



Automating Real-World Data Analysis using R Packages and Shiny: A Three-Stage Workflow

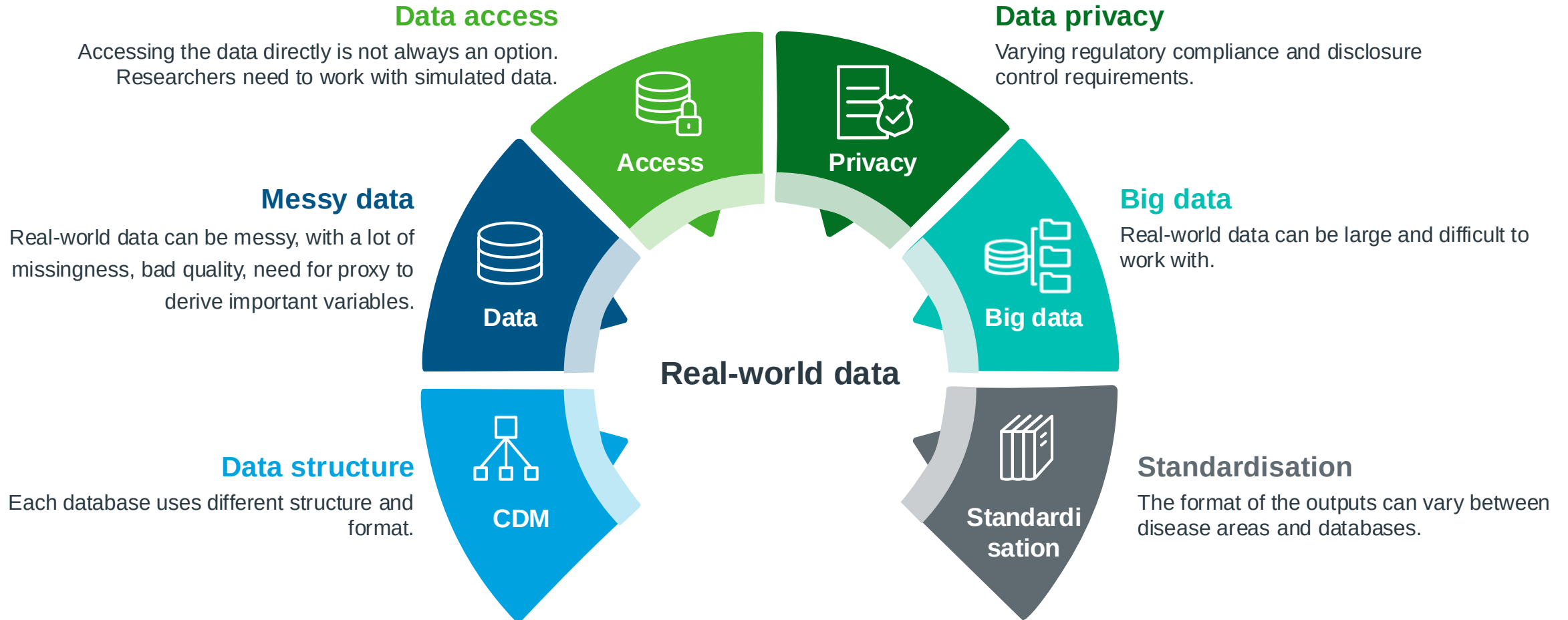
Alexandrina Lambova, Nikolay Trankov, Elena Chaparova,
Stavros Oikonomou, Louise Raiteri



Agenda

- Real-World data challenges
- Typical RWE project workflow
- Our three-stage approach
- Case example
- Package development
- Efficiency gains

