
# STRONGER DATA SCIENCE

## USING R IN YOUR QUALITY CONTROL PROCESS

NOVEMBER 2019

Elicit

# QUALITY CONTROL (QC) IS...

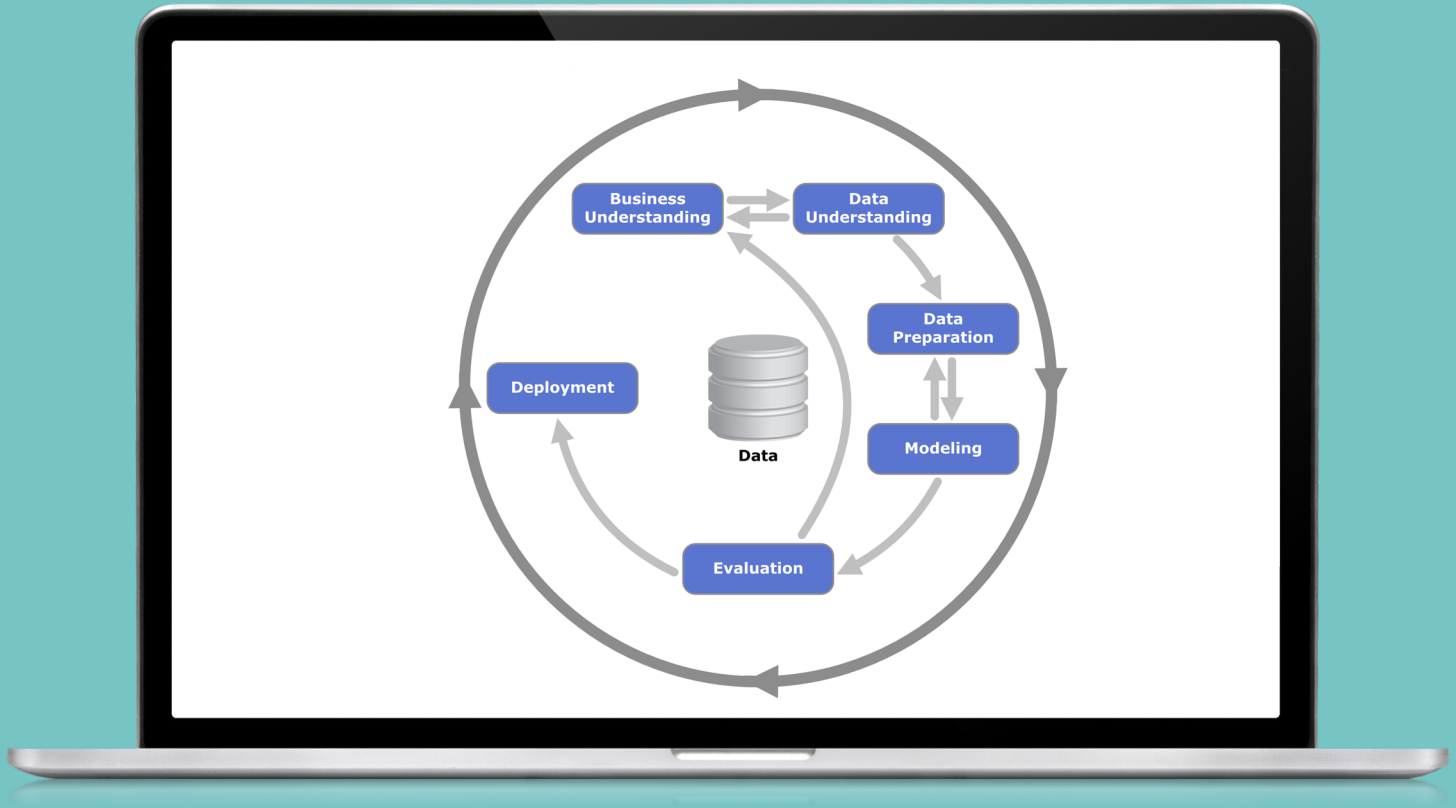
A set of activities used to ensure the “product” is performing as expected, is in line with design objectives, and meets the business goal(s)

