Arabica" "Arabica" "Arabica" "Arabica" ...
 $ origin_title       : chr  "Nicaragua" "Costa Rica" "Costa Rica" "Indonesia" ..
 $ stage_title        : chr  "Completed" "Completed" "Completed" "Completed" ...
 $ createdBy_name     : chr  "COMERCIAL INTERNACIONAL EXPORTADORA, S.A." "CECA"..
 $ createdBy_username : chr  "CISA" "CECA" "CECA" "PT-ROYAL-PACIFIC-INDAH-INTE"..
 $ country_title      : chr  NA NA NA NA ...
 $ completed_desc     : chr  "Jun 21st, 2018" "Jun 6th, 2018" "Jun 6th, 2018" "..
 $ created_desc       : chr  "May 10th, 2018" "May 17th, 2018" "May 17th, 2018"..
 $ icp_title          : chr  "Instituto Hondureño del Café" "Specialty Coffee"..
 $ icp_short          : chr  "ihcafe" "scacr" "scacr" "SCAI" ...
 $ grade_f            : num  83.5 81.8 83.6 78.5 78.3 ...
 $ pass_cert          : num  1 1 1 0 0 1 0 0 1 1 ...
 $ zip               : chr  NA "3095-1000" "3095-1000" "20352" ...
 $ weight_unit        : chr  "kg" "kg" "kg" "kg" ...
 $ weight            : num  69 69 69 1 69 69 1 69 69 60 ...
 $ viewable          : num  1 1 1 1 1 1 1 1 1 1 ...
 $ varietal           : num  5 6 6 14 4 4 0 0 4 37 ...
 $ url               : logi  NA NA NA NA NA NA ...
 $ thiscoffee        : num  NA 1 1 1 NA NA NA 1 NA 5 ...
 $ status            : num  1 1 1 1 1 1 1 1 1 1 ...
 $ state             : chr  "MANAGUA" "San José" "San José" "SUNGGAL" ...
 $ stage             : num  6 6 6 6 6 6 6 6 6 ...
 $ species           : num  1 1 1 1 1 1 1 1 1 1 ...
 $ ship             : logi  NA NA NA NA NA NA ...
```

```

$ region      : chr  "Jinotega" "Tarrazu" "Tarrazu" NA ...
$ reason      : num  NA 3 3 NA NA NA NA 3 NA 1 ...
$ random_id   : num  344617 674386 324816 875590 946016 ...
$ producer    : chr  "Rolando Lacayo" "Martin Gutierrez" "Martin Gutie"..
$ processing   : num  2 2 2 1 2 2 1 NA 2 2 ...
$ phone       : chr  "(505) 2255 9200" "(506)22024400" "(506)22024400"..
$ origin      : num  160 54 54 104 93 93 104 143 93 217 ...
$ offer       : chr  NA NA NA NA ...
$ name        : chr  "COMERCIAL INTERNACIONAL EXPORTADORA, S.A." "CECA"..
$ mill        : chr  "Beneficio San Carlos" "Beneficio Montañas del Di"..
$ lot         : chr  "829721 / K1820012" "9162" "9162-3" "1" ...
$ invoice     : num  44679 44889 44890 45012 45113 ...
$ image_5     : logi  NA NA NA NA NA NA ...
$ image_4     : logi  NA NA NA NA NA NA ...
$ image_3     : logi  NA NA NA NA NA NA ...
$ image_2     : logi  NA NA NA NA NA NA ...
$ image_1     : logi  NA NA NA NA NA NA ...
$ icp         : num  22 3 3 8 6 6 8 31 6 21 ...
$ ico         : chr  "017/001/1578" "5-025-0189" "5-025-0190" NA ...
$ iam         : num  NA 3 3 3 NA NA NA 6 NA 6 ...
$ harvest     : chr  "2018" "2018" "2018" "2018" ...
$ green_pass   : chr  "Y" "Y" "Y" "N" ...
$ grade_id    : num  496470 670071 825826 538985 992773 ...
$ grade       : num  83.5 81.8 83.6 78.5 78.3 ...
$ farm_long   : chr  NA NA NA NA ...
$ farm_lat    : chr  NA NA NA NA ...
$ farm        : chr  "Cafetales Santa Matilde" "Gamboa" "Gamboa" "ELMA"..
$ description  : chr  NA NA NA NA ...
$ createdBy   : num  11014 7765 7765 16482 1209 ...
$ created     : num  1.53e+09 1.53e+09 1.53e+09 1.53e+09 1.53e+09 ...
$ country     : num  NA NA NA NA NA NA NA NA NA NA ...
$ completed   : num  1.53e+09 1.53e+09 1.53e+09 1.53e+09 1.53e+09 ...
$ company     : chr  "COMERCIAL INTERNACIONAL EXPORTADORA, S.A." "CECA"..
$ city        : chr  "MANAGUA" "Montes de Oca" "Montes de Oca" "MEDAN" ..
$ carrier     : logi  NA NA NA NA NA NA ...
$ buyer      : chr  NA NA NA NA ...
$ bags        : num  275 245 80 2 275 50 2 250 275 50 ...
$ altitude_unit : chr  "m" "m" "m" "m" ...
$ altitude    : chr  "1100" "1850" "1850" NA ...
$ address     : chr  NA "De la entrada al Parqueo del Mall San Pedro, "..
$ DT_RowId    : num  344617 674386 324816 875590 946016 ...
$ row_index   : num  1 2 3 4 5 6 7 8 9 10 ...

```

## Grades/Cupping scores

The grades/cupping scores subset consists of 103 observations and has the following structure:

Classes 'tbl\_df', 'tbl' and 'data.frame': 103 obs. of 14 variables:

```

$ aroma      : num  7.75 7.75 7.5 7.42 7 ...
$ flavor     : num  7.75 7.5 7.58 7.58 6.75 ...
$ after      : num  7.5 7.25 7.33 7.17 6.75 ...
$ acidity    : num  7.58 7.33 7.67 7.58 7 ...
$ body       : num  7.83 7.33 7.58 7.33 6.92 ...
$ mouthfeel  : num  0 0 0 0 0 0 0 0 0 0 ...

```

```

$ balance      : num  7.5 7.25 7.58 7.33 6.92 ...
$ uniformity: num  10 10 10 10 10 10 10 10 10 10 ...
$ clean       : num  10 10 10 10 10 10 10 10 10 10 ...
$ sweet       : num  10 10 10 6.67 10 ...
$ overall     : num  7.58 7.42 8.33 7.42 7 ...
$ total       : num  83.5 81.8 83.6 78.5 78.3 ...
$ pass        : num   1 1 1 0 0 1 1 1 1 1 ...
$ random_id   : num  344617 674386 324816 875590 946016 ...

```

## Green analysis

The green analysis subset consists of 103 observations and has the following structure:

Classes 'tbl\_df', 'tbl' and 'data.frame': 103 obs. of 27 variables:

```

$ full_black      : num  0 0 0 0 0 0 0 0 0 0 ...
$ full_sour       : num  0 0 0 0 0 0 0 0 0 0 ...
$ dried_cherry    : num  0 0 0 0 0 0 0 0 0 0 ...
$ fungus          : num  0 0 0 0 0 0 0 0 0 0 ...
$ severe_insect   : num  2 0 0 5 0 0 1 0 0 0 ...
$ foreign_matter  : num  0 0 0 0 0 0 0 0 0 0 ...
$ partial_black   : num  5 0 0 1 5 0 0 0 2 0 ...
$ partial_sour    : num  1 0 0 12 0 0 0 1 0 0 ...
$ parchment       : num  0 0 0 0 0 0 0 0 0 0 ...
$ floater         : num  0 0 0 0 0 0 0 0 0 0 ...
$ immature        : num  0 3 2 32 0 1 26 26 0 0 ...
$ withered        : num  0 0 0 0 0 0 0 7 0 0 ...
$ shell           : num  15 2 2 0 4 3 0 1 8 0 ...
$ broken          : num  9 3 2 160 13 10 62 17 14 1 ...
$ hull            : num  0 0 0 0 0 0 0 0 0 0 ...
$ slight_insect   : num  3 0 10 20 1 0 3 10 0 0 ...
$ moisture        : num  11 11 11 14 10 10 14 11 11 11 ...
$ color           : num  4 3 4 4 4 4 4 4 4 4 ...
$ agtron          : num  0 0 0 0 0 0 0 0 0 0 ...
$ quakers         : num  0 0 0 5 2 1 5 0 2 0 ...
$ color_title     : chr  "Green" "Bluish-Green" "Green" "Green" ...
$ category_one_total : num  2 0 0 5 0 0 1 0 0 0 ...
$ category_two_total : num  33 8 16 225 23 14 91 62 24 1 ...
$ category_one_equivalent: num  0 0 0 1 0 0 0 0 0 0 ...
$ category_two_equivalent: num  5 0 1 44 3 2 17 10 3 0 ...
$ pass            : logi  TRUE TRUE TRUE FALSE TRUE TRUE ...
$ random_id       : num  344617 674386 324816 875590 946016 ...

```

## Data Cleaning methods and process

### Data cleaning with OpenRefine

The coffee information dataset has several issues which were addressed using OpenRefine<sup>5</sup>. The operation history has been stored in the supplemental information as JSON files (\*\_recipe.json) under the data/openrefine directory.

<sup>5</sup>openrefine.github.com. In *OpenRefine*. Retrieved June 3, 2019 from <http://openrefine.org/>

### Coffee information

Corresponding to text columns the following was addressed: spelling mistakes/typos, inconsistency between abbreviation vs. no abbreviation, text casing mismatch, and extra spaces (leading, trailing and in between). These issues were present in the following columns of interest: `country_title`, `state`, `region`, `producer`, `name`, `mill`, `farm`, `company`, `city`, `carrier` and `buyer`. Date columns `completed_desc` and `created_desc` were also standardized to use ISO. Across the board, `NA` and `None` are used to convey the same meaning and were standardized to use either empty or `None` (one or the other).

### Grades/Cupping scores

No major changes were performed on this subset of the data.

### Green analysis

No major changes were performed on this subset of the data.

### Data cleaning with R

More transformations were performed using R which did not fit well using OpenRefine.

### Coffee information

Some altitudes are ranges instead of units: **1500-2000**. Due to this, we **normalized** it by calculating the mid-point between the range, and converted the values to numeric. Two different types of altitude units were used: feet (*ft*) and meters (*m*). These were **standardized** by converting feet to meters. As a point in hand, two different types of weight units were used: pounds (*lbs*) and kilograms (*kg*). These were **standardized** by converting pounds to kilograms. Finally, the green analysis pass column `green_pass` was **normalized** from numeric to logical.