Real-world data challenges



Our goals



Increase the quality of outputs

Reduce or eliminate potential errors

Reduce the time needed to perform the analysis

Reduce resources and effort throughout the whole project

Reduce the time of training for new joiners or unexperienced colleagues

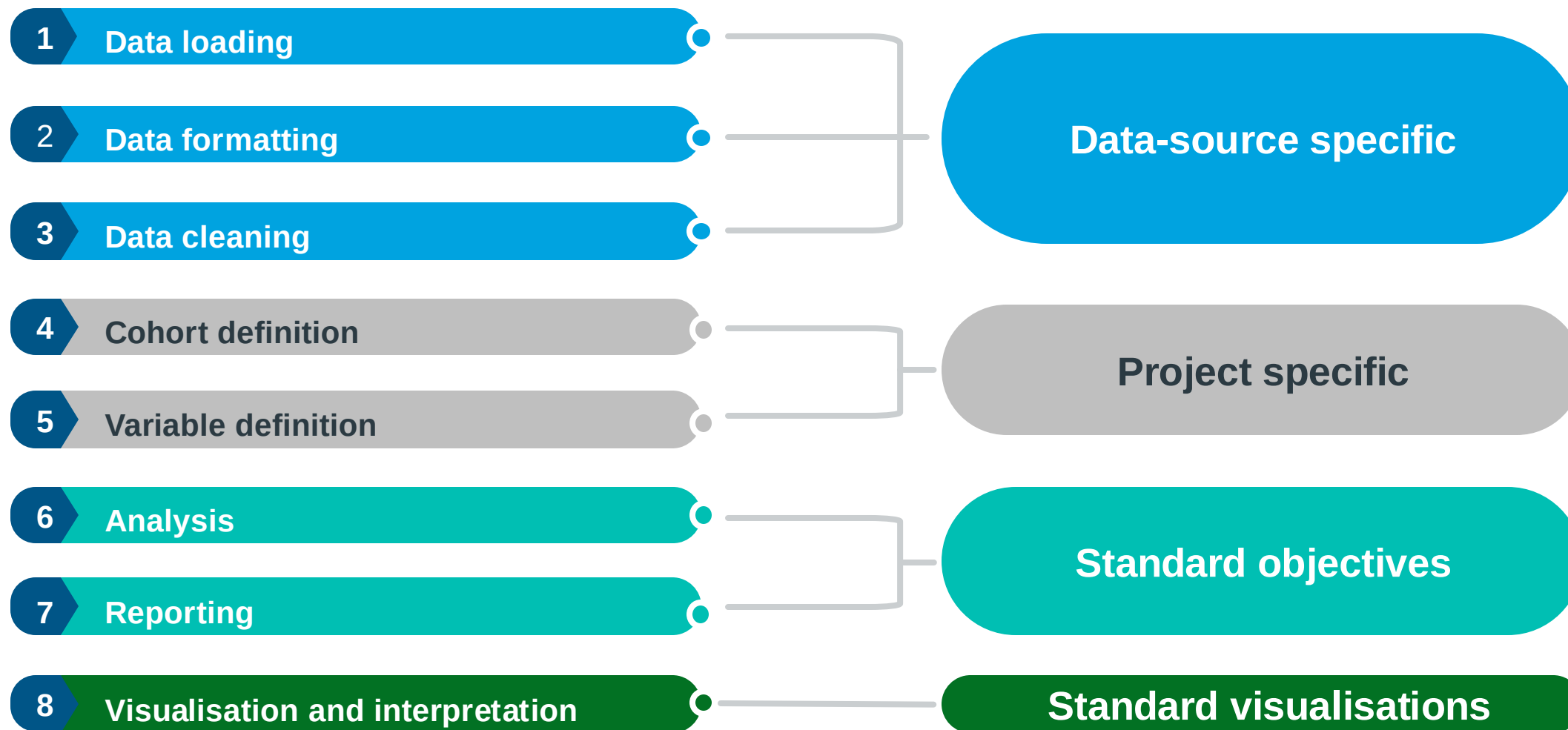
Project workflow

For standard Real-world data studies

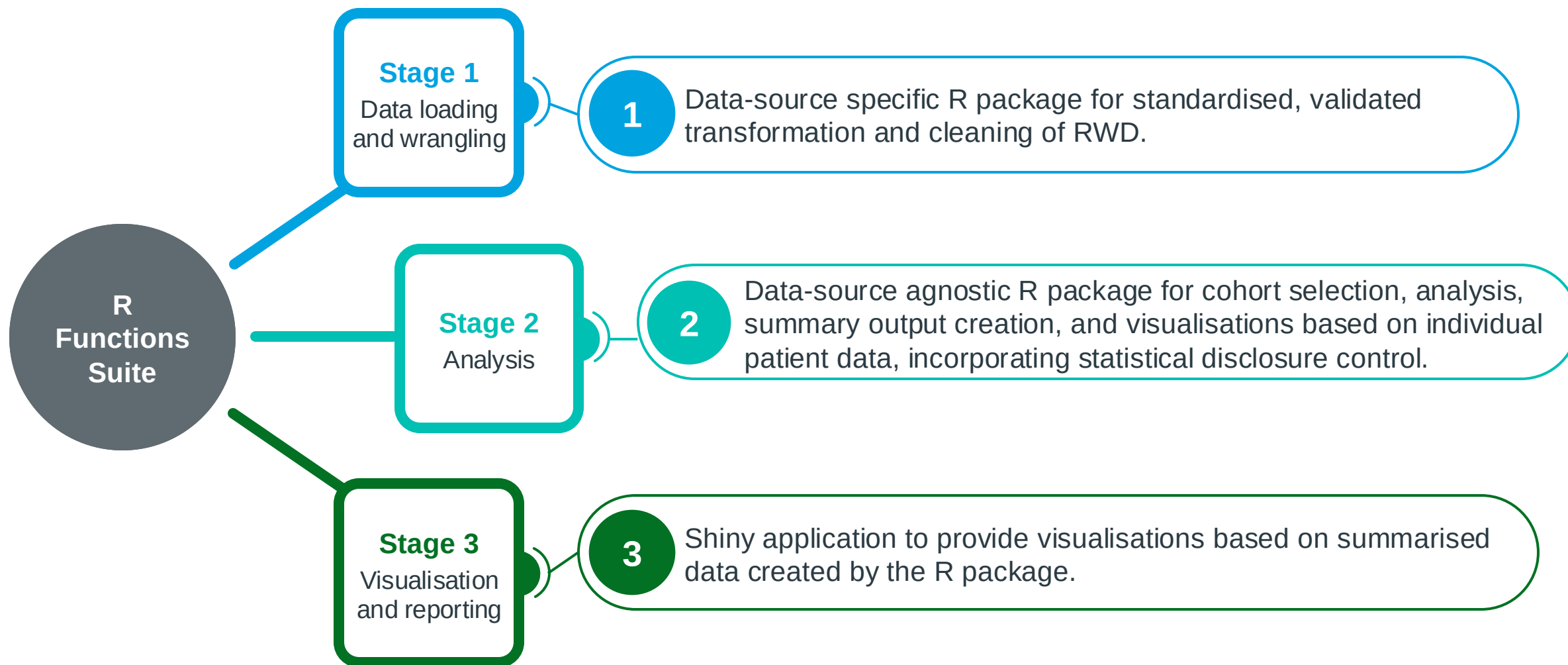
- 1 Data loading
- 2 Data formatting
- 3 Data cleaning
- 4 Cohort definition
- 5 Variable definition
- 6 Analysis
- 7 Reporting
- 8 Visualisation and interpretation

Project workflow

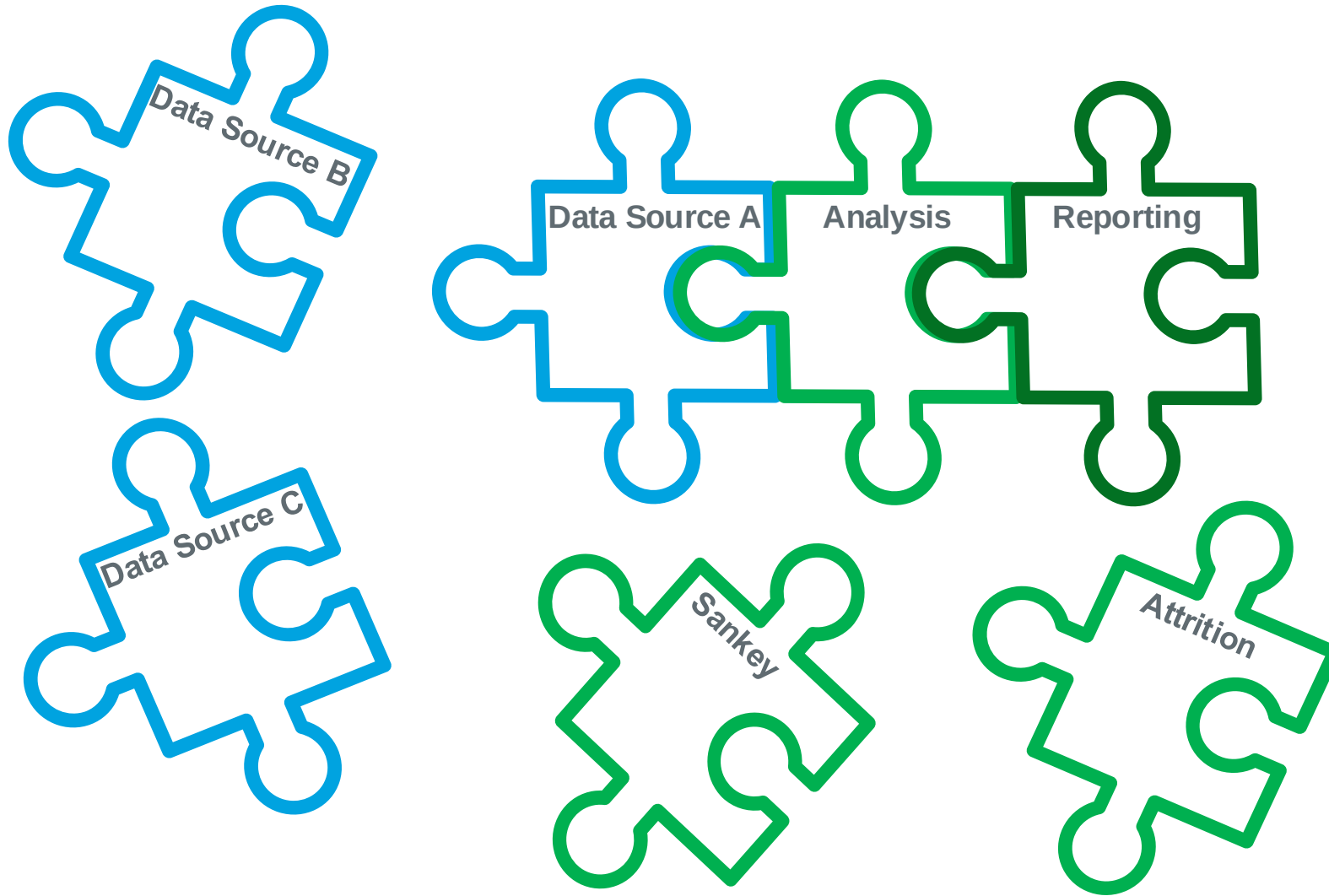
For standard Real-world data studies



Our three-stage approach to data analysis



Our approach is automated using custom modular functions



Stage 1: Data loading and wrangling

Data source specific packages including loading, formatting and cleaning of the data