# THE GOAL OF QC IS TO...

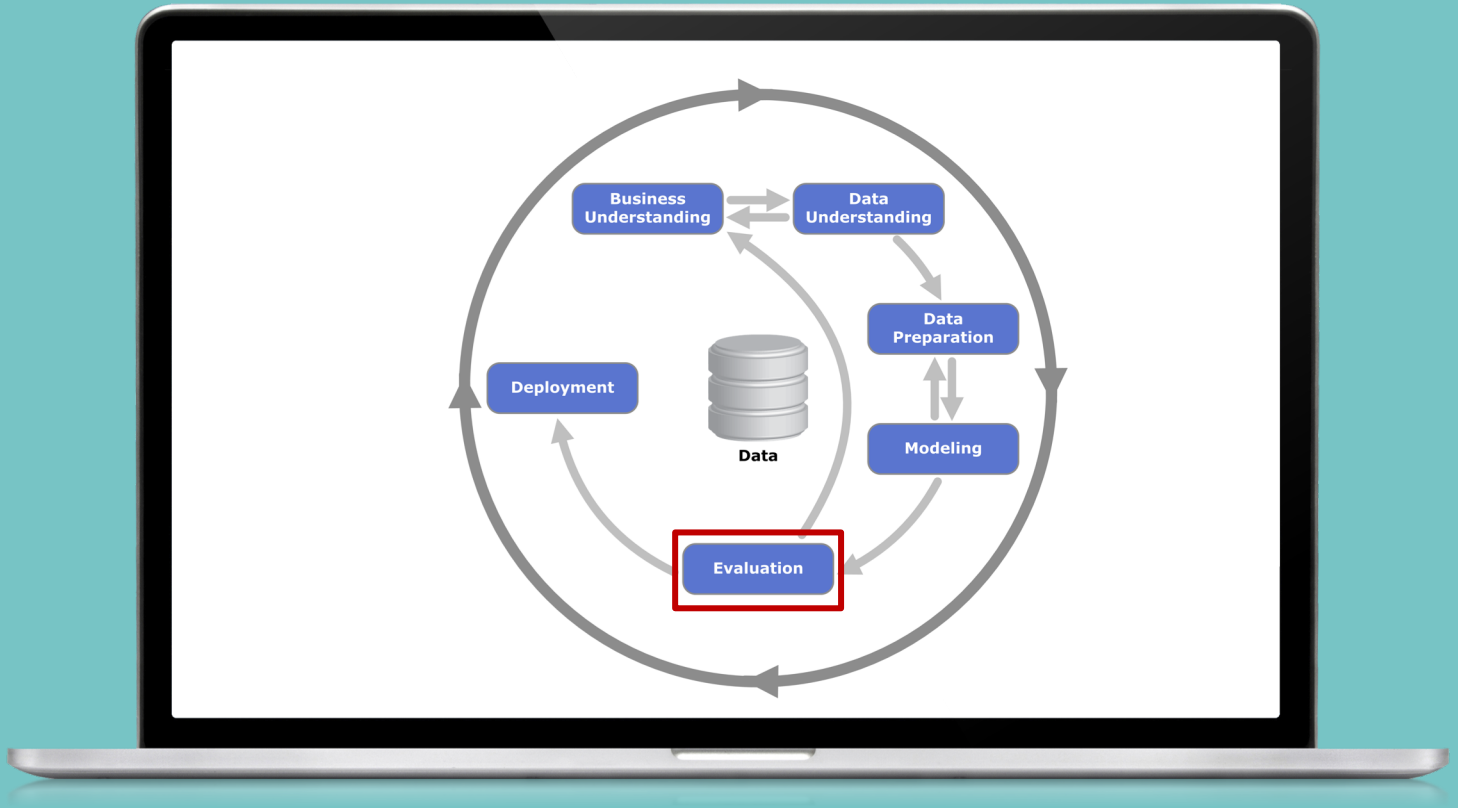
Identify and correct “defects” in the “product” BEFORE it is delivered

# QC HELPS ENSURE THE HIGHEST QUALITY OF DELIVERABLES

# CRISP – DM PROCESS



# A LIMITATION IS WAITING TO THE END OF THE PROCESS TO EVALUATE RESULTS



# DATA SCIENCE PROJECT PROCESS

# A STRONG QC PROCESS LEVERAGES GEEK, NERD AND SUIT SKILLS

## GEEK

Ensure we are getting data from the correct sources, understand it, and that our results are reproducible

## NERD

Ensure we are building new variables correctly, applying the appropriate methodology, and interpreting results correctly

