



Institut für Qualitätssicherung und
Transparenz im Gesundheitswesen

Leseanleitung für QIDB-Rechenregeln

IQTIG-spezifische Variablen, Operatoren und Funktionen in R

Stand: 01. Juni 2021

Impressum

Thema:

Leseanleitung für QIDB-Rechenregeln. IQTIG-spezifische Variablen, Operatoren und Funktionen in R

Datum der Veröffentlichung:

01. Juni 2021

Herausgeber:

IQTIG – Institut für Qualitätssicherung
und Transparenz im Gesundheitswesen

Katharina-Heinroth-Ufer 1
10787 Berlin

Telefon: (030) 58 58 26-0
Telefax: (030) 58 58 26-999

info@iqtig.org

<https://www.iqtig.org>

Inhaltsverzeichnis

1	Einleitung.....	5
1.1	Grundlegende Konventionen.....	5
1.1.1	Datengrundlage.....	5
1.1.2	Mehrfachfelder	6
1.1.3	Operatorpräzedenz	6
1.1.4	Vorfilter	6
2	Variablen	7
2.1	VB	7
2.2	LST	7
3	Operatoren.....	8
3.1	Gleich-Operator	8
3.2	Ungleich-Operator	8
3.3	Kleiner-Gleich-Operator	9
3.4	Größer-Gleich-Operator.....	9
3.5	Kleiner-Operator	10
3.6	Größer-Operator	10
3.7	%all_in%	11
3.8	%all_like%.....	12
3.9	%any_in%	13
3.10	%any_like%	14
3.11	%between%.....	15
3.12	%group_by%	15
4	Funktionen.....	16
4.1	maximum	16
4.2	minimum	16
4.3	replace_na.....	17
4.4	row_sums.....	17
4.5	to_year	18
4.6	sdat_get.....	19
4.7	sdat_list.....	20

5	Darstellung der Ergebnisse für Vergleichsoperatoren	21
6	Import und Kombination von Ergebnissen (über mehrere Leistungsbereiche hinweg)	23
6.1	Import von Ergebnissen anderer QIs und Kennzahlen	23
6.2	Summen von Ergebnissen anderer QIs und Kennzahlen	23
6.3	Quoten-Indikatoren - Zähler und Nenner aus unterschiedlichen Quellen	25
	Hilfsfunktionen	26
7	Follow-Up-Kennzahlen nach Anlage 3 QSKH.....	29
7.1	Follow-Up Typ "Rate", "O/E" und "Sentinel Event"	29
8	Hilfsfunktionen für O/E-Indikatoren	31

1 Einleitung

Zur Erhöhung der Transparenz und einfachen Reproduzierbarkeit der Ergebnisse durch Dritte werden die Rechenregeln der Qualitätsindikatoren und Kennzahlen des IQTIG seit dem Erfassungsjahr 2018 in R veröffentlicht. R ist eine offene und freie Softwareumgebung für statistische Berechnungen. Um die Lesbarkeit der Rechenregeln zu erhöhen, werden zusätzlich zu den standardmäßig in R vorhandenen Funktionen und Operatoren auch IQTIG-spezifische Funktionen und Operatoren verwendet. Diese werden im vorliegenden Dokument beschrieben. Alle darüber hinaus in Rechenregeln verwendeten Funktionen sind Teil von Standard R.

Bei Fragen können Sie sich gern per E-Mail an den Verfahrenssupport wenden.

Ihr Ansprechpartner:

Institut für Qualitätssicherung und Transparenz im Gesundheitswesen

Katharina-Heinroth-Ufer 1

10787 Berlin

Telefon: (+49) 30 58 58 26 340

Fax: (+49) 30 58 58 26 341

verfahrenssupport@iqtig.org

www.iqtig.org

1.1 Grundlegende Konventionen

1.1.1 Datengrundlage

Generell wird bei der Berechnung davon ausgegangen, dass die jeweilige Datengrundlage als sogenannter *data frame* zur Verfügung steht. Einzelne Variablen beziehen sich somit immer auf eine komplette Spalte.

Beispiel:

ENTLGRUND	ENTLGRUND %==% "01"
"00"	FALSE
"01"	TRUE
"07"	FALSE

1.1.2 Mehrfachfelder

Felder, die in der Spezifikation mehrere Werte annehmen, wie z.B. OPSCHLUESSEL, werden innerhalb der Rechenregeln als eine Listenspalte dargestellt. So wird OPSCHLUESSEL_1 = "A", OPSCHLUESSEL_2 = "B", OPSCHLUESSEL_3 = "C" zu einer einzelnen Spalte OPSCHLUESSEL mit dem Eintrag `list(c("A", "B", "C"))`.

Beispiel:

	FELDNAME	Beispiel INHALT
Richtige Abfrage	OPSCHLUESSEL	<code>c("A", "B", "C")</code>

1.1.3 Operatorpräzedenz

Um eine einheitliche Behandlung von fehlenden Werten zu gewährleisten, werden spezielle Vergleichsoperatoren verwendet. Diese unterscheiden sich einzig durch die Behandlung von fehlenden Werten (sog. NAs). Da diese eine höhere Präzedenz als die üblichen Vergleichsoperatoren haben, müssen bei Verwendung dieser Operatoren zusätzliche Klammern gesetzt werden.

Beispiel:

Ausdruck	Ergebnis
<code>15 >= 10 + 8</code>	FALSE (da „10+8=18“ und damit „Ist 15 größer als 18?“=Falsch)
<code>15 %>=% 10 + 8</code>	TRUE + 8 = 9

1.1.4 Vorfilter

Wenn im Zähler oder Nenner einer Kennzahl eine Formel in der folgenden Art steht, dann ist der Code nach dem „WENN“ als Vorfilter zu lesen.

Beispiel:

Ausdruck	Ergebnis
<code>fn_funktionsname WENN fn_funktionsname %>% 0</code>	Zuerst wird geprüft, ob die Bedingung nach dem „WENN“ erfüllt ist. Nur dann wird der Wert (im Beispiel: Ergebnis einer Funktion) vor dem „WENN“ übernommen, sonst nicht.