Load data



loading spec -> loading function

ID	Date 1	Date 2	Sex	Vital status
1	2017-09-06 00.00.00	08MAR2010	1	A
2	2018-03-07 00.00.00	10APR2022	2	D
3	2019-05-08 00.00.00	12JUN2012	1	D3
4	2017-06-09 00.00.00	19DEC2009	0	D4
X	2022-01-01 00.00.00	NA	1	A

Format data



formatting spec -> formatting function

ID	Date 1	Date 2	Sex	Vital status
1	2017-09-06	2010-03-08	Male	Alive
2	2018-03-07	2022-04-10	Female	Dead
3	2019-05-08	2012-06-12	Male	Dead
4	2017-06-09	2009-12-19	NA	Dead
X	2022-01-01	NA	Male	Alive

Clean data



Cleaning functions

ID	Date 1	Date 2	Sex	Vital status
1	2017-09-06	2010-03-08	Male	Alive
2	2018-03-07	2022-04-10	Female	Dead
3	2019-05-08	2012-06-12	Male	Dead
4	2017-06-09	2009-12-19	NA	Dead

Loading and formatting specifications

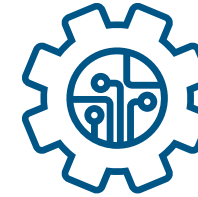
Reusable structure, specific to each data source

Load data



```
loading_spec <- list(  
  <dataset1> = list(  
    input = <path_with_filename1>,  
    select = c(  
      <column1> = <class to be loaded as>,  
      <column2> = <class to be loaded as>  
    )  
  ),  
  <dataset2> = list(  
    input = <path_with_filename2>,  
    select = c(  
      <column1> = <class to be loaded as>,  
      <column2> = <class to be loaded as>,  
      <column3> = <class to be loaded as>  
    )  
  )  
)
```

Format data



```
formatting_spec <- list(  
  <dataset1> = list(  
    list(<column1>, <function1>, <optional function1  
arg1>),  
    list(<column2>, <function2>, <optional function2  
arg1>)  
  ),  
  <dataset2> = list(  
    list(<column1>, <function3>, <optional function3  
arg1>),  
    list(<column2>, <function4>, <optional function4  
arg1>),  
    list(<column3>, <function5>, <optional function5  
arg1>)))
```

Loading and formatting specifications

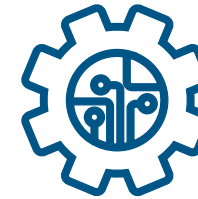
Reusable structure, specific to each data source

Load data



```
loading_spec <- list(  
  patient = list(  
    input = file.path(path, "patient.csv"),  
    select = c(  
      PATIENTID = "character",  
      VITALSTATUS = "character",  
      ETHNICITY = "character")  
    ),  
  tumour = list(  
    input = file.path(path, "tumour.csv"),  
    select = c(  
      TUMOURID = "character",  
      AGE = "character",  
      DIAGNOSISDATE = "character",  
      TUMOUR_STAGE = "character")))
```

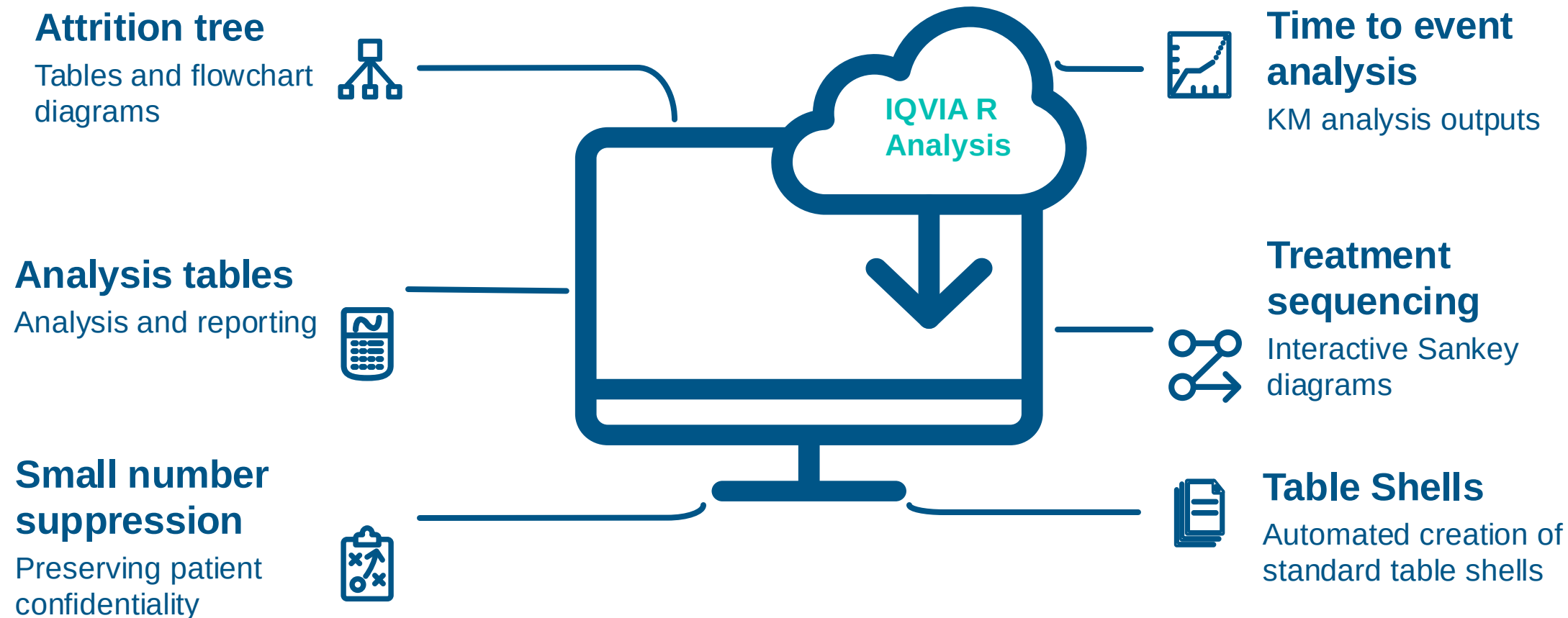
Format data



```
formatting_spec <- list(  
  patient = list(  
    list(VITALSTATUS, format_vitalstatus),  
    list(ETHNICITY, format_ethnicity, granular =  
      FALSE)  
    ),  
  tumour = list(  
    list(AGE, as.numeric),  
    list(DIAGNOSISDATE, format_date),  
    list(TUMOUR_STAGE, format_stage)  
    )  
)
```