## SUIT

Ensure we understand the business request and our results are communicated to the client in a way they can understand and aligns with the business question/objective

# TRANSLATE A REQUEST INTO A CONCRETE, WELL-DEFINED PROBLEM

## Nerd QC

- Identify and document business problem / questions that can be answered with data

## Geek QC

- Document available data sources
  - Do I understand these tables
  - Am I getting the data from the correct sources
- Clearly define how you will measure the data
  - How do we operationally define the unit of measurement (e.g. cust_id)

## Suit QC

- Align on business definitions / terminology
  - How do we define a “customer” (e.g. anyone with a valid cust_id that has made a purchase in the last year excluding employees)

# BUILD THE DATASET THAT WILL BE USED FOR ANALYSIS

## Nerd QC

- Review data joins and look and check for duplicate records and missing values
- Verify any new variable formulations / transformations are correct
- Review data summary statistics looking for outliers and any potential errors
- Implementation of defensive coding practices

## Geek QC

- Assess individual data quality
  - Correct handling of missing values (i.e. NA doesn't always equal to FALSE for boolean fields)
- Verify the use of correct data type when reading in source data
  - No unintended loss of precision for numeric values or loss of data for character values
- Data munging / cleaning
- Save final dataset using version control

## Suit QC

- Verify variable business definitions are correct
- Review summary statistics and works with the client to "gut check"

# PREFORM AN IN-DEPTH ANALYSIS, EVALUATE, AND REFINE

## Nerd QC

- Assess that methodology approaches are appropriate for answering the business question(s)
- Interpretation of results are accurate
- Identify limitations on the conclusions
- Identify and correct/document any bias in the data or methodology

## Geek QC

- Reproducibility
  - How do I verify I'm getting data from the same place every time

## Suit QC

- Validate the analysis output aligns with the business objectives/questions
- Support storytelling of initial results
- Support interim reviews with client to ensure findings are accurate and test drive associated insights story

# SUMMARIZE RESULTS IN A CLEAR AND ACTIONABLE STORY

## Nerd QC

- Review the code used to product the output for the final deliverable
  - Technical deliverable: Final model code
  - Business deliverable: PowerPoint
- Compare final business deliverable with the code output to ensure there are no copy & paste errors

## Geek QC

- Confirm the results reproducible
  - How do I verify I'm getting data from the same place every time
- Code is well documented

## Suit QC

- Ensure results are communicated in a way the client can understand and aligns with our original business question / objective

# OUR QC PROCESS AND TEMPLATE

# QC PROCESS AT ELICIT

Assigning a dedicated DS Developer that completes the work and a dedicated QC Auditor to review the work

DEVELOPER
(KAT)

Any data scientist assigned at the beginning of the project that does the actual work

AUDITOR
(CURT)

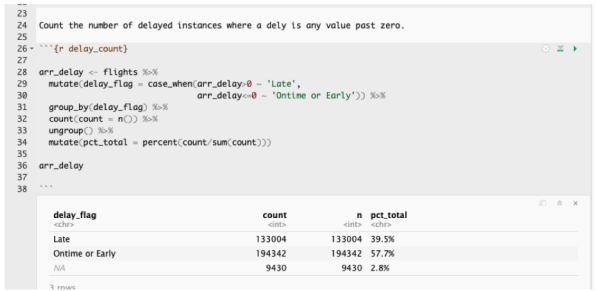
A senior data scientist assigned at the beginning of the project to review the work

# QC PROCESS AT ELICIT

Assigning a dedicated DS Developer that completes the work and a dedicated QC Auditor to review the work

Awesome data science work is completed in R and code is saved in the Github repository

## CURRENT CODE



```
Count the number of delayed instances where a dely is any value past zero.

```{r delay_count}

arr_delay <- flights %>%
  mutate(delay_flag = case_when(arr_delay>0 ~ 'Late',
                                arr_delay<=0 ~ 'Ontime or Early')) %>%
  group_by(delay_flag) %>%
  count(count = n()) %>%
  ungroup() %>%
  mutate(pct_total = percent(count/sum(count)))

arr_delay

```
```

| delay_flag <chr> | count <int> | n <int> | pct_total <chr> |
|---|---|---|---|
| Late | 133004 | 133004 | 39.5% |
| Ontime or Early | 194342 | 194342 | 57.7% |
| NA | 9430 | 9430 | 2.8% |