2 Variablen

2.1 VB

VB	Eine Variable, über die Ergebnisse von Vorberechnungen abgefragt werden können Dabei ist VB eine Variable, die in R als Liste oder data.frame angelegt werden kann. Sie enthält die Ergebnisse aller Vorberechnungen des jeweiligen QS-Verfahrens.
Syntax	VB\$name
Parameter	name – der Name der Vorberechnung
Rückgabewert	<i>numeric vector</i>
Beispiel	to_year(OPDATUM) %==% VB\$Auswertungsjahr Prüfung ob das Jahr aus dem OP-Datum dem Jahr aus der Vorberechnung mit dem Namen „Auswertungsjahr“ entspricht

2.2 LST

LST	Eine Variable, über die alle Listen angesprochen werden können Es handelt sich um eine Listenvariable, die alle Listen aus der QIDB als character-Vektoren enthält
Syntax	LST\$name
Parameter	name – der Name der Liste
Rückgabewert	<i>character vector</i>
Beispiel	Ansprechen der ICD-Liste „ICD_DekGrad_2“: LST\$ICD_DekGrad_2

3 Operatoren

Zusätzlich zu Standard-R-Konstruktionen verwenden die Rechenregeln des IQTIG auch spezielle Funktionen und Operatoren (Benutzerdefinierte Operatoren). Diese Operatoren unterscheiden sich von den Standard-R-Operatoren dahingehend, dass NA-Werte anders behandelt werden.

3.1 Gleich-Operator

%==%	Gleich-Operator									
Syntax	lhs %==% rhs									
Parameter	lhs – Ein Vektor rhs – Ein Vektor gleicher Länge wie <lhs>									
Rückgabewert	Logical Vektor mit der Länge gleich der Länge von <lhs>									
Besonderheiten	Dieser Operator unterscheidet sich von den Standard R Operatoren dahingehend, dass NA-Werte anders behandelt werden.									
Beispiele	<table><tr><td>2 %==% 2</td><td>TRUE</td></tr><tr><td>2 %==% 3</td><td>FALSE</td></tr><tr><td>2 %==% NA</td><td>FALSE</td></tr><tr><td>NA %==% NA</td><td>TRUE</td></tr></table>		2 %==% 2	TRUE	2 %==% 3	FALSE	2 %==% NA	FALSE	NA %==% NA	TRUE
2 %==% 2	TRUE									
2 %==% 3	FALSE									
2 %==% NA	FALSE									
NA %==% NA	TRUE									

3.2 Ungleich-Operator

%!=%	Ungleich-Operator									
Syntax	lhs %!=% rhs									
Parameter	lhs – Ein Vektor rhs – Ein Vektor gleicher Länge wie <lhs>									
Rückgabewert	Logical Vektor mit der Länge gleich der Länge von <lhs>									
Besonderheiten	Dieser Operator unterscheidet sich von den Standard R Operatoren dahingehend, dass NA-Werte anders behandelt werden.									
Beispiele	<table><tr><td>2 %!=% 1</td><td>TRUE</td></tr><tr><td>2 %!=% 2</td><td>FALSE</td></tr><tr><td>2 %!=% NA</td><td>TRUE</td></tr><tr><td>NA %!=% NA</td><td>FALSE</td></tr></table>		2 %!=% 1	TRUE	2 %!=% 2	FALSE	2 %!=% NA	TRUE	NA %!=% NA	FALSE
2 %!=% 1	TRUE									
2 %!=% 2	FALSE									
2 %!=% NA	TRUE									
NA %!=% NA	FALSE									

3.3 Kleiner-Gleich-Operator

%<=%	Kleiner-Gleich-Operator								
Syntax	lhs %<=% rhs								
Parameter	lhs – Ein Vektor rhs – Ein Vektor gleicher Länge wie <lhs>								
Rückgabewert	Logical Vektor mit der Länge gleich der Länge von <lhs>								
Besonderheiten	Dieser Operator unterscheidet sich von den Standard R Operatoren dahingehend, dass NA-Werte anders behandelt werden.								
Beispiele	<table border="1"> <tr> <td>2 %<=% 3</td><td>TRUE</td></tr> <tr> <td>2 %<=% 1</td><td>FALSE</td></tr> <tr> <td>2 %<=% NA</td><td>FALSE</td></tr> <tr> <td>NA %<=% NA</td><td>TRUE</td></tr> </table>	2 %<=% 3	TRUE	2 %<=% 1	FALSE	2 %<=% NA	FALSE	NA %<=% NA	TRUE
2 %<=% 3	TRUE								
2 %<=% 1	FALSE								
2 %<=% NA	FALSE								
NA %<=% NA	TRUE								

3.4 Größer-Gleich-Operator

%>=%	Größer-Gleich-Operator								
Syntax	lhs %>=% rhs								
Parameter	lhs – Ein Vektor rhs – Ein Vektor gleicher Länge wie <lhs>								
Rückgabewert	Logical Vektor mit der Länge gleich der Länge von <lhs>								
Besonderheiten	Dieser Operator unterscheidet sich von den Standard R Operatoren dahingehend, dass NA-Werte anders behandelt werden.								
Beispiele	<table border="1"> <tr> <td>2 %>=% 3</td><td>FALSE</td></tr> <tr> <td>2 %>=% 1</td><td>TRUE</td></tr> <tr> <td>2 %>=% NA</td><td>FALSE</td></tr> <tr> <td>NA %>=% NA</td><td>TRUE</td></tr> </table>	2 %>=% 3	FALSE	2 %>=% 1	TRUE	2 %>=% NA	FALSE	NA %>=% NA	TRUE
2 %>=% 3	FALSE								
2 %>=% 1	TRUE								
2 %>=% NA	FALSE								
NA %>=% NA	TRUE								