Stage 2: Analysis

R package for analysis and reporting



Object-oriented approach

The principles of OOP were combined successfully with the powerful functionalities of R



Flexible

Tree-based hierarchical structures



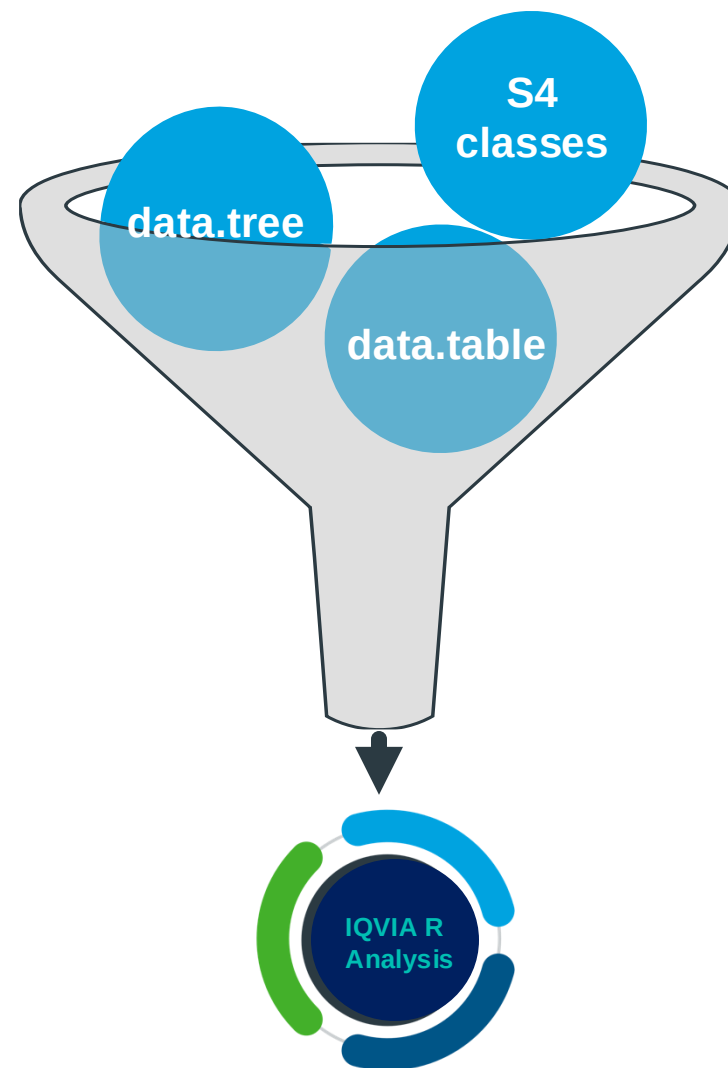
Extensible

Hierarchy of inherited classes



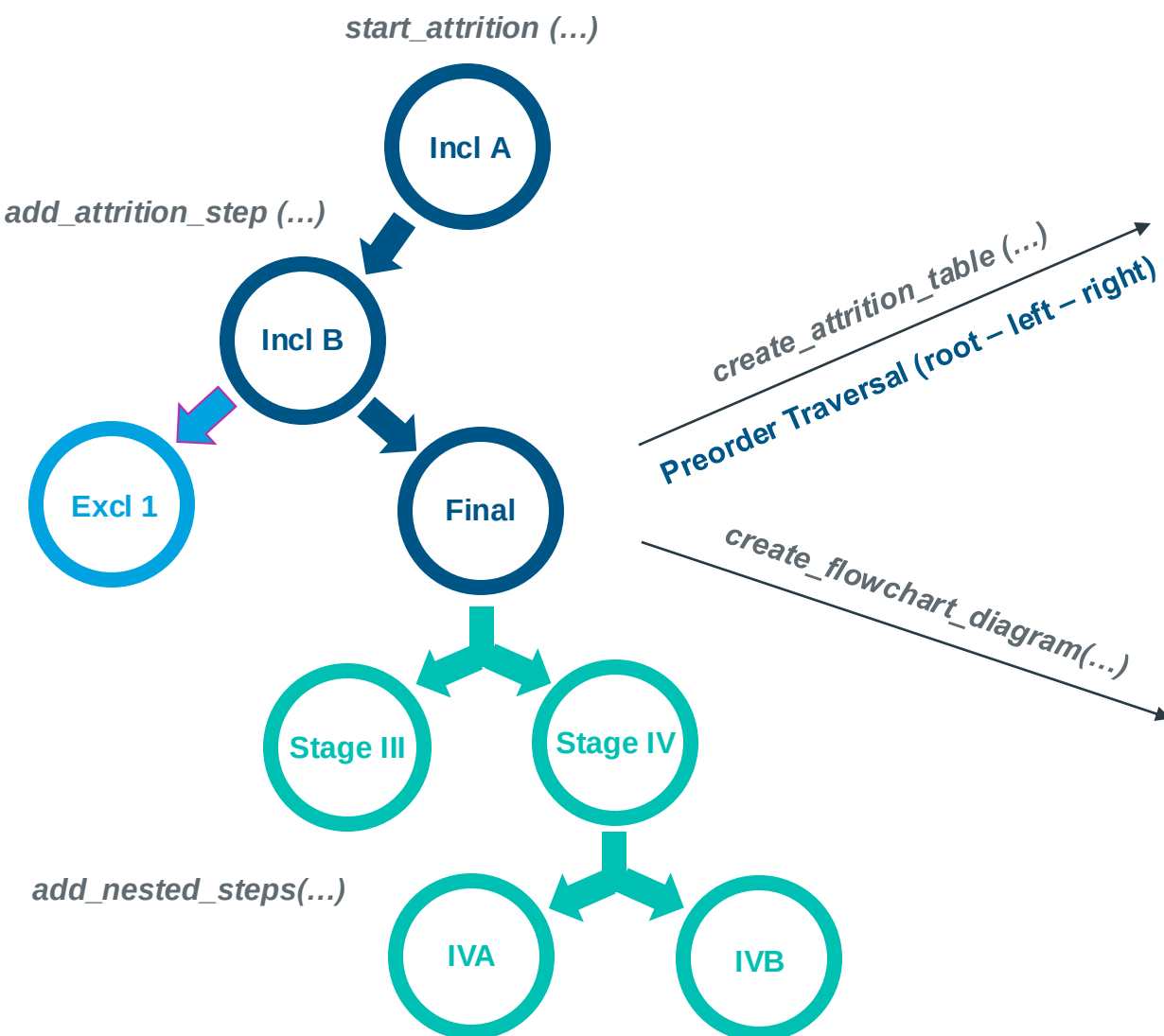
Customisable

Generic functions and interfaces

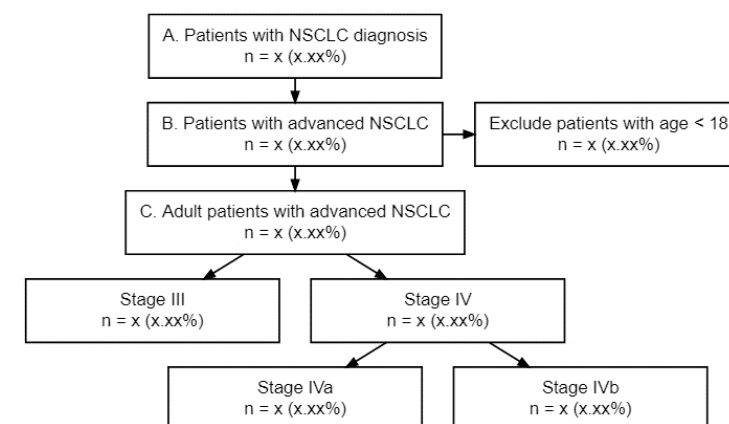


Attrition functions

Step-by-step building of attrition tree during the filtering of the data

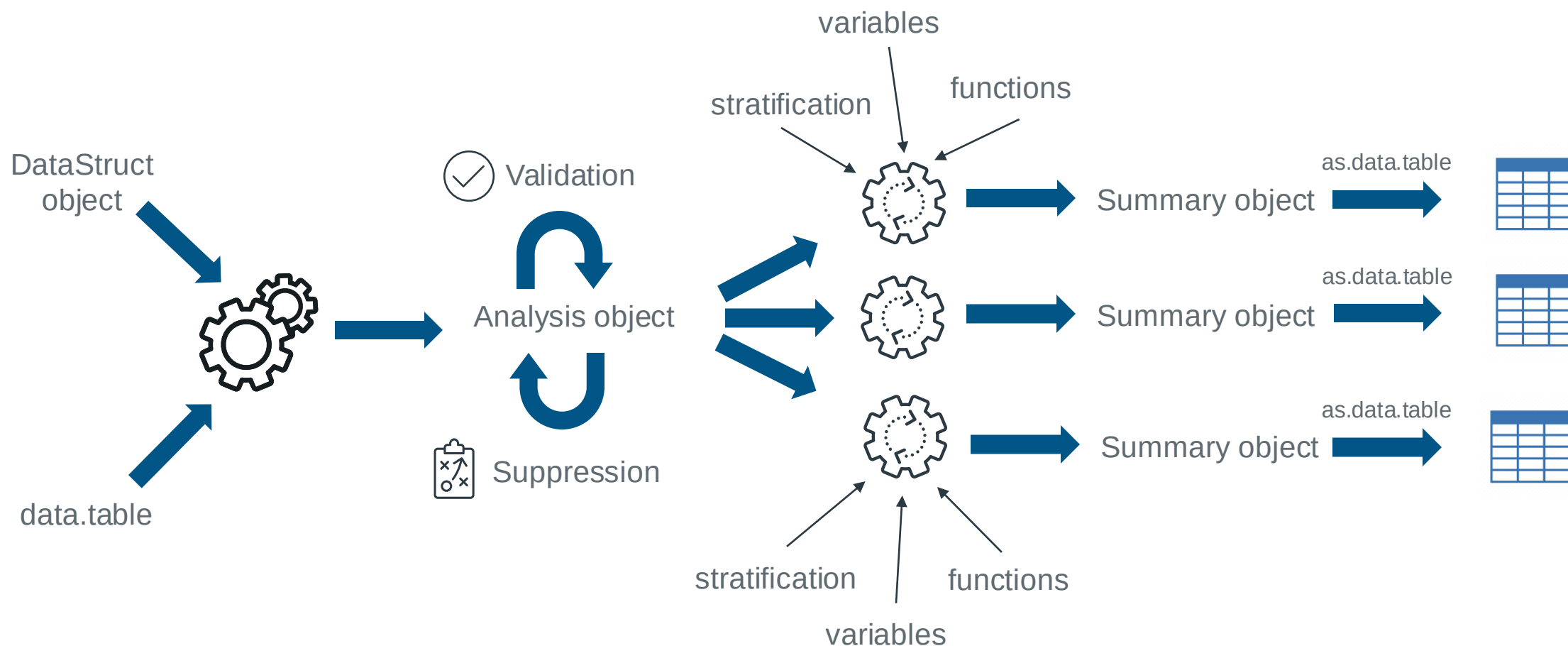


Year of diagnosis	Total		2016		2017		2018	
---	N	% of previous	N	% of previous	N	% of previous	N	% of previous
Inclusion criteria								
A. Patients with NSCLC diagnosis	x	x.xx%	x	x.xx%	x	x.xx%	x	x.xx%
B. Patients with advanced NSCLC	x	x.xx%	x	x.xx%	x	x.xx%	x	x.xx%
Exclusion criteria								
Exclude patients with age < 18	x	x.xx%	x	x.xx%	x	x.xx%	x	x.xx%
Final cohort								
C. Adult patients with advanced NSCLC	x	x.xx%	x	x.xx%	x	x.xx%	x	x.xx%
Stage III	x	x.xx%	x	x.xx%	x	x.xx%	x	x.xx%
Stage IV	x	x.xx%	x	x.xx%	x	x.xx%	x	x.xx%
-- Stage IVa	x	x.xx%	x	x.xx%	x	x.xx%	x	x.xx%
-- Stage IVb	x	x.xx%	x	x.xx%	x	x.xx%	x	x.xx%