### Grades/Cupping scores

The `pass` column was **normalized** from numeric to logical.

### Green analysis

No major changes were performed on this subset of the data.

### Relational schema and integrity constraint checks

Using the clean data a SQLite database was **setup and opened** and each data subset **written to** a corresponding table with the same name. In order to address violation constraints, rows corresponding to these were **removed**.

### Coffee completion date should be after creation

In denial form, **we report all the integrity constraint violations** for coffee entries which completion date is before its creation date:

random_id	created	completed
532954	1530489600	1529971200
990730	1530489600	1529884800
856523	1530230400	1529884800
790329	1530489600	1529884800
127616	1530489600	1529884800
816803	1530489600	1529971200
401052	1530489600	1529971200
768955	1530489600	1529971200
111259	1543017600	1530662400
138073	1538697600	1534982400
858333	1547596800	1540944000
696491	1549670400	1548633600
995923	1549670400	1549497600
565006	1550707200	1549929600
761203	1558396800	1552694400

This could mean that the cupping score/grading was finished before the entry was created within the system. For this reason, we keep the entries.

#### Coffee altitude should be larger or equal than zero

In denial form, **we report all the integrity constraint violations** for coffee entries that have an altitude less or equal than zero:

random_id	altitude
-----------	----------

No issues were found.

#### Coffee cupping scores/grades should be between zero and ten

In denial form, **we report all the integrity constraint violations** for coffee entries that have grades less than zero or larger than ten:

random_id
-----------

No issues were found.

#### Coffee cupping scores/grades should be have approximate total scores

In denial form, **we report all the integrity constraint violations** for coffee grades that do not have a close enough total score:

random_id	actual_total	expected_total
-----------	--------------	----------------

No issues were found.

### Coffee cupping scores/grades should correctly report passing vs. failing

In denial form, **we report all the integrity constraint violations** for coffee grades that do not mark coffees with a score of larger than or equal to 80 as pass, otherwise fail:

random_id	total	pass
-----------	-------	------

No issues were found.

### Coffee green analysis count should be larger than zero

In denial form, **we report all the integrity constraint violations** for coffee green analysis that have grades less than zero:

random_id
-----------

No issues were found.

### Coffee green analysis category totals should equal

In denial form, **we report all the integrity constraint violations** for coffee green analysis whose totals per category do not match the manually calculated values:

random_id	one_total	category_one_total	two_total	category_two_total
995923	46	23	392	196

In this case the manually calculated total amounts (**one\_total** and **two\_total**) are double ( $2x$ ) the ones reported by the data set. This could be caused by a either an ETL process gone wrong or an incorrect data entry so it will be removed from the data set.

### Coffee green analysis category defects should equal

In denial form, **we report all the integrity constraint violations** for coffee green analysis whose calculated defects per category do not match the manually calculated ones:

random_id	one_defects	category_one_equivalent	two_defects	category_two_equivalent
995923	15	7	73	36

As in the case before for the same entry, we see that the manually calculated defect amounts (**one\_defects** and **two\_defects**) are larger by two times plus one ( $2x + 1$ ) the reported value by the data set. This could be caused by a either an ETL process gone wrong or an incorrect data entry so it will be removed from the data set.

### Every coffee entry should have a at least one set of grades/cupping scores and green analysis

In denial form, **we report all the integrity constraint violations** for coffee entries that do not exist in the grades/cupping scores and green analysis subset when they do within the coffee information:



### Grades/cupping scores

random_id	farm	mill	producer
742679	El Prado	Neiva	Franklin Dussan

### Green analysis

random_id	farm	mill	producer
742679	El Prado	Neiva	Franklin Dussan

In this case we are missing an entry in both. It could mean that it still has not been added and/or reviewed. These entries will be removed from the data set.

### Every coffee entry should have at most one set of grades/cupping scores and green analysis

In denial form, **we report all the integrity constraint violations** for coffee entries that have more than one entries for the grades/cupping scores or green analysis subset:

### Grades/cupping scores

random\_id

### Green analysis

random\_id

No issues were found.

## Data Cleaning Results

In order to understand the above process and summarize the data cleaning results, diagrams and changes summaries per data file were constructed.

### Conceptual workflows

A **retrieve and clean workflow** was manually created using YesWorkflow<sup>6</sup> and is provided as supplemental information as `workflow/coffee.yw`. Following are multiple visualization levels of it.

### Overview diagram

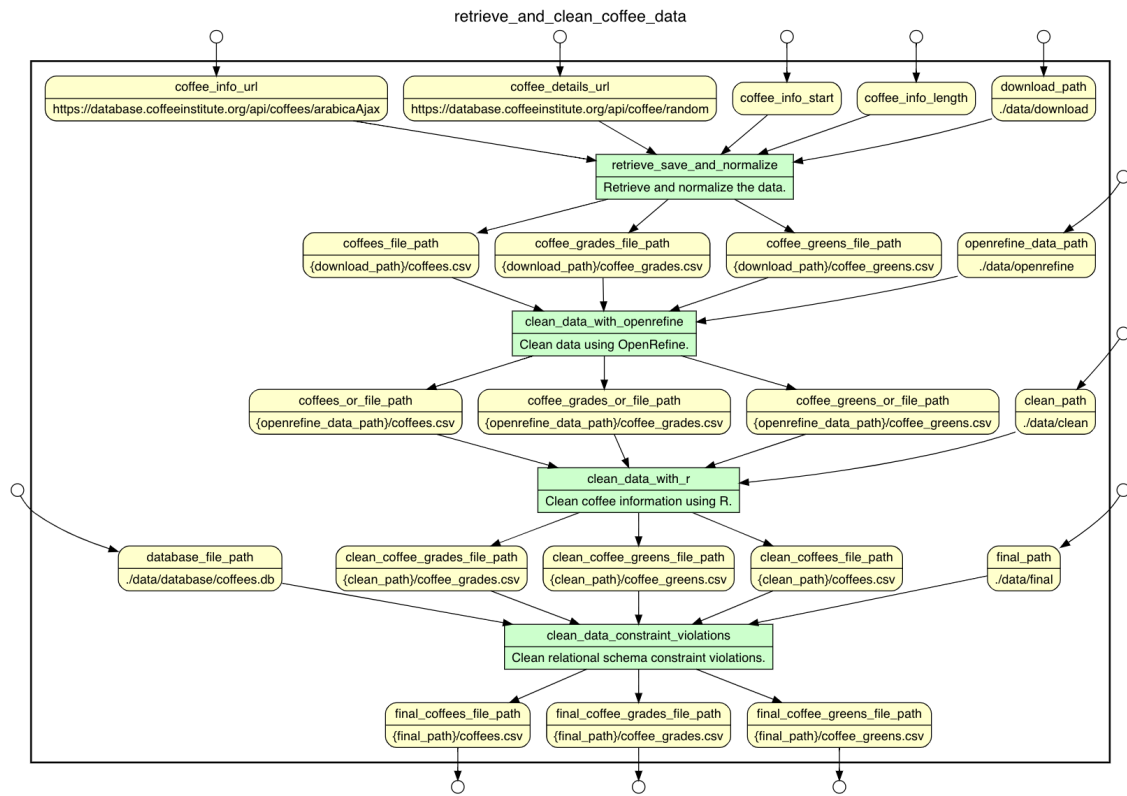


Figure 1: Level 0 - Retrieve and clean workflow

<sup>6</sup>YesWorkflow - GitHub. In *GitHub*. Retrieved June 6, 2019 from <https://github.com/yesworkflow-org/>

## Retrieve, save and normalize coffee data diagrams

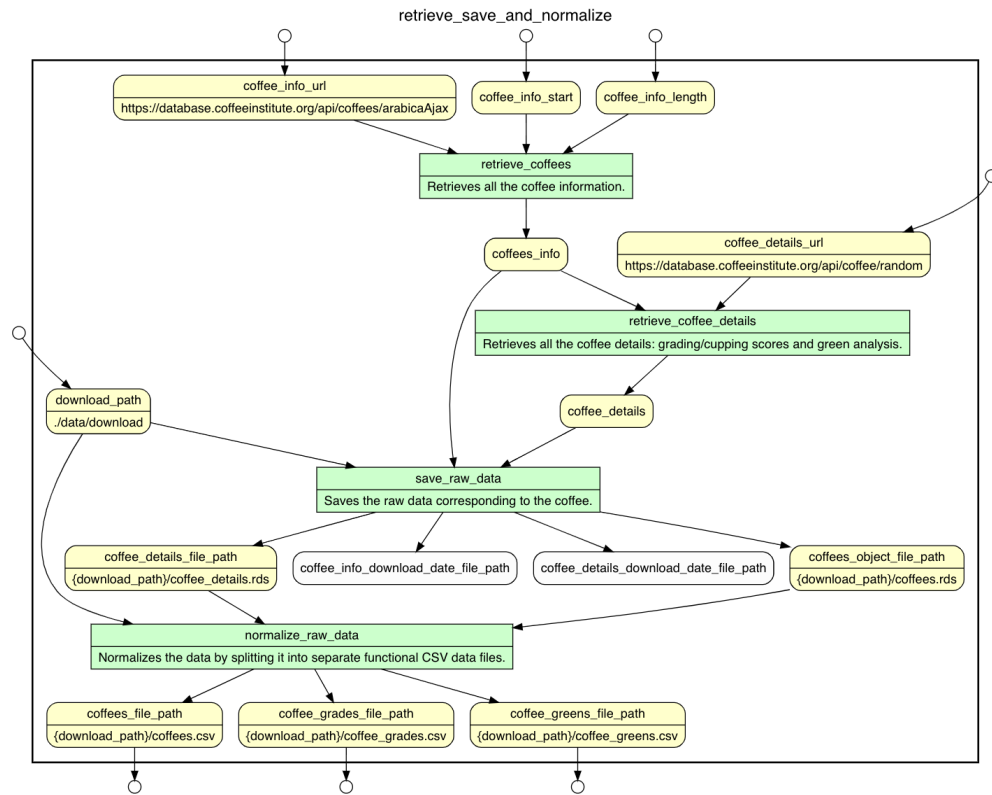


Figure 2: Level 1 - Retrieve, save and normalize coffee data

## Clean data with OpenRefine diagrams

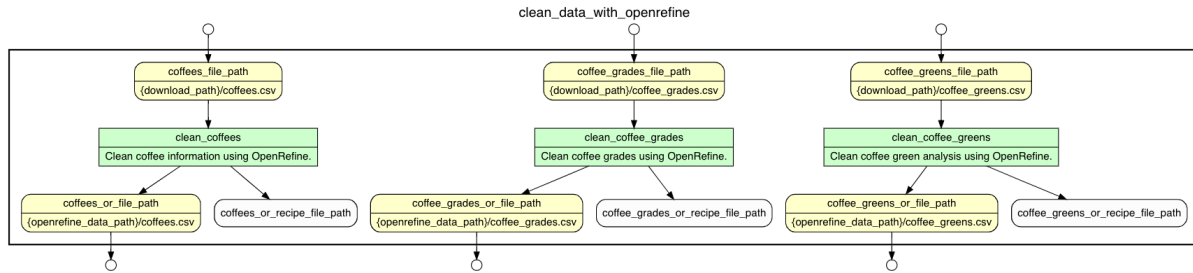


Figure 3: Level 1 - Clean data with OpenRefine



## Clean data with R diagrams

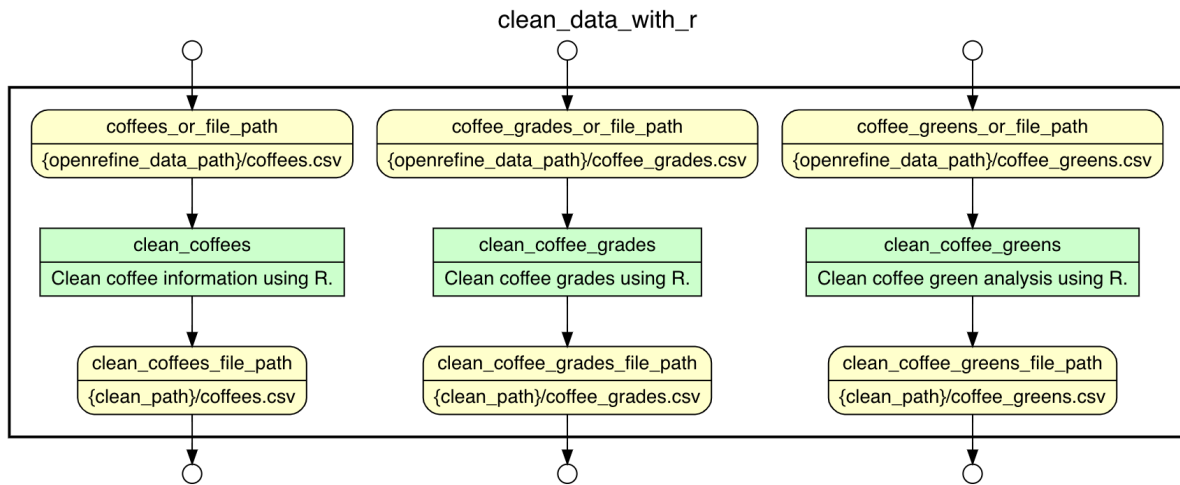


Figure 5: Level 1 - Clean data with R

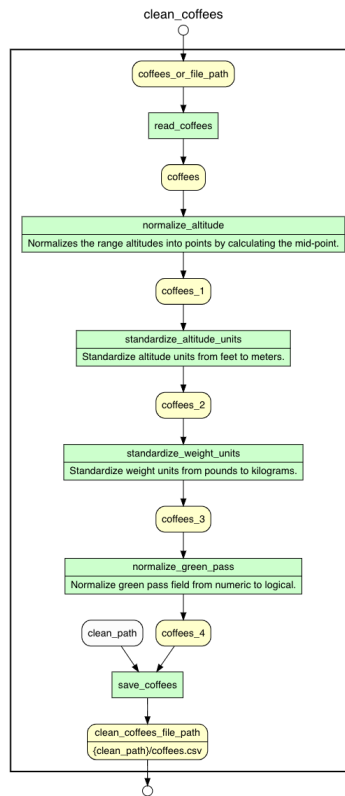


Figure 6: Level 2 - Clean coffee informatino with R

## Clean constraint violations diagrams

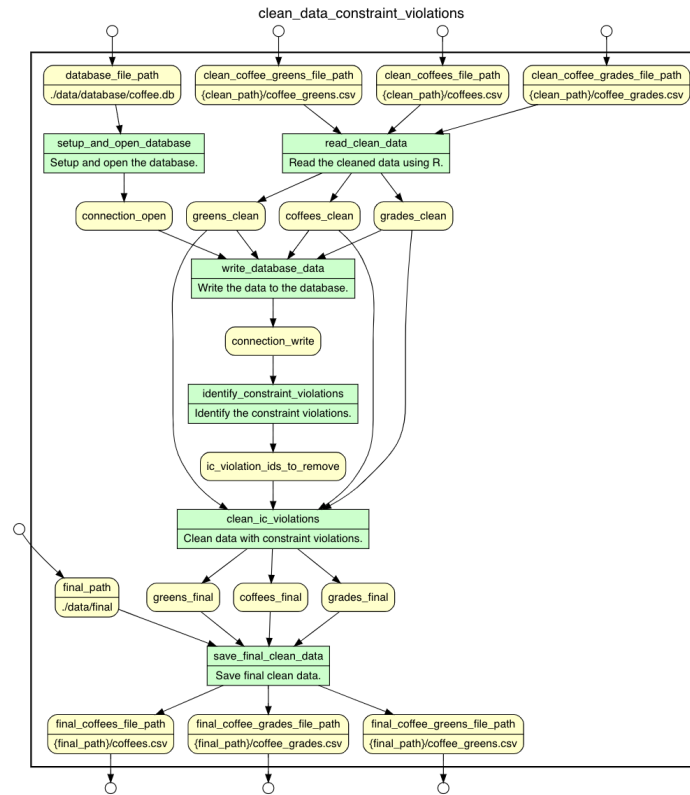


Figure 7: Level 1 - Clean constraint violation



## OpenRefine workflows

Using `or2yw`<sup>7</sup> three YesWorkflow files (included in the supplemental information under the `workflow/openrefine/*.yw` directory) were **automatically generated** for the data cleaning changes performed using OpenRefine. Following, the **corresponding diagrams** (one per each data file) are included.

---

<sup>7</sup>or2ywtool - PyPI. In *PyPI – the Python Package Index*. Retrieved July 17, 2019 from <https://pypi.org/project/or2ywtool/>

## Coffee information diagram



Figure 8: OpenRefine coffee information changes

### Grades/Cupping scores diagram

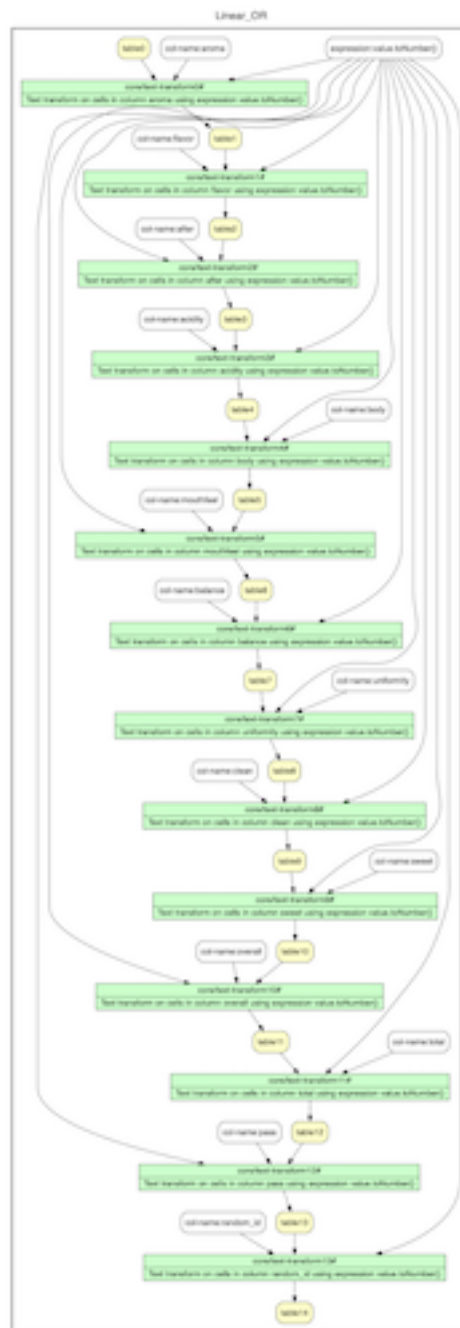