3 rows

## VERSION HISTORY

```
  arr_delay <- flights %>%
-   filter(arr_delay>0) %>%
-   count(count = n())
+   mutate(delay_flag = case_when(arr_delay>0 ~ 'Late',
+                                 arr_delay<=0 ~ 'Ontime or Early')) %>%
+   group_by(delay_flag) %>%
+   count(count = n()) %>%
+   ungroup() %>%
+   mutate(pct_total = percent(count/sum(count)))

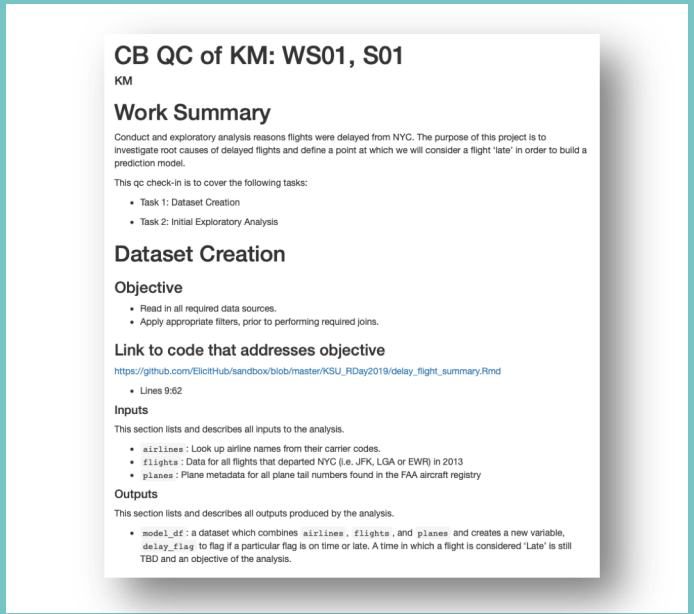
  arr_delay
```

# QC PROCESS AT ELICIT

Assigning a dedicated DS Developer that completes the work and a dedicated QC Auditor to review the work

Awesome data science work is completed in R and code is saved in the Github repository

Developer uses a work summary template to list the items needing to be reviewed along with key concepts and assumptions



## CB QC of KM: WS01, S01

KM

### Work Summary

Conduct and exploratory analysis reasons flights were delayed from NYC. The purpose of this project is to investigate root causes of delayed flights and define a point at which we will consider a flight 'late' in order to build a prediction model.

This qc check-in is to cover the following tasks:

- Task 1: Dataset Creation
- Task 2: Initial Exploratory Analysis

### Dataset Creation

#### Objective

- Read in all required data sources.
- Apply appropriate filters, prior to performing required joins.

#### Link to code that addresses objective

https://github.com/ElicitHub/sandbox/blob/master/KSU_RDay2019/delay_flight_summary.Rmd

- Lines 9:62

#### Inputs

This section lists and describes all inputs to the analysis.

- `airlines` : Look up airline names from their carrier codes.
- `flights` : Data for all flights that departed NYC (i.e. JFK, LGA or EWR) in 2013
- `planes` : Plane metadata for all plane tail numbers found in the FAA aircraft registry

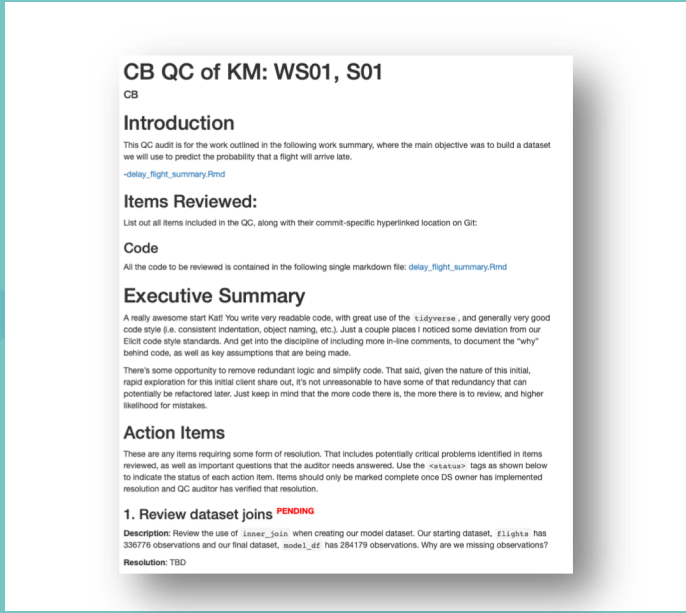
#### Outputs

This section lists and describes all outputs produced by the analysis.