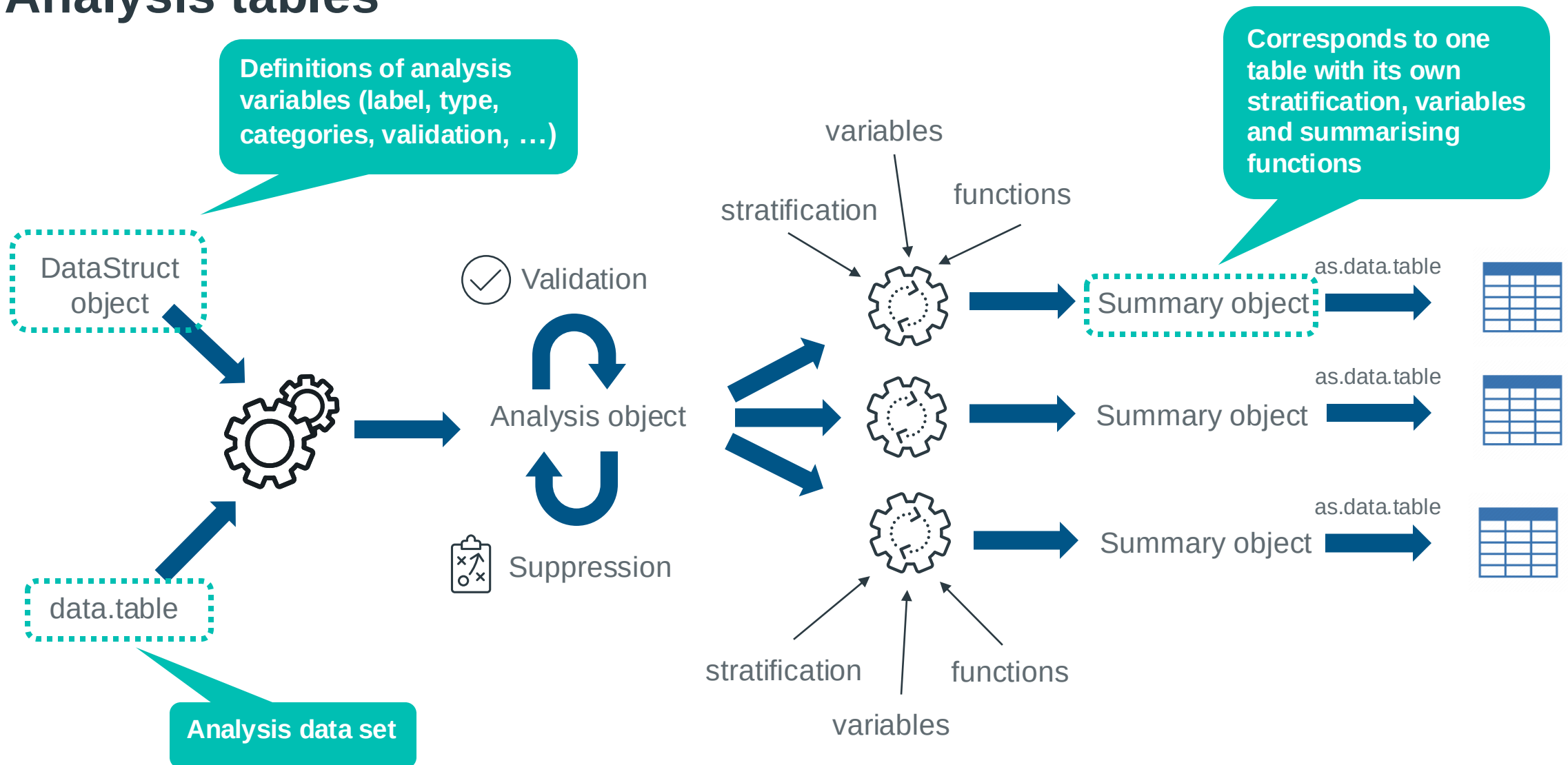


Analysis tables

Pipeline of objects and functions that allows data set level validation and suppression, and flexible and easily customizable reporting of summary statistics and test results



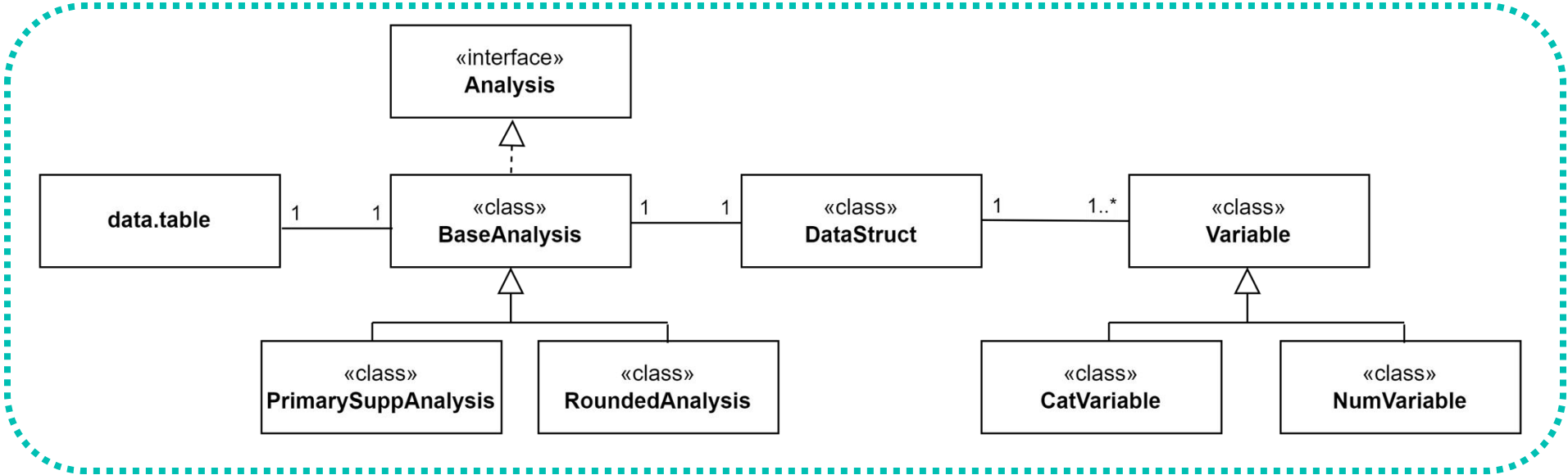
Analysis tables



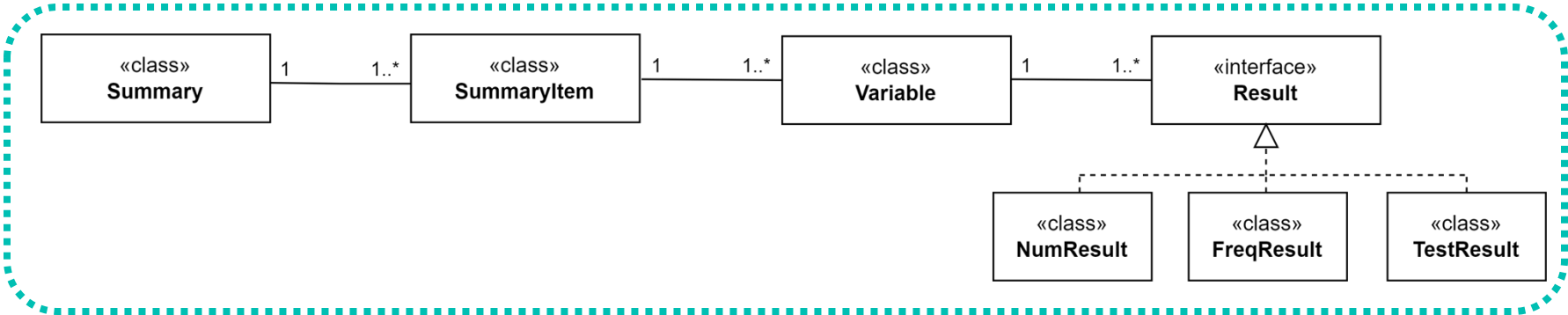
Analysis tables

Hierarchies of classes and interfaces that allow extensibility and flexibility

Combines variables' definitions with the analysis data set. Data set level validation and suppression of counts.



Represents a table with specific stratification, variables and analysis included. SummaryItem object for each strata combination.