Figure 9: OpenRefine grades/cupping scores changes

## Green analysis diagram



Figure 10: OpenRefine green analysis changes

## Changes summary

Using `csvdiff`<sup>8</sup>, the changes performed in total and at each stage can be calculated per each file by the amount rows that were either removed, added and/or changed.

## By type and stages

The changes by type and stages can be summarized with the following tables.

### Coffee information

	Removed	Added	Changed
OpenRefine	0 rows (0.0%)	0 rows (0.0%)	104 rows (100.0%)
R	0 rows (0.0%)	0 rows (0.0%)	104 rows (100.0%)
Constraints	2 rows (1.9%)	0 rows (0.0%)	0 rows (0.0%)

### Grades/Cupping scores

	Removed	Added	Changed
OpenRefine	0 rows (0.0%)	0 rows (0.0%)	0 rows (0.0%)
R	0 rows (0.0%)	0 rows (0.0%)	0 rows (0.0%)
Constraints	1 rows (1.0%)	0 rows (0.0%)	0 rows (0.0%)

### Green analysis

	Removed	Added	Changed
OpenRefine	0 rows (0.0%)	0 rows (0.0%)	2 rows (1.9%)
R	0 rows (0.0%)	0 rows (0.0%)	2 rows (1.9%)
Constraints	1 rows (1.0%)	0 rows (0.0%)	0 rows (0.0%)

## Totals

The total changes between when the data was first downloaded and the final data cleaning changes can then be summarized as follows:

	Removed	Added	Changed
Coffee information	2 rows (1.9%)	0 rows (0.0%)	102 rows (98.1%)
Grades/Cupping scores	1 rows (1.0%)	0 rows (0.0%)	0 rows (0.0%)
Green analysis	1 rows (1.0%)	0 rows (0.0%)	0 rows (0.0%)

The difference files were also calculated and are stored in the supplemental information under the `data/diff` directory.

---

<sup>8</sup>csvdiff - PyPI. In *PyPI – the Python Package Index*. Retrieved July 14, 2019 from <https://pypi.org/project/csvdiff/>

## Conclusions and Future Work

From the [changes summary](#), we can observe that we ended up modifying every single row corresponding to the coffee information using OpenRefine and R. On the other hand, very few modifications were performed on the grades/cupping scores and green analysis data files. This makes sense taking into consideration that the coffee information data contained many string entries while the grades/cupping scores contained primarily numeric values. Overall, the amount of changes with respect to the original data are substantial.

One major problem encountered early on was eagerly updating new data from the source. Initially we ran the data retrieval process to acquire the data from the web site frequently (since it was fully automated) because we believed it changed infrequently. However, that was not the case and therefore it was hard to establish a data cleaning loop during development since we ended up having to compensate for the updates as well. For this reason we decided to take a snapshot of the data (including date/time) and work offline.

With more time, we would have reached out to CQI and try to resolve some of the issues encountered in the data set. To start with, we assumed that it is possible to have a [completion date which is before their creation date](#). This should be confirmed with them. Also, while all constraint violations were removed, some of these could have been fixed instead. Such is the case of [coffee greens total](#) and [coffee greens defect](#). Finally, it would have been great to further clean the data to obtain the coordinates for the producers and have a better sense of location of the farms. With it we could visually inspect the data in a more practical way.

## Appendix

### Retrieve data