- `model_df` : a dataset which combines `airlines` , `flights` , and `planes` and creates a new variable, `delay_flag` to flag if a particular flag is on time or late. A time in which a flight is considered 'Late' is still TBD and an objective of the analysis.

# QC PROCESS AT ELICIT

Assigning a dedicated DS Developer that completes the work and a dedicated QC Auditor to review the work



## CB QC of KM: WS01, S01

CB

### Introduction

This QC audit is for the work outlined in the following work summary, where the main objective was to build a dataset we will use to predict the probability that a flight will arrive late.

-delay_flight_summary.Rmd

### Items Reviewed:

List out all items included in the QC, along with their commit-specific hyperlinked location on Git:

### Code

All the code to be reviewed is contained in the following single markdown file: delay_flight_summary.Rmd

### Executive Summary

A really awesome start Kat! You write very readable code, with great use of the `tidyverse`, and generally very good code style (i.e. consistent indentation, object naming, etc.). Just a couple places I noticed some deviation from our Elicit code style standards. And get into the discipline of including more in-line comments, to document the "why" behind code, as well as key assumptions that are being made.

There's some opportunity to remove redundant logic and simplify code. That said, given the nature of this initial, rapid exploration for this initial client share out, it's not unreasonable to have some of that redundancy that can potentially be refactored later. Just keep in mind that the more code there is, the more there is to review, and higher likelihood for mistakes.

### Action Items

These are any items requiring some form of resolution. That includes potentially critical problems identified in items reviewed, as well as important questions that the auditor needs answered. Use the `<status>` tags as shown below to indicate the status of each action item. Items should only be marked complete once DS owner has implemented resolution and QC auditor has verified that resolution.

#### 1. Review dataset joins PENDING

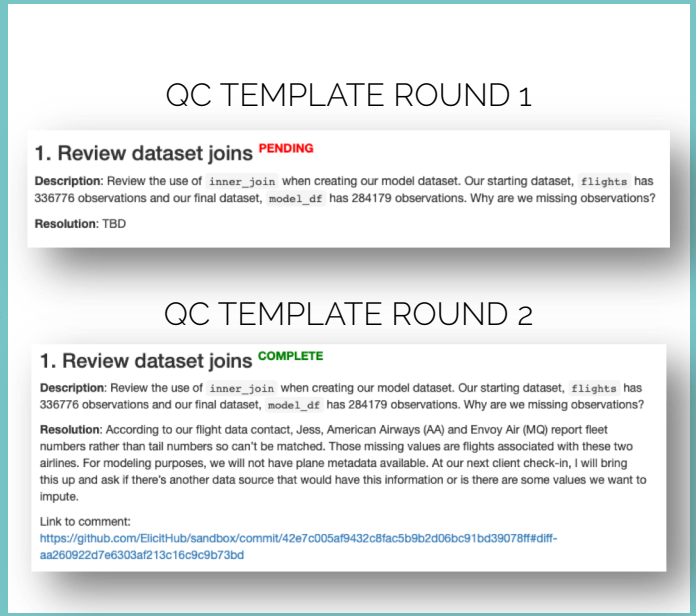
**Description**: Review the use of `inner_join` when creating our model dataset. Our starting dataset, `flights` has 336776 observations and our final dataset, `model_df` has 284179 observations. Why are we missing observations?

**Resolution**: TBD

QC Auditor uses a QC template to note all the items that require some form of resolution

# QC PROCESS AT ELICIT

Assigning a dedicated DS Developer that completes the work and a dedicated QC Auditor to review the work