Stage 3: Visualisation and reporting

Shiny application for visualising and reporting pre-calculated summarised outputs



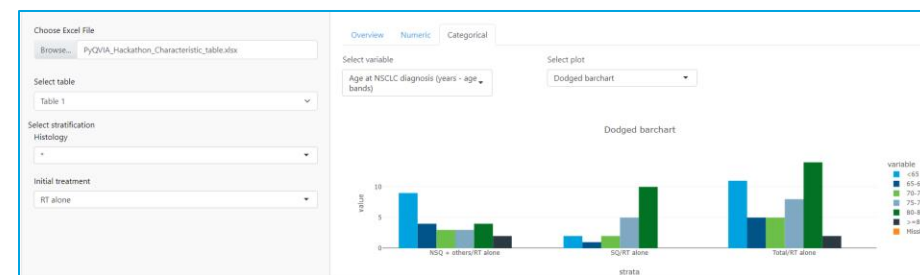
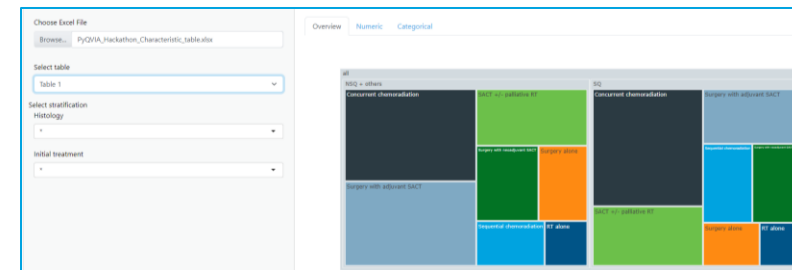
Import summarised outputs to the app



Explore the data by selecting different sub-groups and visualisations



Export plots and tables for reporting



Case example

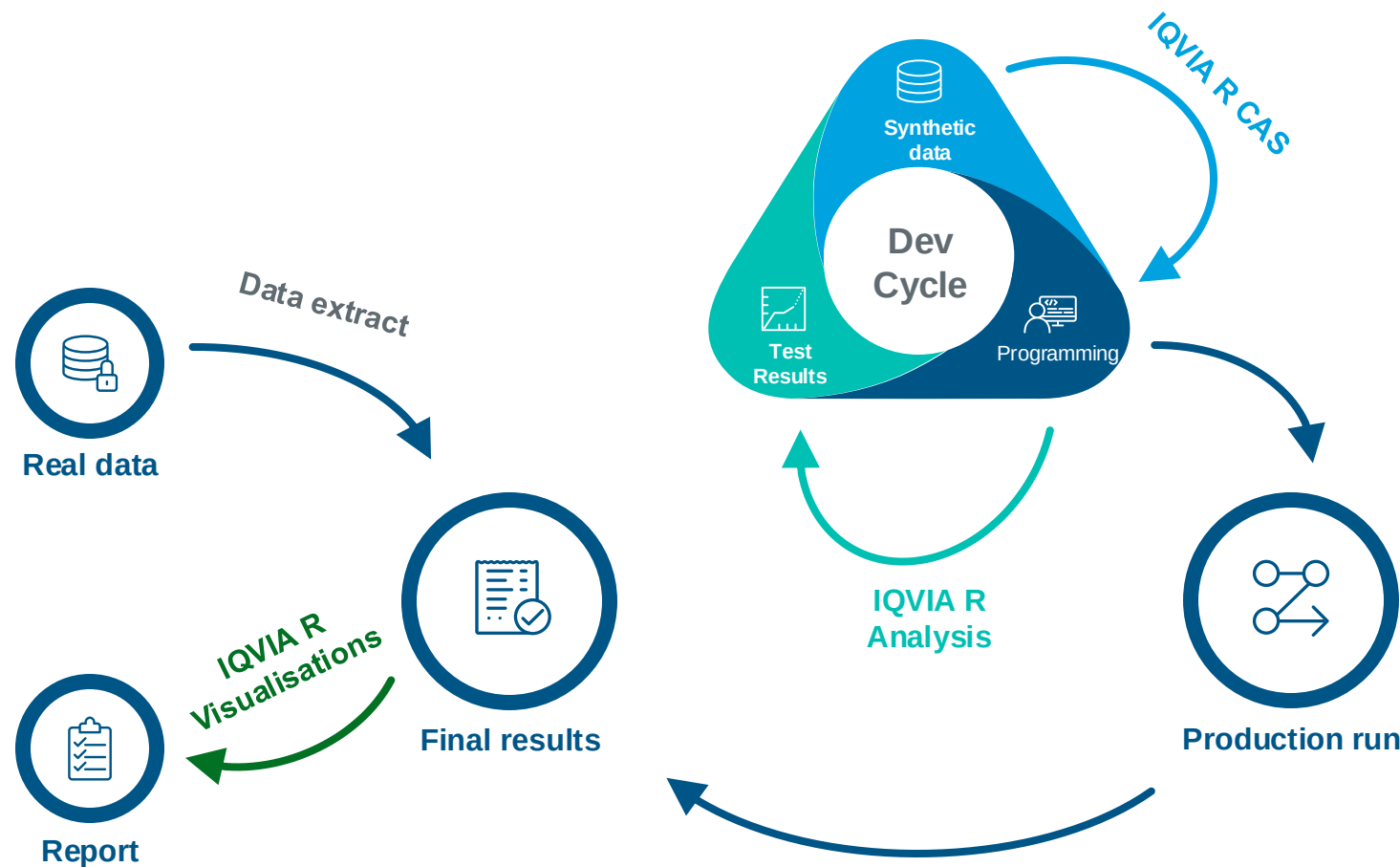
We have successfully implemented our three-stage approach utilising CAS (Cancer Analysis System)

CAS specifics

- Programming code is developed on synthetic data
- Access to real data is granted via virtual “visits”
- Only summary results are available when code is applied to real data