```
create_date_marker_file <- function (file_path) {
  file_connection <- file(file_path)
  current_date <- strftime(now(), "%Y-%m-%dT%H:%M:%S%z")
  writeLines(current_date, file_connection)
  close(file_connection)
}

data_path <- "./data"
download_path <- file.path(data_path, "download")
coffees_object_file_path <- file.path(download_path, "coffees.rds")
coffee_details_object_file_path <- file.path(download_path, "coffee_details.rds")
coffees_file_path <- file.path(download_path, "coffees.csv")
coffee_grades_file_path <- file.path(download_path, "coffee_grades.csv")
coffee_greens_file_path <- file.path(download_path, "coffee_greens.csv")
coffee_date_file_path <- file.path(download_path, "coffees_date.txt")
coffee_details_date_file_path <- file.path(download_path, "coffee_details_date.txt")

data_frame_from_list <- function (data) {
  normalized_values <- lapply(data, function (row) {
    row[sapply(row, is.null)] <- NA
    unlist(row)
  })
  data_matrix <- do.call("rbind", normalized_values)
  data_frame <- as.data.frame(data_matrix, stringsAsFactors = FALSE)
  data_frame
}

retrieve_coffees <- function (
  url = "https://database.coffeeinstitute.org/api/coffees/arabicaAjax",
  start = 0,
  length = 0) {
  body <- list(start = start, length = length)
  response <- POST(url, body = body, encode = "form")
  value <- content(response)
  data_frame <- data_frame_from_list(value$data)
  data_frame
}

retrieve_coffee_details <- function (identifier, base_url) {
  url <- file.path(base_url, identifier)
  response <- GET(url)
  value <- content(response)
  value
}

retrieve_details <- function (
  coffees_df,
  base_url = "https://database.coffeeinstitute.org/api/coffee/random") {
  identifiers <- as.numeric(coffees_df$random_id)
```

```

details_list <- lapply(identifiers, function (identifier) {
  value <- retrieve_coffee_details(identifier, base_url)
  value
})

details_list
}

dir.create(data_path, showWarnings = FALSE)
dir.create(download_path, showWarnings = FALSE)
coffees_df <- retrieve_coffees()
saveRDS(coffees_df, file = coffees_object_file_path)
create_date_marker_file(coffee_date_file_path)
details <- retrieve_details(coffees_df)
saveRDS(details, file = coffee_details_object_file_path)
create_date_marker_file(coffee_details_date_file_path)

```

## Normalize data

```

normalize_details <- function (details_list) {
  grade_by_id <- lapply(details_list, function (details) {
    row <- NA
    if (is.list(details$grade)) {
      row <- details$grade
      row$random_id <- details$random_id
    }

    row
  })
  grades_df <- data_frame_from_list(grade_by_id)
  green_by_id <- lapply(details_list, function (details) {
    row <- NA
    if (is.list(details$green)) {
      row <- details$green
      row$random_id <- details$random_id
    }

    row
  })

  greens_df <- data_frame_from_list(green_by_id)
  list(grades = grades_df, greens = greens_df)
}

normalize_grades <- function (grades_df) {
  grades_df <- grades_df[complete.cases(grades_df$random_id), ]
  grades_df
}

normalize_greens <- function (greens_df) {
  greens_df$pass <- as.logical(greens_df$pass)
  greens_column_names <- colnames(greens_df)
}

```



```

remove <- c("color_title", "pass")
integer_column_names <- greens_column_names[!(greens_column_names %in% remove)]
for (column_name in integer_column_names) {
  greens_df[[column_name]] <- as.numeric(greens_df[[column_name]])
}

greens_df <- greens_df[complete.cases(greens_df$random_id), ]
greens_df
}

normalized_details <- normalize_details(details)
grades_df <- normalize_grades(normalized_details$grades)
greens_df <- normalize_greens(normalized_details$greens)

```

## Save data

```

write_csv(coffees_df, coffees_file_path)
write_csv(grades_df, coffee_grades_file_path)
write_csv(greens_df, coffee_greens_file_path)
rm(details)
rm(normalized_details)

```

## Coffee information altitude normalization

```

range_1500_2000 <- coffees_or_df$altitude == "1500-2000"
coffees_or_df$altitude[range_1500_2000] <- 1500 + ((2000 - 1500) / 2)
coffees_or_df$altitude <- as.numeric(coffees_or_df$altitude)

```

## Coffee information altitude unit standardization

```

feet_units <- coffees_or_df$altitude_unit == "ft"
coffees_or_df$altitude[feet_units] <- coffees_or_df$altitude[feet_units] * 0.3048

```

## Coffee information weight unit standardization

```

pound_units <- coffees_or_df$weight_unit == "lbs"
coffees_or_df$weight[pound_units] <- coffees_or_df$weight[pound_units] * 0.453592

```

## Coffee information green analysis pass normalization

```

yes <- coffees_or_df$green_pass == "Y"
no <- coffees_or_df$green_pass == "N"
coffees_or_df$green_pass[yes] <- "TRUE"
coffees_or_df$green_pass[no] <- "FALSE"
coffees_or_df$green_pass <- as.logical(coffees_or_df$green_pass)

```

## Coffee grades/cupping scores pass conversion

```
greens_or_df$pass <- as.logical(greens_or_df$pass)
```

## Setup clean data

```
clean_path <- file.path(data_path, "clean")
coffees_clean_file_path <- file.path(clean_path, "coffees.csv")
coffee_grades_clean_file_path <- file.path(clean_path, "coffee_grades.csv")
coffee_greens_clean_file_path <- file.path(clean_path, "coffee_greens.csv")
dir.create(clean_path, showWarnings = FALSE)
```

## Save clean data

```
write_csv(coffees_or_df, coffees_clean_file_path)
write_csv(grades_or_df, coffee_grades_clean_file_path)
write_csv(greens_or_df, coffee_greens_clean_file_path)
rm(coffees_or_df)
rm(grades_or_df)
rm(greens_or_df)
```

## Setup and open relational database

```
coffees <- read_csv(coffees_clean_file_path)
grades <- read_csv(coffee_grades_clean_file_path)
greens <- read_csv(coffee_greens_clean_file_path)

ic_violation_ids_to_remove <- NULL
database_path <- file.path(data_path, "database")
dir.create(database_path, showWarnings = FALSE)
database_file_path <- file.path(database_path, "coffee.db")
connection <- dbConnect(RSQLite::SQLite(), database_file_path)
```

## Write data and list tables

```
dbWriteTable(connection, "coffees", coffees)
dbWriteTable(connection, "grades", grades)
dbWriteTable(connection, "greens", greens)
dbListTables(connection)
```

## Coffee completion integrity constraint

```
query <-
"
SELECT random_id, created_desc as created, completed_desc as completed
  FROM coffees
 WHERE created_desc > completed_desc
```

```
"
result <- dbGetQuery(connection, query)
```

### Coffee altitude integrity constraint

```
query <-
"
SELECT random_id, altitude
  FROM coffees
 WHERE altitude <= 0
"
result <- dbGetQuery(connection, query)
```

### Coffee grades integrity constraint