3.5 Kleiner-Operator

%<%	Kleiner-Operator								
Syntax	lhs %<% rhs								
Parameter	lhs – Ein Vektor rhs – Ein Vektor gleicher Länge wie <lhs>								
Rückgabewert	Logical Vektor mit der Länge gleich der Länge von <lhs>								
Besonderheiten	Dieser Operator unterscheidet sich von den Standard R Operatoren dahingehend, dass NA-Werte anders behandelt werden.								
Beispiele	<table border="1"> <tr> <td>2 %<% 3</td><td>TRUE</td></tr> <tr> <td>2 %<% 1</td><td>FALSE</td></tr> <tr> <td>2 %<% NA</td><td>FALSE</td></tr> <tr> <td>NA %<% NA</td><td>FALSE</td></tr> </table>	2 %<% 3	TRUE	2 %<% 1	FALSE	2 %<% NA	FALSE	NA %<% NA	FALSE
2 %<% 3	TRUE								
2 %<% 1	FALSE								
2 %<% NA	FALSE								
NA %<% NA	FALSE								

3.6 Größer-Operator

%>%	Größer-Operator								
Syntax	lhs %>% rhs								
Parameter	lhs – Ein Vektor rhs – Ein Vektor gleicher Länge wie <lhs>								
Rückgabewert	Logical Vektor mit der Länge gleich der Länge von <lhs>								
Besonderheiten	Dieser Operator unterscheidet sich von den Standard R Operatoren dahingehend, dass NA-Werte anders behandelt werden.								
Beispiele	<table border="1"> <tr> <td>2 %>% 3</td><td>FALSE</td></tr> <tr> <td>2 %>% 1</td><td>TRUE</td></tr> <tr> <td>2 %>% NA</td><td>FALSE</td></tr> <tr> <td>NA %>% NA</td><td>FALSE</td></tr> </table>	2 %>% 3	FALSE	2 %>% 1	TRUE	2 %>% NA	FALSE	NA %>% NA	FALSE
2 %>% 3	FALSE								
2 %>% 1	TRUE								
2 %>% NA	FALSE								
NA %>% NA	FALSE								

3.7 %all_in%

%all_in%	Prüft, ob alle Elemente der linken Seite in der Menge der rechten Seite vorhanden sind.	
Syntax	lhs %all_in% rhs	
Parameter	lhs - Ein Character-Vektor oder eine Liste von Character-Vektoren. rhs - Ein Character-Vektor.	
Rückgabewert	Logical Vektor mit der Länge gleich der Länge von <lhs>	
Besonderheiten	Falls die linke Seite die Länge 0 hat (z. B. weil in einer Mehrfachauswahl kein Wert angegeben wurde), ist das Ergebnis immer TRUE.	
Beispiele	OPSCHLUESSEL_1 <- "5-652.40" OPSCHLUESSEL_2 <- "5-303.20" OPSCHLUESSEL_3 <- NA OPS_GEB_VAG_OP <- c("5-652.40", "5-303.20", "5-680.12")	
	OPSCHLUESSEL %all_in% LST\$OPS_GEB_VAG_OP	TRUE
	OPSCHLUESSEL %all_in% c("5-652.40", "5-303.20", "5-680.12")	TRUE
	list(c("a", "b"), "c") %all_in% c("a", "c")	c(FALSE, TRUE)
	list(character(0)) %all_in% c("a", "b", "c")	TRUE

3.8 %all_like%

%all_like%	Prüft, ob alle Elemente der linken Seite mit mindestens einem Element der rechten Seite „gematched“ werden können				
Syntax	lhs %all_like% rhs				
Parameter	lhs - Ein Character-Vektor oder eine Liste von Character-Vektoren. rhs - Ein Character-Vektor. Die einzelnen Strings können Wildcard Operatoren "%" enthalten.				
Rückgabewert	Logical Vektor mit der Länge gleich der Länge von <lhs>				
Besonderheiten	Es werden hier SQL-ähnliche reguläre Ausdrücke verwendet, die nur die Wildcard "%" kennen. Falls die linke Seite die Länge 0 hat (z. B. weil in einer Mehrfachauswahl kein Wert angegeben wurde), ist das Ergebnis immer TRUE.				
Beispiel	<pre>OPSCHLUESSEL_1 <- "5-12.32" OPSCHLUESSEL_2 <- "5-33.20:L" OPSCHLUESSEL_3 <- NA OPS_GEB_VAG_OP <- c("5-12%", "5-33.2%", "5-652.40:R")</pre> <table border="1"> <tr> <td>OPSCHLUESSEL %all_like% LST\$OPS_GEB_VAG_OP</td><td>TRUE</td></tr> <tr> <td>list(character(0)) %all_like% c("a%", "b%", "c%")</td><td>TRUE</td></tr> </table>	OPSCHLUESSEL %all_like% LST\$OPS_GEB_VAG_OP	TRUE	list(character(0)) %all_like% c("a%", "b%", "c%")	TRUE
OPSCHLUESSEL %all_like% LST\$OPS_GEB_VAG_OP	TRUE				
list(character(0)) %all_like% c("a%", "b%", "c%")	TRUE				

3.9 %any_in%

%any_in%	Prüft, ob mindestens ein Element der linken Seite in der Menge der rechten Seite vorhanden ist.				
Syntax	lhs %any_in% rhs				
Parameter	lhs - Ein Character-Vektor oder eine Liste von Character-Vektoren. rhs - Ein Character-Vektor.				
Rückgabewert	Logical Vektor mit der Länge gleich der Länge von <lhs>				
Beispiele	<pre> OPSCHLUESSEL_1 <- "5-652.40:R" OPSCHLUESSEL_2 <- "5-744.5x" OPSCHLUESSEL_3 <- NA OPS_GEB_VAG_OP <- c("5-657.6x", "5-653.35", "5-652.40:R") </pre> <table border="1"> <tr> <td>OPSCHLUESSEL %any_in% LST\$OPS_GEB_VAG_OP</td><td>TRUE</td></tr> </table> <pre> OPSCHLUESSEL_1 <- "5-652.40" OPSCHLUESSEL_2 <- "5-744.5x" OPSCHLUESSEL_3 <- NA OPS_GEB_VAG_OP <- c("5-657.6x", "5-653.35", "5-652.40:R") </pre> <table border="1"> <tr> <td>OPSCHLUESSEL %any_in% LST\$OPS_GEB_VAG_OP</td><td>FALSE</td></tr> </table>	OPSCHLUESSEL %any_in% LST\$OPS_GEB_VAG_OP	TRUE	OPSCHLUESSEL %any_in% LST\$OPS_GEB_VAG_OP	FALSE
OPSCHLUESSEL %any_in% LST\$OPS_GEB_VAG_OP	TRUE				
OPSCHLUESSEL %any_in% LST\$OPS_GEB_VAG_OP	FALSE				