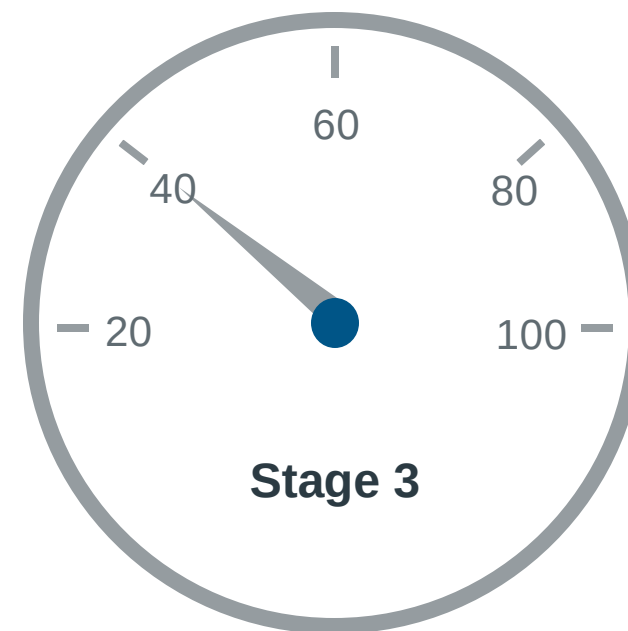
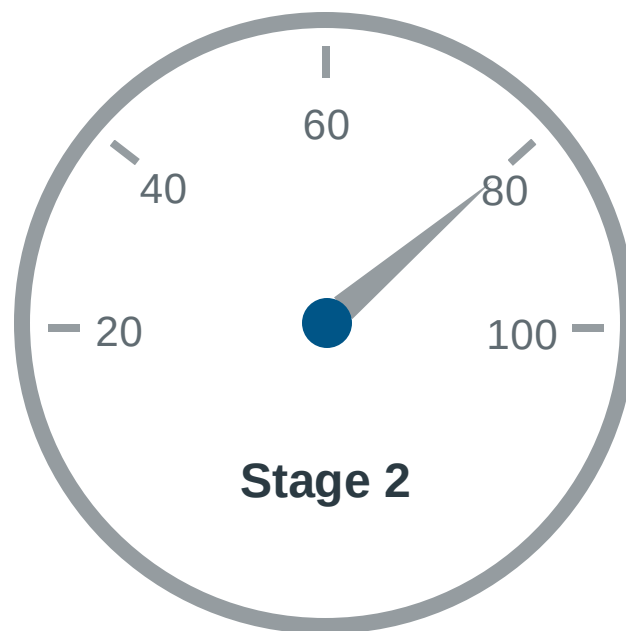
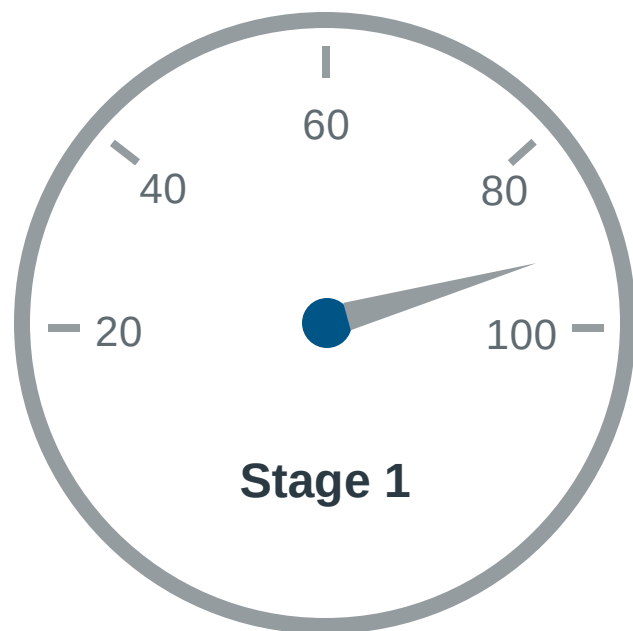
Our solution

- We use **IQVIA R CAS** to load, format and clean both the synthetic and the real data
- We use **IQVIA R Analysis** to analyse the data
- We use **IQVIA R Visualisations** to create additional figures and tables based on summarised outputs



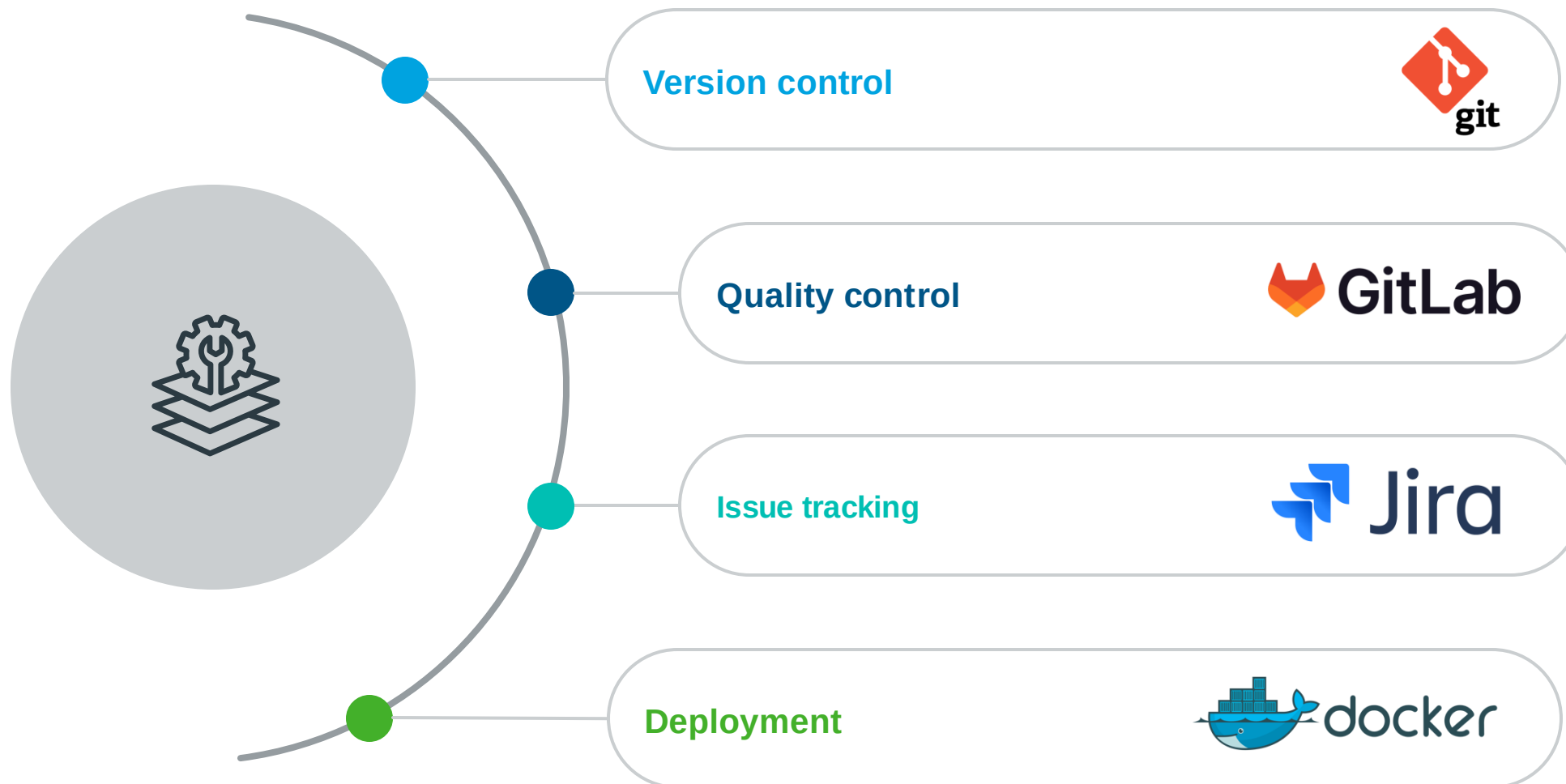
Estimated time savings

Based on our three-stage approach, compared to before its implementation



Package development

Programming best practices are used





Questions?



**Thank you for the attention! To
learn more about us, please
contact:**

alexandrina.lambova2@iqvia.com



Disclaimer

The analyses, their interpretation, and related information contained herein are made and provided subject to the assumptions, methodologies, caveats, and variables described in this report and are based on third party sources and data reasonably believed to be reliable. No warranty is made as to the completeness or accuracy of such third party sources or data.

As with any attempt to estimate future events, the forecasts, projections, conclusions, and other information included herein are subject to certain risks and uncertainties, and are not to be considered guarantees of any particular outcome.

All reproduction rights, quotations, broadcasting, publications reserved. The contents of this presentation are confidential, and no part of this presentation may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without express written consent of IQVIA.

Copyright © 2024 IQVIA. All rights reserved. IQVIA® is a registered trademark of IQVIA Inc. in the United States and various other countries.