```
query <-
"
SELECT random_id
  FROM grades
 WHERE aroma < 0 OR aroma > 10 OR flavor < 0 OR flavor > 10
       OR after < 0 OR after > 10 OR acidity < 0 OR acidity > 10
       OR body < 0 OR body > 10 OR mouthfeel < 0 OR mouthfeel > 10
       OR balance < 0 OR balance > 10 OR uniformity < 0 OR uniformity > 10
       OR clean < 0 OR clean > 10 OR sweet < 0 OR sweet > 10
       OR overall < 0 OR overall > 10
"
result <- dbGetQuery(connection, query)
```

### Coffee grades total integrity constraint

```
query <-
"
SELECT E.random_id, A.total as actual_total, E.total_sum as expected_total
  FROM grades as A
 JOIN (
      SELECT random_id,
             (aroma + flavor + after + acidity + body
              + mouthfeel + balance + uniformity + clean
              + sweet + overall) as total_sum
        FROM grades
      ) as E
 ON A.random_id = E.random_id
 WHERE ABS(A.total - E.total_sum) > 0.0001
"
result <- dbGetQuery(connection, query)
```

### Coffee grades pass integrity constraint

```
query <-  
"  
SELECT random_id, total, pass  
  FROM grades  
 WHERE (total >= 80 AND pass < 1)  
        OR (total < 80 AND pass > 0)  
"  
result <- dbGetQuery(connection, query)
```

### Coffee green analysis values integrity constraint

```
query <-  
"  
SELECT random_id  
  FROM greens  
 WHERE full_black < 0 OR full_sour < 0 OR dried_cherry < 0 OR fungus < 0  
        OR severe_insect < 0 OR foreign_matter < 0 OR partial_black < 0  
        OR partial_sour < 0 OR parchment < 0 OR floater < 0 OR immature < 0  
        OR withered < 0 OR shell < 0 OR broken < 0 OR hull < 0 OR slight_insect < 0  
        OR moisture < 0 OR quakers < 0 OR category_one_total < 0  
        OR category_two_total < 0 OR category_one_equivalent < 0  
        OR category_two_equivalent < 0  
"  
result <- dbGetQuery(connection, query)
```

### Coffee green analysis totals

```
query <-  
"  
SELECT A.random_id, A.one_total, B.category_one_total,  
       A.two_total, B.category_two_total  
  FROM (  
    SELECT random_id,  
           (full_black + full_sour + dried_cherry  
            + fungus + severe_insect + foreign_matter) as one_total,  
           (partial_black + partial_sour + parchment + floater + immature  
            + withered + shell + broken + hull + slight_insect) as two_total  
      FROM greens  
    ) as A  
 JOIN greens as B  
   ON A.random_id = B.random_id  
 WHERE ABS(A.one_total - B.category_one_total) > 0  
        OR ABS(A.two_total - B.category_two_total) > 0  
"  
result <- dbGetQuery(connection, query)
```

### Coffee green analysis defects

```
query <-
"
SELECT A.random_id, A.one_defects, B.category_one_equivalent,
       A.two_defects, B.category_two_equivalent
FROM (
  SELECT random_id,
         (full_black + full_sour + dried_cherry
          + fungus + FLOOR(severe_insect / 5) + foreign_matter) as one_defects,
         (FLOOR(partial_black / 3) + FLOOR(partial_sour / 3)
          + FLOOR(parchment / 5) + FLOOR(floater / 5)
          + FLOOR(immature / 5) + FLOOR(withered / 5)
          + FLOOR(shell / 5) + FLOOR(broken / 5)
          + FLOOR(hull / 5) + FLOOR(slight_insect / 10)) as two_defects
  FROM greens
) as A
JOIN greens as B
  ON A.random_id = B.random_id
WHERE ABS(A.one_defects - B.category_one_equivalent) > 0
      OR ABS(A.two_defects - B.category_two_equivalent) > 0
"
result <- dbGetQuery(connection, query)
```

### Coffee at least one grade or greens integrity constraint

```
query <-
"
SELECT random_id, farm, mill, producer
FROM coffees
WHERE random_id NOT IN (SELECT random_id FROM grades)
"
result <- dbGetQuery(connection, query)
```

```
query <-
"
SELECT random_id, farm, mill, producer
FROM coffees
WHERE random_id NOT IN (SELECT random_id FROM greens)
"
result <- dbGetQuery(connection, query)
```

### Coffee at most one grades or greens integrity constraint

```
query <-
"
SELECT random_id
FROM coffees
WHERE random_id IN (
  SELECT random_id
  FROM (
```

```

        SELECT random_id, count(*) as count
        FROM grades GROUP BY random_id
    )
    WHERE count > 1
)
"
result <- dbGetQuery(connection, query)

query <-
"
SELECT random_id
FROM coffees
WHERE random_id IN (
    SELECT random_id
    FROM (
        SELECT random_id, count(*) as count
        FROM greens GROUP BY random_id
    )
    WHERE count > 1
)
"
result <- dbGetQuery(connection, query)

```

#### Setup final clean data without constraint violation entries

```

final_path <- file.path(data_path, "final")
coffees_final_file_path <- file.path(final_path, "coffees.csv")
coffee_grades_final_file_path <- file.path(final_path, "coffee_grades.csv")
coffee_greens_final_file_path <- file.path(final_path, "coffee_greens.csv")
dir.create(final_path, showWarnings = FALSE)

```

#### Remove integrity constraint violation entries

```

coffees_final <- coffees[!(coffees$random_id %in% ic_violation_ids_to_remove),]
grades_final <- grades[!(grades$random_id %in% ic_violation_ids_to_remove),]
greens_final <- greens[!(greens$random_id %in% ic_violation_ids_to_remove),]

```

#### Save final clean data

```

write_csv(coffees_final, coffees_final_file_path)
write_csv(grades_final, coffee_grades_final_file_path)
write_csv(greens_final, coffee_greens_final_file_path)
rm(coffees_final)
rm(grades_final)
rm(greens_final)

```

## Create OpenRefine YesWorkflow's