3.10 %any_like%

%any_like%	Prüft ob mindestens ein Element der linken Seite mit mindestens einem Element der rechten Seite „gematched“ werden kann.		
Syntax	lhs %any_like% rhs		
Parameter	lhs - Ein Character-Vektor oder eine Liste von Character-Vektoren. rhs - Ein Character-Vektor. Die einzelnen Strings können Wildcard Operatoren "%" enthalten.		
Rückgabewert	Logical Vektor mit der Länge gleich der Länge von <lhs>		
Besonderheiten	Es werden hier SQL-ähnliche reguläre Ausdrücke verwendet, die nur die Wildcard "%" kennen.		
Beispiel	<pre>OPSCHLUESSEL_1 <- "5-652.40:R" OPSCHLUESSEL_2 <- "5-744.5x" OPSCHLUESSEL_3 <- NA OPS_GEB_VAG_OP <- c("5-657.6x", "5-653.35", "5-652.40%")</pre> <table border="1"> <tr> <td>OPSCHLUESSEL %any_like% LST\$OPS_GEB_VAG_OP</td><td>TRUE</td></tr> </table>	OPSCHLUESSEL %any_like% LST\$OPS_GEB_VAG_OP	TRUE
OPSCHLUESSEL %any_like% LST\$OPS_GEB_VAG_OP	TRUE		

3.11 %between%

%between%	Prüft, ob Werte der linken Seite zwischen den zwei Werten der rechten Seite liegen.						
Syntax	lhs %between% rhs						
Parameter	lhs – Ein Vektor beliebigen Typs rhs – Ein Vektor der Länge 2 mit gleichem Typ wie <lhs>						
Rückgabewert	Logical Vektor mit der Länge gleich der Länge von <lhs>						
Besonderheiten	Äquivalent zu lhs >= rhs[1] & lhs <= rhs[2]. D.h. der Wert in lhs kann auch auf den Grenzen des Intervalls liegen. Dieser Operator liefert immer TRUE oder FALSE zurück, auch wenn lhs ein NA-Wert ist.						
Beispiele	<table border="1"> <tr> <td>5 %between% c(1, 10)</td><td>TRUE</td></tr> <tr> <td>alter <- 39</td><td>FALSE</td></tr> <tr> <td>alter %between% c(40, 50)</td><td></td></tr> </table>	5 %between% c(1, 10)	TRUE	alter <- 39	FALSE	alter %between% c(40, 50)	
5 %between% c(1, 10)	TRUE						
alter <- 39	FALSE						
alter %between% c(40, 50)							

3.12 %group_by%

%group_by%	Evaluation einer Funktion, welche nach einer Menge von Feldern gruppiert wurde.
Syntax	lhs %group_by% rhs
Parameter	lhs – Code, der für jede Gruppe evaluiert wird. Er sollte typstabil sein und entweder einen Wert der Länge 1 oder einen Vektor in Länge der Anzahl Zeilen der jeweiligen Gruppe zurückgeben. rhs – Ein Name oder ein Vektor von Namen, die Spalten des globalen Datensatzes entsprechen.
Rückgabewert	Ein Vektor, abhängig vom Typ von <lhs>. Die Länge des Vektors ist immer gleich der Anzahl Zeilen des kompletten Datensatzes.
Besonderheiten	%group_by% bezieht sich immer implizit auf einen globalen Datensatz
Beispiel	minimum(OPDATUM) %group_by% TDS_B Erklärung: Ermittlung des kleinsten OP-Datums aller dem Basisbogen untergeordneter Bögen

4 Funktionen

4.1 maximum

maximum	Ermittelt das Maximum eines Vektors.							
Syntax	maximum(x)							
Parameter	x - ein Vektor vom Typ integer, character, Date, POSIXct, POSIXlt, oder logical							
Rückgabewert	Vektor der Länge 1 vom Typ des Eingabe-Parameters							
Besonderheiten	Unterschied zu base R: - NA Werte werden ignoriert - Maximum eines leeren Vektors ist immer NA Es wird versucht, typischer NAs zurück zu geben. Bei DateTime Typen, wird die Zeitzone aus dem Attribut "tzone" auf einen möglichen NA Rückgabewert angewendet.							
Beispiele	<table><tr><td>maximum(c(1, 2, 3))</td><td>3</td></tr><tr><td>maximum(NA)</td><td>NA</td></tr><tr><td>maximum(character(0L))</td><td>NA_character_</td></tr></table>		maximum(c(1, 2, 3))	3	maximum(NA)	NA	maximum(character(0L))	NA_character_
maximum(c(1, 2, 3))	3							
maximum(NA)	NA							
maximum(character(0L))	NA_character_							

4.2 minimum

minimum	Ermittelt das Minimum eines Vektors.							
Syntax	minimum(x)							
Parameter	x - ein Vektor vom Typ integer, character, Date, POSIXct, POSIXlt, oder logical							
Rückgabewert	Vektor der Länge 1 vom Typ den Eingabe-Parameters							
Besonderheiten	Unterschied zu base R: - NA Werte werden ignoriert - Maximum eines leeren Vektors ist immer NA Es wird versucht, typischer NAs zurück zu geben. Bei DateTime Typen, wird die Zeitzone aus dem Attribut "tzone" auf einen möglichen NA Rückgabewert angewendet.							
Beispiele	<table><tr><td>minimum(c(1, 2, 3))</td><td>1</td></tr><tr><td>minimum(NA)</td><td>NA</td></tr><tr><td>minimum(character(0L))</td><td>NA_character</td></tr></table>		minimum(c(1, 2, 3))	1	minimum(NA)	NA	minimum(character(0L))	NA_character
minimum(c(1, 2, 3))	1							
minimum(NA)	NA							
minimum(character(0L))	NA_character							