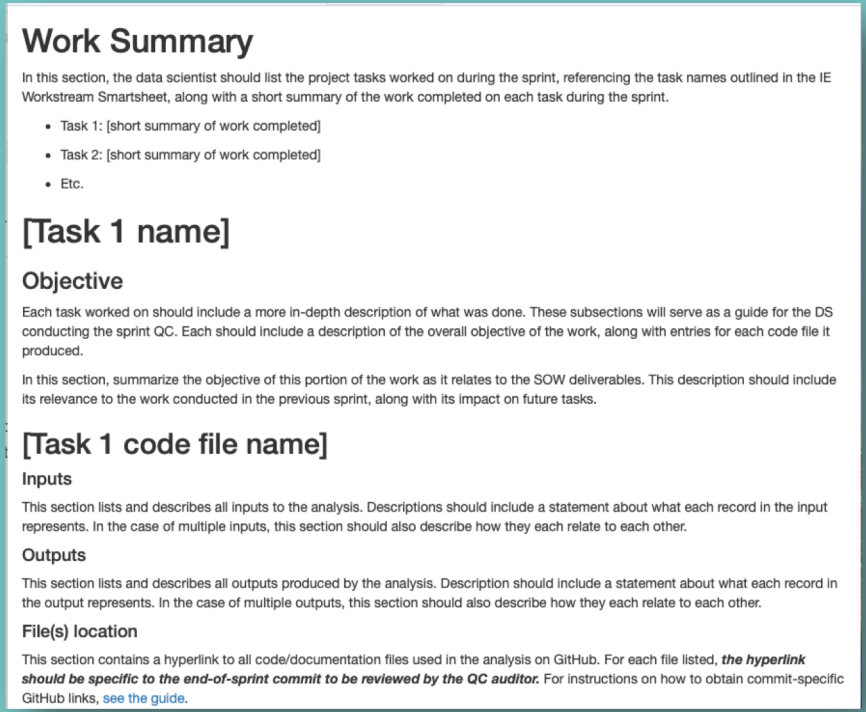

QC Auditor uses a QC template to note all the items that require some form of resolution

The Developer makes changes and documents the resolutions and sends the file back to the QC auditor to review

# WORK SUMMARY TEMPLATE

## WORK SUMMARY

- Description of the overall project
- In-depth description of each tasks completed that need to be reviewed
- For each task, include:
  - A description of the inputs used for analysis
  - A description of the outputs produced
  - GitHub hyperlink of the code/documentation files used in the analysis



## Work Summary

In this section, the data scientist should list the project tasks worked on during the sprint, referencing the task names outlined in the IE Workstream Smartsheet, along with a short summary of the work completed on each task during the sprint.

- Task 1: [short summary of work completed]
- Task 2: [short summary of work completed]
- Etc.

## [Task 1 name]

### Objective

Each task worked on should include a more in-depth description of what was done. These subsections will serve as a guide for the DS conducting the sprint QC. Each should include a description of the overall objective of the work, along with entries for each code file it produced.

In this section, summarize the objective of this portion of the work as it relates to the SOW deliverables. This description should include its relevance to the work conducted in the previous sprint, along with its impact on future tasks.

## [Task 1 code file name]

### Inputs

This section lists and describes all inputs to the analysis. Descriptions should include a statement about what each record in the input represents. In the case of multiple inputs, this section should also describe how they each relate to each other.

### Outputs

This section lists and describes all outputs produced by the analysis. Description should include a statement about what each record in the output represents. In the case of multiple outputs, this section should also describe how they each relate to each other.