```
full_file_path <- function (file_path) { file.path(getwd(), file_path) }

or2yw_yw <- function (source_file_path, target_file_path) {
  source <- file.path(getwd(), source_file_path)
  target <- file.path(getwd(), target_file_path)
  command <- paste0(
    "or2yw -i \"",
    source,
    "\" -o \"",
    target,
    "\"")
  result <- try(system(command, intern = TRUE))
  result
}

openrefine_workflow_path <- "./workflow/openrefine"
dir.create(openrefine_workflow_path, showWarnings = FALSE)
recipe_names <- c(
  "coffees_recipe.json",
  "coffee_grades_recipe.json",
  "coffee_greens_recipe.json")
recipe_file_paths <- sapply(
  recipe_names,
  function (name, path) { file.path(path, name) },
  openrefine_path)
workflow_names <- c("coffees.yw", "coffee_grades.yw", "coffee_greens.yw")
workflow_paths <- sapply(
  workflow_names,
  function (name, path) { file.path(path, name) },
  openrefine_workflow_path)
result <- mapply(
  function (source, target) { or2yw_yw(source, target) },
  recipe_file_paths,
  workflow_paths)
```

## Create OpenRefine diagrams

```
or2yw_png <- function (source_file_path, target_file_path) {
  source <- file.path(getwd(), source_file_path)
  target <- file.path(getwd(), target_file_path)
  target <- gsub(" ", "\\ ", target)
  command <- paste0(
    "or2yw -i \"",
    source,
    "\" -o \"",
    target,
    "\"\\\"\\\"\\\" -ot png")
  result <- try(system(command, intern = TRUE))
  result
}
```

```

openrefine_workflow_image_path <- "./workflow/openrefine/png"
dir.create(openrefine_workflow_image_path, showWarnings = FALSE)
workflow_image_names <- c("coffees.png", "coffee_grades.png", "coffee_greens.png")
workflow_image_paths <- sapply(
  workflow_image_names,
  function (name, path) { file.path(path, name) },
  openrefine_workflow_image_path)
result <- mapply(
  function (source, target) { or2yw_png(source, target) },
  recipe_file_paths,
  workflow_image_paths)

```

## Calculate CSV difference summaries

```

create_file_path_pairs <- function (paths, file_names) {
  lapply(file_names, function (file_name) {
    file_paths <- sapply(paths, function (path, file) {
      file_path <- file.path(path, file)
      file_path
    },
    file_name)

    file_path_pairs <- list()
    file_paths_length <- length(file_paths)
    if (file_paths_length > 1) {
      for (index in 2:file_paths_length) {
        source <- file_paths[index - 1]
        target <- file_paths[index]
        file_path_pairs[[index - 1]] <- list(source = source, target = target)
      }
    }

    file_path_pairs
  })
}

csvdiff_summary <- function (source_file_path, target_file_path) {
  source <- file.path(getwd(), source_file_path)
  target <- file.path(getwd(), target_file_path)
  command <- paste0(
    "csvdiff --style=summary random_id \"",
    source,
    "\" \"",
    target,
    "\"")
  result <- try(system(command, intern = TRUE))
  result
}

paths <- c(download_path, openrefine_path, clean_path, final_path)
file_names <- c("coffees.csv", "coffee_grades.csv", "coffee_greens.csv")

```



```

file_path_pairs <- create_file_path_pairs(paths, file_names)

summaries <- lapply(file_path_pairs, function (file_pairs) {
  file_summaries <- lapply(file_pairs, function (pair) {
    summary <- csvdiff_summary(pair$source, pair$target)
    result <- NULL
    if (length(summary) > 1) {
      removed <- gsub("removed ", "", summary[1])
      added <- gsub("added ", "", summary[2])
      changed <- gsub("changed ", "", summary[3])
      result <- data.frame(
        Removed = removed,
        Added = added,
        Changed = changed,
        stringsAsFactors = FALSE)
    } else if (length(summary) == 1) {
      result <- data.frame(
        Removed = "0 rows (0.0%)",
        Added = "0 rows (0.0%)",
        Changed = "0 rows (0.0%)",
        stringsAsFactors = FALSE)
    }

    result
  })

  df <- do.call(rbind, file_summaries)
  rownames(df) <- c("OpenRefine", "R", "Constraints")
  df
})

```

Calculate total CSV difference summaries

```

paths <- c(download_path, final_path)
file_names <- c("coffees.csv", "coffee_grades.csv", "coffee_greens.csv")
file_path_pairs <- create_file_path_pairs(paths, file_names)

total_summaries <- lapply(file_path_pairs, function (file_pair) {
  pair <- file_pair[[1]]
  summary <- csvdiff_summary(pair$source, pair$target)
  result <- NULL
  if (length(summary) > 1) {
    removed <- gsub("removed ", "", summary[1])
    added <- gsub("added ", "", summary[2])
    changed <- gsub("changed ", "", summary[3])
    result <- data.frame(
      Removed = removed,
      Added = added,
      Changed = changed,
      stringsAsFactors = FALSE)
  } else if (length(summary) == 1) {
    result <- data.frame(

```

```

    Removed = "0 rows (0.0%)",
    Added = "0 rows (0.0%)",
    Changed = "0 rows (0.0%)",
    stringsAsFactors = FALSE)
  }
})

total_summaries_df <- do.call(rbind, total_summaries)
rownames(total_summaries_df) <- c(
  "Coffee information",
  "Grades/Cupping scores",
  "Green analysis")

```

## Create total CSV difference files

```

csvdiff <- function (source_file_path, target_file_path, output_file_path) {
  source <- file.path(getwd(), source_file_path)
  target <- file.path(getwd(), target_file_path)
  output <- file.path(getwd(), output_file_path)
  command <- paste0(
    "csvdiff --style=pretty --output=\"",
    output,
    "\" random_id \"",
    source,
    "\" \",
    target,
    "\"")
  result <- try(system(command, intern = TRUE))
  result
}

diff_path <- file.path(data_path, "diff")
dir.create(diff_path, showWarnings = FALSE)

paths <- c(download_path, final_path)
file_names <- c("coffees.csv", "coffee_grades.csv", "coffee_greens.csv")
file_path_pairs <- create_file_path_pairs(paths, file_names)
diff_file_names <- sapply(
  file_names,
  function (file_name) { gsub(".csv", ".json", file_name) })
diff_file_paths <- sapply(
  diff_file_names,
  function (file_name) { file.path(diff_path, file_name) })

diffs <- mapply(function (file_pair, output_file) {
  pair <- file_pair[[1]]
  summary <- csvdiff(pair$source, pair$target, output_file)
  result <- NULL
  if (length(summary) > 1) {
    removed <- gsub("removed ", "", summary[1])
    added <- gsub("added ", "", summary[2])
    changed <- gsub("changed ", "", summary[3])
  }

```

```
    result <- data.frame(
      Removed = removed,
      Added = added,
      Changed = changed,
      stringsAsFactors = FALSE)
  } else if (length(summary) == 1) {
    result <- data.frame(
      Removed = "0 rows (0.0%)",
      Added = "0 rows (0.0%)",
      Changed = "0 rows (0.0%)",
      stringsAsFactors = FALSE)
  }
},
file_path_pairs,
diff_file_paths)
```