4.3 replace_na

replace_na	Ersetzt NA-Werte durch einen anderen Wert.				
Syntax	replace_na(x, by)				
Parameter	x – ein Vektor beliebiger Länge by – ein Wert der Länge 1 mit gleichem Typ wie <x>				
Rückgabewert	Ein Vektor mit gleicher Länge und gleichem Typ wie <x>. Alle NA-Werte sind durch <by> ersetzt.				
Beispiele	<table border="1"> <tr> <td>replace_na(alter, 20)</td><td>Wenn Feld „Alter“ leer ist, wird es durch 20 ersetzt</td></tr> <tr> <td>replace_na(c(20, NA, 10), 15)</td><td>c(20, 15, 10)</td></tr> </table>	replace_na(alter, 20)	Wenn Feld „Alter“ leer ist, wird es durch 20 ersetzt	replace_na(c(20, NA, 10), 15)	c(20, 15, 10)
replace_na(alter, 20)	Wenn Feld „Alter“ leer ist, wird es durch 20 ersetzt				
replace_na(c(20, NA, 10), 15)	c(20, 15, 10)				

4.4 row_sums

row_sums	Zeilenweise Summe mehrerer Vektoren				
Syntax	row_sums(Vektor1, Vektor2, ..., VektorN)				
Parameter	... - Vektoren gleicher Länge				
Rückgabewert	<i>numeric vector</i>				
Besonderheiten	NA-Werte werden entfernt. Äquivalent zu <code>rowSums(cbind(...), na.rm = TRUE)</code> Es können auch <i>logical vectors</i> verarbeitet werden. <i>TRUE</i> wird dann bei der Summenbildung als 1 gezählt und <i>FALSE</i> als 0				
Beispiele	GEHTRAINING = 1 HILFSMITTEL = 3 MEDIKATION = NA <table border="1"> <tr> <td>row_sums(GEHTRAINING, HILFSMITTEL, MEDIKATION)</td><td>4</td></tr> </table> ----- bf_Reizschwelle_VO = TRUE bf_Reizschwelle_VE1 = FALSE <table border="1"> <tr> <td>row_sums(bf_Reizschwelle_VO, bf_Reizschwelle_VE1)</td><td>1</td></tr> </table>	row_sums(GEHTRAINING, HILFSMITTEL, MEDIKATION)	4	row_sums(bf_Reizschwelle_VO, bf_Reizschwelle_VE1)	1
row_sums(GEHTRAINING, HILFSMITTEL, MEDIKATION)	4				
row_sums(bf_Reizschwelle_VO, bf_Reizschwelle_VE1)	1				

4.5 to_year

to_year	Extrahiert die Jahreszahl aus einem Character Vektor							
Syntax	to_year(x)							
Parameter	x – Ein <i>character vector</i> oder ein Vektor, der in einen <i>character vector</i> konvertiert werden kann oder ein Vektor vom Type Date							
Rückgabewert	Ein Integer Vektor von Jahreszahlen oder <i>NA_integer</i>							
Besonderheiten	Es werden drei Muster unterstützt: • mm.yyyy • [1-4]/yyyy • yyyy Sollten nicht alle Elemente des Vektors eines dieser Muster erfüllen, so wird angenommen, dass es sich bei <x> um ein <i>Date</i> handelt.							
Beispiele	<table><tr><td>to_year("12.2015")</td><td>2015L</td></tr><tr><td>to_year("1.2014")</td><td>2014L</td></tr><tr><td>to_year(c(NA_character, "2014"))</td><td>c(NA_integer, 2014L)</td></tr></table>		to_year("12.2015")	2015L	to_year("1.2014")	2014L	to_year(c(NA_character, "2014"))	c(NA_integer, 2014L)
to_year("12.2015")	2015L							
to_year("1.2014")	2014L							
to_year(c(NA_character, "2014"))	c(NA_integer, 2014L)							

4.6 sdat_get

sdat_get	Durchsucht die zu einem QS-Bogen zugeordneten Sozialdaten	
Syntax	sdat_get(x, from, filter = NULL)	
Parameter	• x - Der Name des Datenfeldes aus den Sozialdaten, das zurückgegeben werden soll. • from - Ein Character-Vektor mit Listennamen oder der Name einer konkreten Liste als Symbol. Dieser Parameter gibt an, welche Sozialdatenlisten durchsucht werden sollen. • filter - Eine Filterbedingung, die die durchsuchten Sozialdaten-Informationen einschränkt.	
Rückgabewert	Eine Liste mit den Werten von x für die Fälle, die die Filterbedingungen erfüllen.	
Besonderheiten	Zu jedem QS-Bogen sind i. d. R. alle Sozialdaten-Informationen der betreffenden Patienten bzw. des betreffenden Patienten zugeordnet.	
Beispiele	<pre>sdat_get("fall_id", sdat_301_ops, code %any_like% c("5.12%", "5.13%"))</pre>	Gibt die IDs sämtlicher Fälle der Patientin bzw. des Patienten zurück, die einen OPS-Code enthalten, der mit "5.12" oder "5.13" beginnt
	<pre>lengths(sdat_get("fall_id", sdat_301_ops, code %any_like% LST\$OPS_CHE_Reintervention)) > 0L</pre>	Wurde mindestens ein OPS-Code aus der Liste LST\$OPS_CHE_Reintervention in den Sozialdaten dokumentiert?
	<pre>sdat_get("fall_id", sdat_301_ops, (datum <= OPDATUM + 90) & (datum > OPDATUM) & (code %any_like% LST\$OPS_CHE_Reintervention))</pre>	Gibt die IDs sämtlicher Fälle zurück, bei denen ein OPS-Code aus der Liste LST\$OPS_CHE_Reintervention im Zeitraum von 90 Tagen nach OPDATUM dokumentiert wurde.
	<pre>sdat_get("code", "301_icd", (datum >= ENTLDATUM) & (datum <= OPDATUM + 7))</pre>	Eine Liste von ICD-Codes nach §301 SGB V, welche ein Diagnosedatum zwischen (jeweils einschließlich) dem Entlassdatum und OP-Datum + 7 Tage haben. Äquivalent zu: sdat_list("301_icd", ENTLDATUM, OPDATUM + 7)