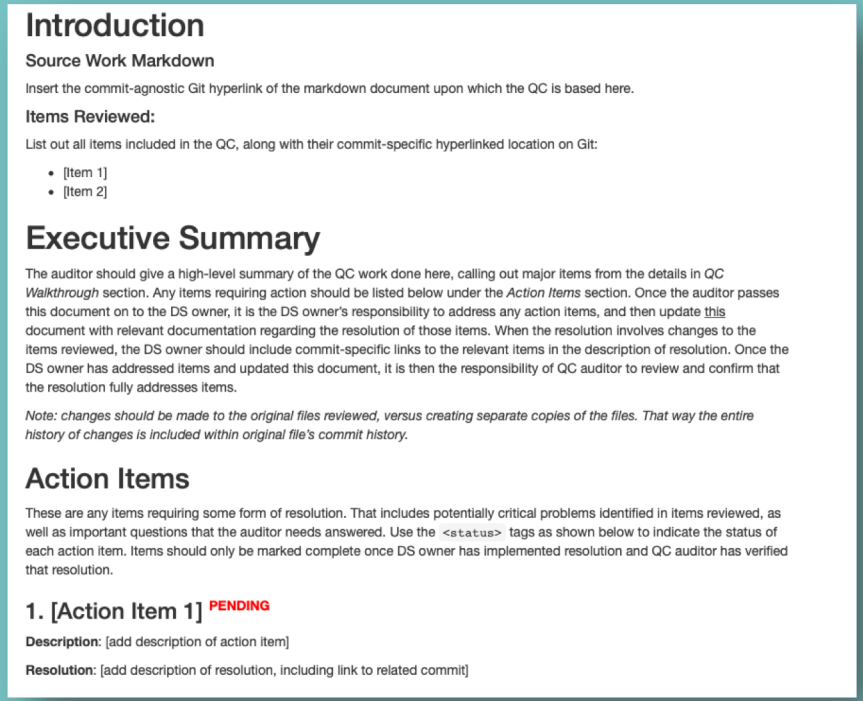
### File(s) location

This section contains a hyperlink to all code/documentation files used in the analysis on GitHub. For each file listed, ***the hyperlink should be specific to the end-of-sprint commit to be reviewed by the QC auditor.*** For instructions on how to obtain commit-specific GitHub links, see the guide.

# QC AUDITOR TEMPLATE

## QC TEMPLATE

- GitHub hyperlinked items reviewed for the particular QC check
- High-level summary of the QC work done highlighting any action items that need to be addressed by the developer
- For each action item, include:
  - The description of the action items
  - The resolution of the developer including the link to the change committed in GitHub



## Introduction

### Source Work Markdown

Insert the commit-agnostic Git hyperlink of the markdown document upon which the QC is based here.

### Items Reviewed:

List out all items included in the QC, along with their commit-specific hyperlinked location on Git:

- [Item 1]
- [Item 2]

## Executive Summary

The auditor should give a high-level summary of the QC work done here, calling out major items from the details in *QC Walkthrough* section. Any items requiring action should be listed below under the *Action Items* section. Once the auditor passes this document on to the DS owner, it is the DS owner's responsibility to address any action items, and then update this document with relevant documentation regarding the resolution of those items. When the resolution involves changes to the items reviewed, the DS owner should include commit-specific links to the relevant items in the description of resolution. Once the DS owner has addressed items and updated this document, it is then the responsibility of QC auditor to review and confirm that the resolution fully addresses items.

*Note: changes should be made to the original files reviewed, versus creating separate copies of the files. That way the entire history of changes is included within original file's commit history.*

## Action Items

These are any items requiring some form of resolution. That includes potentially critical problems identified in items reviewed, as well as important questions that the auditor needs answered. Use the `<status>` tags as shown below to indicate the status of each action item. Items should only be marked complete once DS owner has implemented resolution and QC auditor has verified that resolution.

### 1. [Action Item 1] PENDING

**Description**: [add description of action item]

**Resolution**: [add description of resolution, including link to related commit]

# TEMPLATE DEMO

# QC BEST PRACTICES

### DEFINE THE PROBLEM

Clearly define and document the primary objectives

### CLIENT & G,N,S COLLABORATION

The client helps validate assumptions and our Geeks, Nerds, and Suits help ensure our output is relevant, accurate, meaningful, and actionable

### USE A VERSION CONTROL TOOL

Version control software keeps track of modifications and ensures everyone is working from the latest and greatest

### DEVELOPER ≠ AUDITOR

The one developing the code is not the same person QC'ing

### QC ALL THE WAY THROUGH

Don't forget to review the final presentation both the code used to create the output and the deliverable

### INVEST TIME FOR QC

Build QC time into the project at the beginning during the project planning and resource allocation

# WHAT CAN YOU DO NOW?

## EVALUATE CURRENT QC PROCESS

Evaluate your current QC process starting ensuring you are checking off all the best practices

## WORK WITH A BUDDY

Find a buddy and practice the QC role

## PRACTICE DEFENSIVE DESIGN

Keep code simple and readable and design such that bugs and bad data don't corrupt your process

# REFERENCES

## OUR PREFERRED TOOLS/PACKAGES

- Reproducibility
  - R markdown and R notebooks
  - drake
- Version Control
  - Github
  - archivist
- Data Munging & Summary Statistics
  - tidyverse
  - dataMaid
- Defensive Code Design
  - assertthat
  - testthat

# QUESTIONS?

kat.morgan@elicitinsights.com

elicitinsights.com

*github.com/ElicitHub/*
*conferences/ksu-rday-2019*