4.7 sdat_list

sdat_list	Schränkt Sozialdatenkodelisten pro QS Bogen nach Datums- und sonstigen Filtern ein.	
Syntax	sdat_list(list_names, date_from, date_end, filter = NULL)	
Parameter	- list_names - Ein Character Vektor mit Listennamen oder der Name einer konkreten Liste als Symbol. Dieser Parameter gibt an, welche Sozialdatenlisten eingeschränkt werden sollen. - date_from - Jeder Sozialdatenkode hat ein Datum. Dieses Datum muss gleich date_from oder später sein. - date_end - Jedes Datum muss früher oder gleich date_end sein. - filter - Neben dem Datum können Sozialdaten noch weitere Eigenschaften wie z. B. die Diagnoseart besitzen. Die interessierenden Kodes können über ein optionales Filterargument eingeschränkt werden.	
Rückgabewert	Eine Liste mit Kodes, die die Bedingungen (Datumseinschränkungen und optionaler Filter) erfüllen. Diese Kodelisten werden dann in der Regel mit ICD/OPS Listen abgeglichen.	
Besonderheiten	Zu jedem QS-Bogen sind i. d. R. alle Sozialdaten-Informationen der betreffenden Patienten bzw. des betreffenden Patienten zugeordnet.	
Beispiele	sdat_list("301_icd", ENTLDATE, OPDATE + 7)	Eine Liste von ICD-Kodes nach §301 SGB V, welche ein Diagnosedatum zwischen (jeweils einschließlich) dem Entlassdatum und OP-Datum + 7 Tage haben.
	sdat_list(sdat_301_icd, ENTLDATE, OPDATE + 7)	Äquivalent zum Ausdruck oben. Die Verwendung der Variable als Symbol hat in der Regel technische Gründe.
	sdat_list("301_ops", OPDATE, OPDATE + 7) %any_like% LST\$OPS_PCI_Erneut	Jeder OPS-Kode, der die Datumseinschränkung erfüllt, wird mit einer Liste von OPS-Kodes verglichen. Ist der Kode Teil dieser Liste, so wird TRUE zurückgegeben, ansonsten FALSE.
	sdat_list("301_icd", OPDATE , OPDATE + 7, diagnoseart %==% 1) %any_like% LST\$ICD_PCI_- AneurysmaSpurium	Ähnlich zum Beispiel oben, allerdings müssen die ICD-Kodes noch Einschränkung "diagnoseart %==% 1" erfüllen, bevor sie mit der entsprechenden ICD-Liste abgeglichen werden.

5 Darstellung der Ergebnisse für Vergleichsoperatoren

Die folgenden Tabellen zeigen beispielhaft, wie aktuell bei der Berechnung von Qualitätsindikatoren mit NA Werten umgegangen wird. Es gilt für alle Tabellen

- a und b sind vom Typ *numeric*
und
- „a ist kleiner b“

%==%	a	b	NA
a	TRUE	FALSE	FALSE
b	FALSE	TRUE	FALSE
NA	FALSE	FALSE	TRUE

%!=%	a	b	NA
a	FALSE	TRUE	TRUE
b	TRUE	FALSE	TRUE
NA	TRUE	TRUE	FALSE

%<=%	a	b	NA
a	TRUE	TRUE	FALSE
b	FALSE	TRUE	FALSE
NA	FALSE	FALSE	TRUE

%>=%	a	b	NA
a	TRUE	FALSE	FALSE
b	TRUE	TRUE	FALSE
NA	FALSE	FALSE	TRUE

%<%	a	b	NA
a	FALSE	TRUE	FALSE
b	FALSE	FALSE	FALSE
NA	FALSE	FALSE	FALSE

%>%	a	b	NA
a	FALSE	FALSE	FALSE
b	TRUE	FALSE	FALSE
NA	FALSE	FALSE	FALSE

6 Import und Kombination von Ergebnissen (über mehrere Leistungsbereiche hinweg)

Mit den folgenden Funktionen können QIs und Kennzahlen definiert werden, die Ergebnisse anderer Kennzahlen direkt weiterverwenden.

6.1 Import von Ergebnissen anderer QIs und Kennzahlen

import_indicator	Importiert das Ergebnis eines anderen QIs bzw. einer anderen Kennzahl.
Syntax	<code>import_indicator(module, id)</code>
Parameter	<code>module</code> – Name des Leistungsbereiches, aus dem der QI bzw. die Kennzahl importiert werden soll (character). <code>id</code> – ID des zu importierenden QIs bzw. der zu importierenden Kennzahl (character).
Rückgabewert	Das Ergebnis des importierten QIs bzw. der importierten Kennzahl als <code>data.frame</code> .
Besonderheiten	-
Beispiel	Risikostatistik: Anzahl aller Fälle: <pre>import_indicator(module = "RST", id = "24851")</pre>

6.2 Summen von Ergebnissen anderer QIs und Kennzahlen

sum_indicator	Summiert die Ergebnisse anderer QIs bzw. anderer Kennzahlen auf.
Syntax	<code>sum_indicator(list(module, id), list(module, id), ...)</code>
Parameter	<code>list(module, id)</code> – Eine beliebige Anzahl von Referenzen auf andere QIs bzw. Kennzahlen. Die Parameter <code>list</code> und <code>module</code> sind dabei analog zu den Parametern von <code>import_indicator</code> definiert.
Rückgabewert	Die Summe der Ergebnisse der angegebenen QIs bzw. Kennzahlen. Dabei werden Zähler und Nenner getrennt aufsummiert. Das Ergebnis wird dann als Quotient von Zähler und Nenner berechnet. Alle Leistungserbringer, die mindestens für ein Element ein Ergebnis aufweisen, werden Teil des Ergebnisdatensatzes.
Besonderheiten	-

Beispiel	<pre>ID 52307 (09/2) - Index: sum_indicator(list(module = "09/1", id = "52312"), list(module = "09/2", id = "52313"), list(module = "09/3", id = "52314"))</pre>
----------	--

6.3 Quoten-Indikatoren - Zähler und Nenner aus unterschiedlichen Quellen

quotient_indicator	Quoten-Indikator, für den Zähler und Nenner aus unterschiedlichen Berechnungen auf unterschiedlichen Datensätzen stammen.										
Syntax	<code>quotient_indicator(numerator, denominator, expected = NULL, units_from = "union")</code>										
Parameter	numerator – Zähler-Ergebnisse auf Leistungserbringer-Ebene in einem data.frame. Die unten aufgeführten Hilfsfunktionen stehen hierfür zur Verfügung. denominator – Nenner-Ergebnisse auf Leistungserbringer-Ebene in einem data.frame. Die unten aufgeführten Hilfsfunktionen stehen hierfür zur Verfügung. expected – Ergebnisse für die erwartete Anzahl interessierender Ereignisse (E) in einem data.frame (optional). Die unten aufgeführten Hilfsfunktionen stehen hierfür zur Verfügung. units_from – Gibt an, für welche Leistungserbringer der Quoten-Indikator Ergebnisse liefert (character). Möglich sind: <table border="1"> <thead> <tr> <th>Wert</th><th>Bedeutung</th></tr> </thead> <tbody> <tr> <td>"union"</td><td>Default. Alle Leistungserbringer, für die entweder in denominator ein Ergebnis für den Nenner oder in numerator ein Ergebnis für den Zähler vorliegt (entspricht full join).</td></tr> <tr> <td>"intersection"</td><td>Alle Leistungserbringer, für die sowohl in denominator ein Ergebnis für den Nenner als auch in numerator ein Ergebnis für den Zähler vorliegt (entspricht inner join).</td></tr> <tr> <td>"denominator"</td><td>Alle Leistungserbringer, für die in denominator ein Ergebnis für den Nenner vorliegt (entspricht left bzw. right join).</td></tr> <tr> <td>"numerator"</td><td>Alle Leistungserbringer, für die in numerator ein Ergebnis für den Zähler vorliegt (entspricht right bzw. left join).</td></tr> </tbody> </table>	Wert	Bedeutung	"union"	Default. Alle Leistungserbringer, für die entweder in denominator ein Ergebnis für den Nenner oder in numerator ein Ergebnis für den Zähler vorliegt (entspricht full join).	"intersection"	Alle Leistungserbringer, für die sowohl in denominator ein Ergebnis für den Nenner als auch in numerator ein Ergebnis für den Zähler vorliegt (entspricht inner join).	"denominator"	Alle Leistungserbringer, für die in denominator ein Ergebnis für den Nenner vorliegt (entspricht left bzw. right join).	"numerator"	Alle Leistungserbringer, für die in numerator ein Ergebnis für den Zähler vorliegt (entspricht right bzw. left join).
Wert	Bedeutung										
"union"	Default. Alle Leistungserbringer, für die entweder in denominator ein Ergebnis für den Nenner oder in numerator ein Ergebnis für den Zähler vorliegt (entspricht full join).										
"intersection"	Alle Leistungserbringer, für die sowohl in denominator ein Ergebnis für den Nenner als auch in numerator ein Ergebnis für den Zähler vorliegt (entspricht inner join).										
"denominator"	Alle Leistungserbringer, für die in denominator ein Ergebnis für den Nenner vorliegt (entspricht left bzw. right join).										
"numerator"	Alle Leistungserbringer, für die in numerator ein Ergebnis für den Zähler vorliegt (entspricht right bzw. left join).										
Rückgabewert	Das Ergebnis des Quoten-Indikators data.frame.										
Besonderheiten	-										

Hilfsfunktionen

Die folgenden Hilfsfunktionen können verwendet werden, um die Parameter denominator und numerator zu bestimmen. Sie unterscheiden sich von den obigen Funktionen technisch dahingehend, dass sie stets ein data.frame auf Leistungserbringer-Ebene (d.h. eine Zeile pro Leistungserbringer) zurückgeben. Die Ermittlung eines Gesamtergebnisses durch Zusammenfügen von Zähler und Nenner und Summation findet erst nachgelagert in der Funktion quotient_indicator statt. Je nachdem, ob das Ergebnis in quotient_indicator als Zähler, Nenner oder E eingebunden wird, findet auch nur dieser Teil des Ergebnisses dort Verwendung.

evaluate	Wertet eine Rechenregel auf dem Datensatz des aktuellen Moduls mit dem Teildatensatzbezug des aktuellen Indikators aus.
Syntax	evaluate(code, module)
Parameter	code – Die zu evaluierende Rechenregel. module – Der Leistungsbereich, auf dessen regulärem Datensatz die Rechenregel evaluiert wird. Falls nicht spezifiziert, wird der Datensatz des Leistungsbereiches des aktuellen QIs bzw. der aktuellen Kennzahl genutzt.
Rückgabewert	Das Ergebnis der Ausführung der Rechenregel auf Leistungserbringer-Ebene (Anzahl Teildatensätze, für die die Rechenregel auf "wahr" evaluiert).
Besonderheiten	-

import_results	Importiert das Ergebnis eines anderen QIs bzw. einer anderen Kennzahl auf Leistungserbringer-Ebene.
Syntax	import_results(module, id)
Parameter	module – Name des Leistungsbereiches, aus dem der QI bzw. die Kennzahl importiert werden soll (character). id – ID des zu importierenden QIs bzw. der zu importierenden Kennzahl (character).
Rückgabewert	Das Ergebnis des importierten QIs bzw. der importierten Kennzahl als data.frame auf Leistungserbringer-Ebene.
Besonderheiten	-

sum_results	Summiert Ergebnisse auf Leistungserbringer-Ebene auf.
Syntax	sum_results(...)
Parameter	... - Eine beliebige Anzahl an Leistungserbringer-Ergebnissen als data.frames. Diese können beispielsweise mit import_results erzeugt werden.
Rückgabewert	Die Summe der Ergebnisse der angegebenen QIs bzw. Kennzahlen als data.frame auf Leistungserbringer-Ebene. Dabei werden Zähler und Nenner getrennt aufsummiert.
Besonderheiten	-

Beispiele	ID 52001 (09/6) - Prozedurassoziiertes Problem als Indikation zum Folgeeingriff: <pre> numerator <- evaluate(ORTLETZTEOP %in% c(1, 3) & ((TASCHENPROBLEM %in% c(1, 9) & bf_LaufzeitICDAggregat %in% c(0, 1)) bf_FruehKomplVorhofsonde bf_FruehKomplVentrikelsonde1 bf_FruehKomplVentrikelsonde2 bf_FruehKomplVentrikelsonde3 bf_FruehKomplAndereSonde)) denominator <- sum_results(import_results(module = "09/4", id = 50001), import_results(module = "09/5", id = 50002)) quotient_indicator(numerator = numerator, denominator = denomi- nator, units_from = "denominator") </pre> ID 52009 (DEK) - O/E Dekubitus: <pre> quotient_indicator(numerator = evaluate(bf_DEKDatensatzPlausibel & bf_DEKGrad_4 & bf_DEKnichtPOA), denominator = import_results("RST", 24851)) </pre>
-----------	---

ID 52010 (DEK) - Sentinel-Event Dekubitus Grad 4:

```
quotient_indicator(  
  numerator = evaluate(bf_DEKDatensatzPlausibel &  
    bf_DEKGrad_2bis4 & bf_DEKnichtPOA),  
  denominator = import_results("RST", 24851),  
  expected = import_results("RST", 23014),  
  units_from = "denominator"  
)
```

7 Follow-Up-Kennzahlen nach Anlage 3 QSKH

Mit den folgenden Funktionen werden die Follow-Up-Kennzahlen nach Anlage 3 QSKH abgebildet. Details zur Implementierung der statistischen Methodik sind in IQTIG 2017 [*Ereigniszeitanalyse-Methodik für die Follow-up-Indikatoren nach QSKH-RL*. Stand: 06.04.2017. Berlin: IQTIG. URL: https://www.iqtig.org/downloads/berichte/2017/IQTIG_Ereigniszeitanalyse-Methodik-für-Follow-up-Indikatoren-nach-QSKH-RL_2017-04-06.pdf] veröffentlicht.

7.1 Follow-Up Typ "Rate", "O/E" und "Sentinel Event"

follow_up_rate follow_up_oe follow_up_sentinel	Gibt die Ergebnisse eines Follow-Up-Indikators vom Typ "Rate", "O/E" bzw. "Sentinel-Event" nach Anlage 3 QSKH zurück.
Syntax	<code>follow_up_rate(dataset, numerator, denominator)</code> <code>follow_up_oe(dataset, numerator, denominator)</code> <code>follow_up_sentinel(dataset, numerator, denominator)</code>
Parameter	dataset – Der der Follow-Up-Berechnung zu Grunde liegende Datensatz. Für <code>follow_up_oe</code> muss der Datensatz eine Spalte <code>expected_events</code> enthalten oder der Parameter "expected_events" im Funktionsaufruf muss eine Spalte für die Risikoadjustierung definieren. numerator – Die Zähler-Rechenregel als Code. Darin muss ein Term der Form Beobachtungszeit %<=% ... enthalten sein, der den Follow-Up-Zeitraum spezifiziert. denominator – Die Nenner-Rechenregel als Code.
Rückgabewert	Das Ergebnis des Follow-Up-QIs als <code>data.frame</code> .
Besonderheiten	-
Beispiele	Follow-Up Typ Rate: <pre>follow_up_rate(dataset = get_dataset_by_name("FU2018MKEP"), denominator = OPDATUM_IND >= as.Date('2016-01-01') & OPDATUM_IND <= as.Date('2016-12-31') & alter_IND >= 18 & ENTLGRUND_IND != '07', numerator = Status & Beobachtungszeit %<=% 90 & bf_isolierterWechsel_FU != TRUE)</pre>

Follow-Up Typ O/E:

```
data <- get_dataset_by_name("FU2018MHEP")
data[["expected_events"]] <- 0.0019
follow_up_oe(
  dataset = data,
  denominator = OPDATUM_IND >= as.Date("2016-01-01") &
    OPDATUM_IND <= as.Date("2016-12-31") & alter_IND >= 18 &
    ENTLGRUND_IND != "07",
  numerator = Status & Beobachtungszeit %<=% 90 &
    bf_isolierterWechsel_FU != TRUE,
  expected_events = "expected_events"
)
```

Follow-Up Typ Sentinel Event:

```
follow_up_sentinel(
  dataset = get_dataset_by_name("FU2018M09N1"),
  denominator = OPDATUM_IND >= as.Date("2015-01-01") &
    OPDATUM_IND <= as.Date("2017-12-31") & alter_IND >= 18 &
    ENTLGRUND_IND != "07" & ASMSYSTEM_IND %in% c(1, 2, 3, 4),
  numerator = AGGW_FU == TRUE & Beobachtungszeit %<=% 1460
)
```

8 Hilfsfunktionen für O/E-Indikatoren

Hilfsfunktionen, um aus Ergebnissen von O / E-Indikatoren die Ergebnisse der zugehörigen kalkulatorischen Kennzahlen zu berechnen.

as_o_indicator_result as_e_indicator_result	Berechnet aus den Ergebnisinformationen eines O / E-Qualitätsindikators das Ergebnis der zugehörigen O-Kennzahl bzw. E-Kennzahl
Syntax	as_o_indicator_result(result) as_e_indicator_result(result)
Parameter	result – das Ergebnis eines O / E-Indikators
Rückgabewert	Das Ergebnis der zugehörigen O-Kennzahl (Zähler: O, Nenner: Anzahl Grundgesamtheit) bzw. E-Kennzahl (Zähler: E, Nenner: Anzahl Grundgesamtheit).
Besonderheiten	-
Beispiel	ID 52013 (DEK) – O-Kennzahl: result <- import_indicator(module = "DEK", id = "52009") as_o_indicator_result(result)